SPRINGER
REFERENCE

Bruce G. Batchelor
Editor

Machine Vision Handbook

Springer

Machine Vision Handbook

Bruce G. Batchelor (Ed.)

Machine Vision Handbook

With 1295 Figures and 117 Tables

Springer

Editor
Bruce G. Batchelor
Professor Emeritus
School of Computer Science and Informatics
Cardiff University
Cardiff, Wales
UK

Please note that additional material for this book can be downloaded from
http://extras.springer.com

ISBN 978-1-84996-168-4		e-ISBN 978-1-84996-169-1
DOI 10.1007/978-1-84996-169-1
ISBN Bundle 978-1-84996-170-7
Springer London Dordrecht Heidelberg New York

Library of Congress Control Number: 2011942208

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Rejoice in the Lord always.
I will say it again: Rejoice!
Let your gentleness be evident to all.
The Lord is near.
Do not be anxious about anything,
but in everything, by prayer and petition,
with thanksgiving, present your requests to God.
And the peace of God, which transcends all understanding,
will guard your hearts and your minds in Christ Jesus.

For those I hold dearest: my wife Eleanor, mother Ingrid, daughter Helen,
son David and my seven grandchildren.

Preface

When people ask me what work I do, I say that I am an academic engineer, studying *Machine Vision* and therefore call myself a *Machine Vision Systems Engineer* (*Vision Engineer for short*). When asked to explain what that means, I reply in the following way:

"I study machines that can 'see'. I use a video camera, connected to a computer, to find defects in industrial artifacts as they are being made."

In my experience, this is just short enough to avoid total boredom in fellow guests at dinner parties. However, it is woefully inadequate, since it does not encompass many topics that are essential for building a successful Machine Vision system. Compare the "dinner-party definition" with the following, more formal statement, which introduces the broader concept of *Artificial Vision*:

Artificial Vision is concerned with the analysis and design of opto-mechanical-electronic systems that can sense their environment by detecting spatio-temporal patterns of electro-magnetic radiation and then process that data, in order to perform useful practical functions.

Machine Vision is the name given to the *engineering* of Artificial Vision systems. When browsing through the technical literature, the reader will soon encounter another term: *Computer Vision (CV)* and will quickly realise that it and *Machine Vision (MV)* are often used synonymously. This is a point on which I strongly disagree! Indeed, this book is founded on my firm belief that these subjects are fundamentally different. There is a certain amount of academic support for this view, although it must be admitted that the loudest voices of dissent come from within universities. On the other hand, most designers of industrial vision systems implicitly acknowledge this dichotomy, often by simply ignoring much of the academic research in CV, on the grounds that it is not relevant to their immediate needs. In the ensuing pages, I and my co-authors argue that *Machine Vision* should be recognised as a distinct academic and practical subject that is fundamentally different from Computer Vision.

Computer Vision, Artificial Intelligence, Pattern Recognition and Digital Image Processing (DIP) all contribute to MV, which makes use of algorithmic and heuristic techniques that were first devised through research in these other fields. Machine Vision concentrates on making them operate in a useful and practical way. This means that we have to consider all aspects of the system, not just techniques for representing, storing and processing images inside a computer. With this in mind, we come to the following working definition.

Machine Vision is concerned with the engineering of integrated mechanical-optical-electronic-software systems for examining natural objects and materials, human artifacts and manufacturing processes, in order to detect defects and improve quality, operating efficiency and the safety of both products and processes. It is also used to control machines used in manufacturing. Machine Vision requires the harmonious integration of elements of the following areas of study

- *Mechanical handling*
- *Lighting*
- *Optics (conventional, fibre optics, lasers, diffractive optics)*
- *Sensors (video cameras, UV, IR and X-ray sensors, laser scanners)*
- *Electronics (digital, analogue and video)*

- *Signal processing*
- *Image processing*
- *Digital systems architecture*
- *Software*
- *Industrial engineering*
- *Human-computer interfacing*
- *Control systems*
- *Manufacturing*
- *Existing work practices and QA methods.*

Machine Vision has also been applied successfully to several high-volume niche markets, including:

- Reading automobile registration plates
- Face recognition (security purposes)
- Fingerprint recognition
- Iris recognition
- Document processing
- Signature verification
- Security surveillance
- Print inspection
- Fabricating micro-electronic devices
- Karyotyping (chromosome identification and classification)
- Inspecting bare printed cicuit boards
- Controlling/checking the placement of components on printed circuit boards.

However, we shall not discuss these niche application areas, preferring instead to concentrate on manufacturing industry, which is distinctive in presenting *many millions* of low-volume potential applications. These vary from trivial to extremely difficult. Rather than solving one application at a time, I have spent my working life devising tools that are appropriate for a very wide range of situations. The aim of my work over the last 35 years has been to build versatile development systems that can be used in the study of newly defined industrial inspection, measurement or control tasks. Using such systems enables new applications to be solved, or dismissed as impractical, in a very short time. There is a never-ending stream of new industrial applications. In the 1980s, the 3M Company, with whom I was privileged to work, introduced more than 10 new products each day! The challenge we faced was to devise tools allowing the potential benefits of MV to be explored quickly and cheaply; we could spend no more than a few hours evaluating each new application. Contrast this with the niche markets mentioned above, where there is the potential for a high financial return for a single design. Unlike most industrial MV projects, these niche markets were able to justify the high design costs needed.

Let us be more specific about what we include within the scope of our discussion in this book. We concentrate on manufacturing industry: "metal bashing" and processing materials such as plastics, wood, glass, rubber, and ceramics. A similar approach is appropriate in the manufacture and packaging of food, beverages, pharmaceuticals, toiletries, clothing and furniture. Many of the same ideas are also applicable to processing fabrics, leather, minerals, plant materials, fruit, vegetables, nuts, timber, animal and bird carcasses, meat, fish, etc.

The problem of nomenclature arises because MV, CV and DIP are all concerned with the processing and analysis of pictures within electronic equipment. I did not specify computers explicitly, because Artificial Vision does not necessarily involve a device that is recognizable as a computer. Of course, it is implicit in the very name of CV that image processing takes place inside a computer. On the other hand, MV does not impose such a limitation. For example, it also allows the processing of images to take place in specialised electronic or electro-optical hardware.

Although MV, CV and DIP share a great many terms, concepts and algorithmic techniques, they require a completely different set of priorities, attitudes and mental skills. The division between MV and CV reflects the division between engineering and science. Engineers have long struggled to establish a clear distinction between engineering and science. This book tries to do the same for MV Systems Engineering and its scientific-mathematical counterpart: Computer Vision.

CV is usually concerned with general questions, such as *"What is this?"*. On the other hand, MV almost always addresses very specific questions, for example *"Is this a well made widget?"*. This reflects one very simple fact: in a factory, we know what product is being made at any given moment, so we know what an object should look like. As a result, MV systems are usually used for *verification*, not *recognition*. The reverse is often true for CV. As we shall see in the following pages, this has profound implications.

The people responsible for generating MV data (i.e. images) are usually cooperative. A manufacturing engineer *wants* to produce a good product and will, as far as is practical, modify the inspection millieu, or even the product itself, to improve matters. Contrast this with the task of designing systems for *recognising* military targets. The enemy does not want to cooperate and actually tries to obscure his presence and actions! Similar remarks apply to fingerprint identification and forensic image analysis. Not surprisingly, criminals show little interest in cooperating with those people who are trying to catch them! In medicine, the situation is little better; tumours do not cooperate with equipment designers, although a battery of techniques (e.g. staining) has been developed for highlighting the presence of cancer cells. On a continuum scale recording the degree of cooperation that we can expect, MV applications are clustered at one end and CV applications near the other.

In MV, the allowed unit cost for analysing each image is likely to be very low. In CV applications, such as military target recognition and oncology, the cost of making a mistake can be very high indeed, so a higher unit cost per scene inspected can often be tolerated. Most CV problems are "difficult" in terms of artificial intelligence, while MV is usually "easy".

The distinction between MV and CV can be illustrated by considering a seemingly simple task: determining whether a coin has its obverse, or reverse, face uppermost. (This might well be a component in the bigger process of inspecting proof-quality coins.) CV begins with an image that was obtained by somebody else, not the CV specialist. On the other hand, MV begins with an embossed metal disc. Deriving an image from it requires that the MV engineer designs the lighting, optics and mechanical handling of the coins, as well as choosing an appropriate camera. The crucial importance of lighting and optics is obvious from the fact that very different images of the same coin can be produced, simply by moving the lights very slightly. In the MV engineer's mind, consideration of the representation, storage, processing and analysis of images follows once a good image has been produced. Computer processing of images for MV is important but it is not the vision engineer's only concern. He cannot afford to concentrate on this, or any other part of a system, and then neglect all others. *This is the primary and inviolate rule for Machine Vision system design.*

During the 1970s and 1980s, Japan taught the rest of the world the importance of ensuring the highest quality in manufactured goods. The West learned the hard way: markets were lost to companies whose names were hitherto unknown. Many long-established and well-respected companies were unable to meet the challenge and failed to survive. Those that did were often faced with difficult years, as their share of the market shrank. Most companies in Europe and America have come to terms with this now and realize that quality has a vital role in establishing and maintaining customer loyalty. Hence, any technology that improves or simply guarantees product quality is welcome. Machine Vision has much to offer manufacturing industry in improving product quality and safety, as well as enhancing process efficiency and operational safety. Machine Vision owes its rising popularity to the fact that optical sensing is inherently clean, safe, and versatile. It is possible to do certain things using vision that no other known sensing method can achieve. Imagine trying to detect stains, rust or surface corrosion over a large area by any means other than vision!

The recent growth of interest in industrial applications of Machine Vision is due, in large part, to the falling cost of computing power. This has led to a proliferation of vision products and industrial installations. It has also enabled the development of cheaper and faster computing machines, with increased processing power. In many manufacturing companies, serious consideration is being given now to applying Machine Vision to such tasks as inspecting grading, sorting, counting, monitoring, measuring, gauging, controlling and guiding. Automated Visual Inspection systems allow manufacturers to keep control of product quality, thus maintaining / enhancing their competitive position. Machine Vision has also been used to ensure greater safety and reliability of the manufacturing processes. The whole area of flexible automation is an important and growing one, and Machine Vision will continue to be an essential element in future *Flexible Manufacturing Systems*. There are numerous situations where human inspection is unable to meet production demands. The advent of Machine Vision has often enabled the development of entirely new products and processes. Many companies now realise that Machine Vision forms an integral and necessary part of their long-term plans for automation. This, combined with the legal implications of selling defective and dangerous products, highlights the case for using Machine Vision in automated inspection. A similar argument applies to the application of vision to robotics and automated assembly, where human operators were previously exposed to dangerous, unhealthy / or unpleasant working conditions. Machine Vision is a valuable tool in helping to avoid this.

No Machine Vision system existing today, or among those planned for the foreseeable future, approaches the interpretative powers of a human being. However, current Machine Vision systems are better than people at some repetitive quantitative tasks, such as making measurements under tightly controlled conditions. Machine Vision systems can out-perform people, in certain limited circumstances. Industrial vision systems can generally inspect simple, well-defined mass-produced products at very high speeds, whereas people have considerable difficulty making consistent inspection judgments in these circumstances. Machine Vision systems exist that can inspect several thousand objects per minute, which is well beyond human ability. Studies have shown that, at best, a human inspector can only expect to be 70–80% efficient, even under ideal conditions. On many routine inspection tasks, a Machine Vision system can improve efficiency substantially, compared to a human inspector. A machine can, theoretically, do this for 24 hours / day, 365 days / year. Machine Vision can also be useful in detecting gradual changes in continuous processes. Tracking gradual colour, shade or texture variations is not easy for a person.

Currently, the main application areas for industrial vision systems occur in automated inspection and measurement and, to a lesser extent, robot vision. Automated Visual Inspection and measurement devices have, in the past, tended to develop in advance of robot vision systems. In fact, QC-related applications, such as inspection, gauging and verfication, currently account for well over half of the industrial Machine Vision market. This has been achieved, in many cases, by retrofitting inspection systems onto existing production lines. There is a large capital investment involved in developing a completely new robotic work cell. Moreover, the extra uncertainty and risks involved in integrating two relatively new and complex technologies makes the development of robot vision system seem a daunting task for many companies and development has often lagged behind that of inspection devices. The technical difficulties involved in controlling flexible visually guided robots have also limited the development. On the other hand, Automated Visual Inspection systems now appear in every major industrial sector, including such areas as consumer goods, electronics, automobile, aerospace, food, mining, agriculture, pharmaceuticals, etc.

Machine Vision systems for industry first received serious attention in the mid-1970s. Throughout the early 1980s, the subject developed slowly, with a steady contribution being made by the academic research community, but with only limited industrial interest being shown. It seemed in the mid-1980s that there would be a major boost to progress, with serious interest being shown in vision systems by the major American automobile manufacturers. Then, came a period of disillusionment in the USA, with a large number of small vision companies failing to survive. In the late 1980s and early 1990s, interest grew again, due largely to significant progress being made in making fast, dedicated image processing hardware. During the first decade of the 21st century, the growth of the Machine Vision market was unrelenting. This was helped considerably by the development of new technologies, especially plastic optical components, LED lighting, high-resolution cameras and fast computer hardware. For many applications, it is possible now to provide sufficiently fast processing speed on a standard computing platform. Throughout the last 35 years, academic workers have demonstrated feasibility in a very wide range of products, representing all of the major branches of manufacturing industry. I have been very privileged to have participated in this.

I have watched this subject develop and blossom, from the first tentative steps taken in the mid-1970s to today's confident multi-billion dollar industry. In those early days, industrial engineers and managers were mentally unprepared for the new technology. Although there is far greater acceptance of vision system technology nowadays, many customers are still unwilling to accept that installing and operating a vision system places a new intellectual responsibility on them. They often fail to understand that machines and people see things very differently. During the last 35 years, I have seen similar mistakes being made, through a lack of appreciation of this point. Sadly, customer education is still lagging far behind technological developments in Machine Vision. The responsibility for making Machine Vision systems work properly does not lie solely on the shoulders of the vision engineer. The customer has a duty too: to listen, read, and observe, Making the most from the strengths and weaknesses of Machine Vision technology involves appreciating a lot of subtle points. This book discusses just a small proportion of these. My colleagues and I have had great fun developing and writing about this amazing technology and hope that some of our excitement will be evident in these pages.

It is customary in many areas of writing to use so-called gender-neutral constructions, such as "*he/she*", "*s/he*", "*his/her*" etc. I regard these as being both clumsy and counter-productive. To improve clarity and to avoid placing women after men, I use the words, "*he*", "*his*" and "*him*" in the traditional way, to encompass both sexes, without granting precedence to either.

I am indebted to my friends and colleagues who have contributed to this handbook. Without exception, their enormous experience is evident in the splendid essays that grace this description of this exciting subject. I was inspired to study Machine Vision systems when I was preparing a book (*Pattern Recognition: Ideas in Practice*, Plenum, New York & London, 1978, ISBN 0-306-31020-1) with invited contributions from expert authors. One of them, Dr. John R Parks, offered a chapter entitled *Industrial Sensory Devices*, which so excited me that 37 years later I am still conducting research in that area. It is a pleasure to acknowledge John Parks, for his foresight, technical knowledge and humour. This book describes an interactive image processing system, called *QT*, which has its origins, in part, to work in the 1970s on *SUSIE* (*Southampton System for Image Evaluation*) by Dr. Jonathan Brumfitt. Toby Holland developed the video interface hardware for *SUSIE*, which inspired the development of the *AutoView Viking* system. (British Robotic Systems Ltd.) Peter Heywood, Dr. Graham Page, Dr. David Mott and David Upcott all contributed to that. In parallel, Tim Loker, Mike Rippingale and Martin Reeves, of Vision Dynamics Ltd, developed a similar system called *VCS*. The *Intelligent Camera* (Image Inspection Ltd.) also followed from *SUSIE* and was developed by Dr. Piers Plummer. Dr. David Mott integrated *AutoView Viking* with *Prolog*. Inspired by this work, *VCS*, the *Intelligent Camera* and *QT* (described in detail here) were, in turn, interfaced to Prolog. Dr. Andrew Jones, Ralf Hack and Stephen Palmer wrote an integrated package, called *PIP*, incorporating *LPA Prolog* (Logic Programming Associates Ltd) and our own image processing software, written in C. I built *QT*, on top of the *MATLAB*™ *Image Processing Toolbox* (Mathworks, Inc). Recently, *QT* has itself been extended by Dr. Simon Caton, who developed a multi-host, multi-process version. Dr. Luke Chatburn developed a web-based, multi-computer system, called *Myriad*, for image processing. Dr. Frederick Waltz and I worked together for many years, applying interactive image processing to real-world industrial inspection, measurement and control tasks. Michael Daley, Garry Jackson and Richard Crowther all worked with me, interfacing image processing systems to various electro-mechanical devices, including our model train. In addition, Dr. Barry Marlow, Dr. Simon Cotter, Dr. Chris Bowman, Dr. Tony McCollum, Dr. John Chan, Dr. Mark Graves, Dr. Graham Wilson, Dr. Robin Tillet, Prof. Paul Whelan, and Rolf Iverson all helped to guide my thinking. I am pleased to express my gratitude to all of the former colleagues that I have mentioned above. I recall with great pleasure the fulfilling times that I have enjoyed working with them all. It was great fun! My special thanks are due to Tim Loker, Peter Heywood and Fred Waltz who opened doors for me.

This book would not have been completed without the unerringly patient support and encouragement of my dear wife Eleanor. Gratitude is not enough for her love!

My clumsy attempts at building machines that can "see" simply emphasizes how wonderful and mysterious my own sense of sight is. It is humbling to realise that our supposedly sophisticated technology is no match for the gift of sight that our Creator graciously chose to bestow on us. It is well to remember that we can at best supplement but not replace human vision.

Bruce G. Batchelor
Cardiff, Wales, United Kingdom
November 2011

Table of Contents

Volume 1

Volume 2

Volume 3

List of Contributors

Donald Bailey
School of Engineering and Advanced Technology
Massey University
Palmerston North
New Zealand

Bruce G. Batchelor
School of Computer Science and Informatics
Cardiff University
Cardiff, Wales
UK

Tim Carew
Centre for Image Processing and Analysis
School of Electronic Engineering
Dublin City University
Glasnevin, Dublin
Ireland

Simon J. Caton
Karlsruher Institut für Technologie (KIT)
Karlsruhe
Germany

E. Roy Davies
Department of Physics
Royal Holloway
University of London
Egham, Surrey
UK

Jonathan T. Erichsen
Cardiff School of Optometry and Vision Sciences
Cardiff University
Cardiff
UK

Andrew K. Forrest
London
East Finchley
UK

Ovidiu Ghita
Centre for Image Processing and Analysis
School of Electronic Engineering
Dublin City University
Glasnevin, Dublin
Ireland

David L. Gilblom
Alternative Vision
Tucson, AZ
USA

C. Griffith
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI
USA

P. Hamel
Metal Forming and Coining Corporation
Maumee, OH
USA

Spencer D. Luster
Light Works, LLC
Toledo, OH
USA

John W. V. Miller
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI
USA

T. Peatee
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI
USA

J. P. Sacha
Edison Industrial Systems Center
Toledo, OH
USA

Robert Sadleir
Centre for Image Processing and Analysis
Dublin City University
Glasnevin, Dublin
Ireland

Murat Sena
Visionex, Inc.
Green Bay, WI
USA

Behrouz N. Shabestari
Edison Industrial Systems Center
Toledo, OH
USA

Malayappan Shridhar
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI
USA

V. Shridhar
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI
USA

Robert Sweney
Electroplasma, Inc.
Millbury, OH
USA

Frederick M. Waltz
Department of Electrical and Computer Engineering (Retired)
University of Minnesota
Falcon Heights, MN
USA

Paul Watta
The University of Michigan-Dearborn
Dearborn, MI
USA

Paul F. Whelan
Centre for Image Processing and Analysis
School of Electronic Engineering
Dublin City University
Glasnevin, Dublin
Ireland

K. D. Whitehead
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI
USA

E. Wicke
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI
USA

J. Margaret Woodhouse
Cardiff School of Optometry and Vision Sciences
Cardiff University
Cardiff
UK

Dongming Zhao
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI
USA

1 Machine Vision for Industrial Applications

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_1,

Abstract: Machine Vision is related to, but distinct from Computer Vision, Image Processing, Artificial Intelligence & Pattern Recognition. The subject is concerned with the engineering of integrated mechanical-optical-electronic-software systems for examining natural objects and materials, human artifacts and manufacturing processes, in order to detect defects and improve quality, operating efficiency and the safety of both products and processes. It is also used to control machines used in manufacturing. Machine Vision necessarily involves the harmonious integration of mechanical handling, lighting, optics, video cameras, image sensors (visible, UV, IR and X-ray sensor arrays, as well as laser scanners), digital, analogue and video electronics, signal processing, image processing, computer systems architecture, software, industrial engineering, human-computer interfacing, control systems, manufacturing, existing work practices and quality assurance methods. Machine Vision is not a scientific endeavour; it is a branch of Systems Engineering. Hence, consideration of application requirements pays a key role in the design of practical vision systems. The basic philosophy of Machine Vision is set out in this chapter and the structure of this book is outlined. This is a pragmatic, empirical, experimental subject and is not unduly burdened by the search for mathematical rigour, or theoretical purity. There is one simple design maxim: *if it works, use it!* This is justified here by consideration of lessons learned from four practical application studies. Later in the book these same ideas will emerge many times over.

1.1 Natural and Artificial Vision

Vision is critical for the survival of almost all species of the higher animals, including fish, amphibians, reptiles, birds and mammals. In addition, many lower animal phyla, including insects, arachnids, crustacea and molluscs possess well-developed optical sensors for locating food, shelter, a mate, or a potential predator. Even some unicellular organisms are photosensitive, although they do not have organs that can form high resolution images. Vision bestows great advantages on an organism. Looking for food during foraging, or prior to giving chase, is very efficient in terms of the energy expended. Animals that look into a crevice in a rock, to check that it is safe to go in, are more likely to survive than those that do not do so. Animals that use vision to identify a potential breeding partner are more likely to find a fit, healthy mate than those that ignore its appearance. Animals that can see a predator approaching are more likely to be able to escape capture than those that cannot. Compared to other sensing methods, based on smell, sound and vibration, vision offers greater range, versatility, sensitivity and resolution.

Of course, vision is just as important to *Homo sapiens* as it is to any other animal species; life in both the forest and a modern industrial society would be impossible without the ability to see. Human society is organised around the highly refined techniques that people have developed to communicate visually. People dress in a way that signals their mood, social standing and sexual availability. Commerce is dependent upon hand-written, printed, and electronic documents that all convey messages visually. Education relies heavily upon the students' ability to absorb information that is presented visually. Writers of technical books, such as this, exploit the reader's ability to understand complex abstract ideas through the visual medium provided in printed text, diagrams and pictures. The leisure and entertainment industries rely on the fact that vision is the dominant human sense. So important is vision for our everyday lives that its loss is regarded by most people as one of the worst fates that could happen to a human being.

Manufacturing industry is critically dependent upon human beings' ability to analyse complex visual scenes. This is exploited in innumerable and highly variable tasks, ranging from the initial design process, through initial forming, machining, finishing, assembly, and quality control, to final packing. The value of vision as a safe way to sense many of the dangers that exist in a factory does not need explanation. Vision is hygienic and does not pollute the object under observation. With very few exceptions, the light levels needed to inspect an industrial artifact do not affect it in any way. Vision is potentially very fast; propagation delays from the object under inspection to sensor are typically just a few nanoseconds. (The sensor and its associated processing system will normally introduce much longer delays than this.) Vision provides a great deal of very detailed information about a wide variety of features of an object, including its shape, colour, texture, surface finish, coating, surface contamination, etc. Vision is extremely versatile and we can employ a wide range of optical techniques to widen its range of application even further. For example, it is possible to use stroboscopic light sources, structured lighting, colour filters, coherent illumination polarised light, fluorescence and many other clever optical "tricks" to sense a very broad range of physical properties.

Human vision is truly remarkable in its ability to cope with both subtlety and huge variations in intensity, colour, form and texture. The dynamic range is enormous: about 10^{14}:1. People can see extremely subtle changes of colour that are difficult to detect in a machine. (This is why it is so difficult to match colours after automotive body repair.) People can identity faces they know in a milling crowd, despite great changes in appearance that have taken place since the individuals concerned last met. On the other hand, they can register two matching pictures with great precision. A person can drive a car at high speed, yet a watchmaker can make very fine adjustments to a mechanism. A player can strike the cue ball in the game of billiards, or pool, with exquisite accuracy. A painter can capture the subtlety of colour in a woodland scene on his canvas, while an experienced foundry worker can identify the temperature of an incandescent ingot from its colour very accurately. All of these visual skills are exercised for inspection, grading, sorting, counting and a great many other tasks required for manufacturing industry.

Despite its obvious benefit to industry, human vision has its limitations (➔ *Fig. 1.1*). Many factors contribute to this, most notably fatigue, discomfort, illness, distraction, boredom, and alcohol/drug use. People cannot work at the speed required to provide 100% inspection of the products of many fast manufacturing processes. Boring repetitive tasks, such as examinig high-speed conveyors carrying unstructured heaps of raw materials, are particularly prone to unreliable performance. Many inspection tasks require precise dimensional/volumetric measurement, which people cannot perform accurately. For safety's sake, people working in factories must not be exposed to high temperatures, toxic/stifling atmospheres, high levels of smoke, dust and fumes, biological hazards, risk of explosion, excessively high noise levels, and ionizing radiation. On the other hand, people can damage delicate products, by clumsy handling and readily infect biological materials, by coughing, sneezing, shedding skin and hair. Furthermore, people cannot always apply objective inspection criteria rigorously, particularly when aesthetic properties of highly variable items are to be judged. For these reasons, engineers have long dreamed of building a machine that can "see." Such a machine should be able to sense its environment optically, using a video camera, or other imaging sensor, interfaced to an electronic information-processing system: a standard computer, or dedicated electronic hardware.

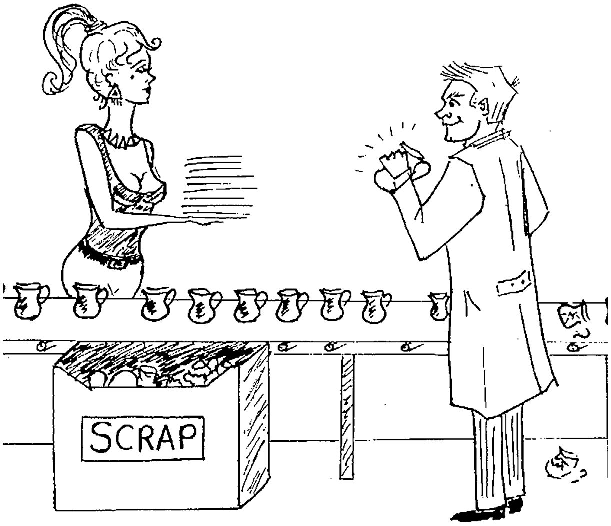


Fig. 1.1
Human beings are easily distracted from boring tasks. Almost anything, for example attractive members of the opposite sex, are more exciting to look at than objects moving past on an industrial conveyor belt! (This diagram simply reflects the reality of life and is not intended to denigrate either sex)

1.2 Artificial Vision

For the moment, we will use the phrase *Artificial Vision*, since we do not yet want to get into a detailed discussion about the precise meanings of the more commonly used terms *Computer Vision* and *Machine Vision*. The application of Artificial Vision systems to manufacturing industry motivates the techniques described in this handbook. This is an area where machines are beginning to find application, in preference to people. Artificial Vision does not necessarily attempt to emulate human, or animal, vision systems. Indeed, the requirement that industrial vision systems must be fast, cost effective and reliable, together with our rather limited knowledge about how people and animals actually see, make this approach unprofitable. By studying vision in nature, we may gain insight but we must be flexible and pragmatic when designing industrial vision systems.

When the author of this chapter is asked by a non-technical person what he does for a living, he replies that, as an academic, he studies *machines that consist of a television (more properly, video) camera connected to a computer and uses them to inspect objects as they are being made in a factory*. Such a definition is acceptable around the dinner-table but not for a technical book, such as this (➲ *Fig. 1.2*). In fact, designing a vision system that is of practical value in manufacturing applications requires a multi-disciplinary approach, encompassing aspects of all of the following technologies:

- Sample preparation
- Sample presentation to the camera
- Illumination

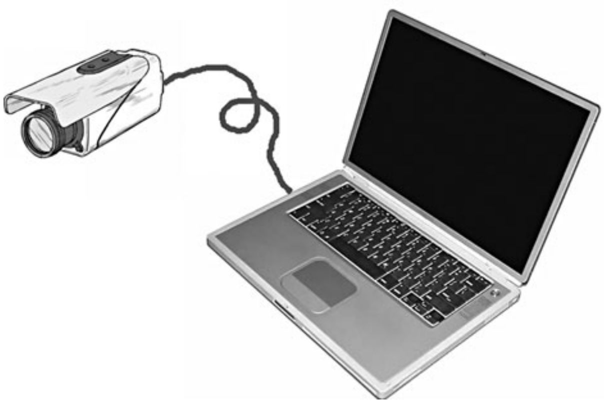


Fig. 1.2
This is not a Machine Vision system! The usefulness of pictures like this is limited to introducing total novices to image processing

- Optics
- Image sensing
- Analogue signal processing
- Digital information processing
- Digital systems architecture
- Software
- Interfacing vision systems to other machines
- Computer networking
- Ergonomics: interfacing vision systems for easy use by operating personnel
- Manufacturing and Quality Assurance practices.

These points must also be considered when a human being performs a visual inspection task. *Figure 1.3* illustrates this. However, the real significance of this cartoon is that the same features are needed in an automated visual inspection system. To justify the remainder of these points, in the context of Artificial Vision, the reader is referred to *Fig. 1.4*, which illustrates a range of typical industrial vision applications.

Lighting is obviously of critical importance when inspecting transparent, glossy and glinting artifacts, such as glass bottles, coins and electric light bulbs. Specialised lighting and viewing methods are needed to examine inside deep holes or crevices. In some situations, we may need to use sensing techniques based on ultra-violet, infra-red, or x-ray imaging, to detect details that we could not otherwise see. The safety implications of using ultra-violet and penetrating radiation are of crucial importance, since even the most technically successful project will be judged a failure, if the equipment does not meet certain minimum legal safety requirements.

Mechanical, rather than human handling of the parts to be inspected is vital when the process would place the operator in danger, for example when x-ray sensing is used or inspecting 25 ton slabs of red-hot steel.

We may need to apply a high level of intelligent interpretation to analyse images of natural products, such as growing plants, since they are totally unpredictable in form. Inspecting cake decoration patterns also requires intelligent analysis, because they are highly variable, as are the

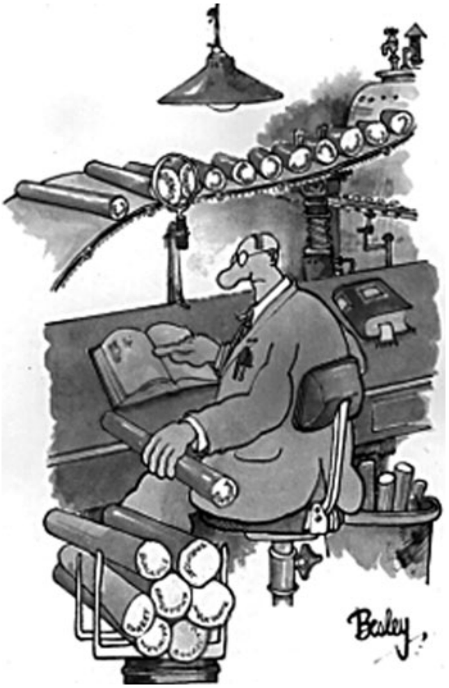


Fig. 1.3

The essential elements of typical Machine Vision system translated into human terms. Notice the presence of the following: object transport mechanism (*conveyor belt*), dedicated lighting, optics (*spectacles*), image sensor (*eyes*), product information database (*book*), data processing unit (*brain*), device for rejecting faulty product (*left arm*), reject bin. (Original caption: "*Checking the spelling of sea-side rock.*" Rock is a confectionary bar, typically with the name of a holiday resort spelled out by lettering that runs internally along its whole length. Reproduced by kind permission of the artist, Rupert Besley)

shapes of baked products, such as loaves, cakes, croissants, etc. Intelligence is also needed to determine the time from an analogue clock. Although not important in itself these days, this task is a model of a range of instrument calibration tasks.

Complex analytical methods of a different type are needed to analyse the surface texture of a slice of bread, or cork floor tiles. Certain tasks may apparently require complicated solutions, if we do not pay due attention to the application requirements. However, the solution may be quite straightforward if we constrain the task in some way. For example, an electrical connector (*Fig. 1.5*) would be difficult to inspect, if it were presented haphazardly to the camera. On the other hand, if it is delivered to the camera in a well-controlled manner (i.e., its position and orientation are both fixed), the inspection task becomes almost trivial.

In some applications, defining the inspection task unambiguously clearly requires a sophisticated user interface. For example, to define suitable criteria for inspecting the hydraulics manifold shown in *Fig. 1.4a* (second row/second column), we would need to tell the inspection machine which holes to measure, how large they are, whether they are

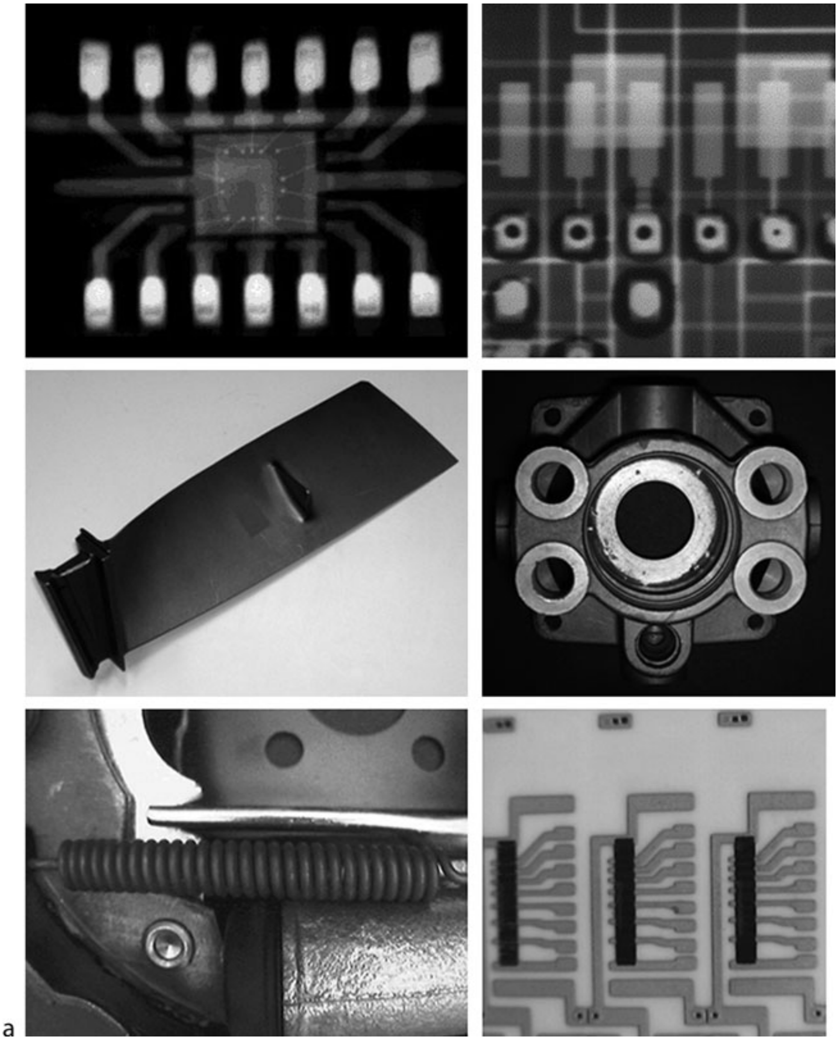


Fig. 1.4 (Continued)

tapped and chamfered and how their positions relate to one another. This requires close interaction between the user and the inspection machine. Sometimes detailed information about a product's specification is available in the form of a CAD model. Interpreting this to guide an inspection system clearly requires sophisticated software.

In our present state of knowledge about this subject, it is absolutely essential that we tailor each system to the specific demands of the application. In this respect, the design process is driven by the application requirements in the broadest sense, not by the computational procedures alone. We cannot, for example, design a "universal" vision system. Neither can we devise a system that can tolerate uncontrolled lighting, ranging from moon-light, through standard room-lighting to sun-light. We *must* control the light falling on the scene that is to be examined. We do this in two ways:

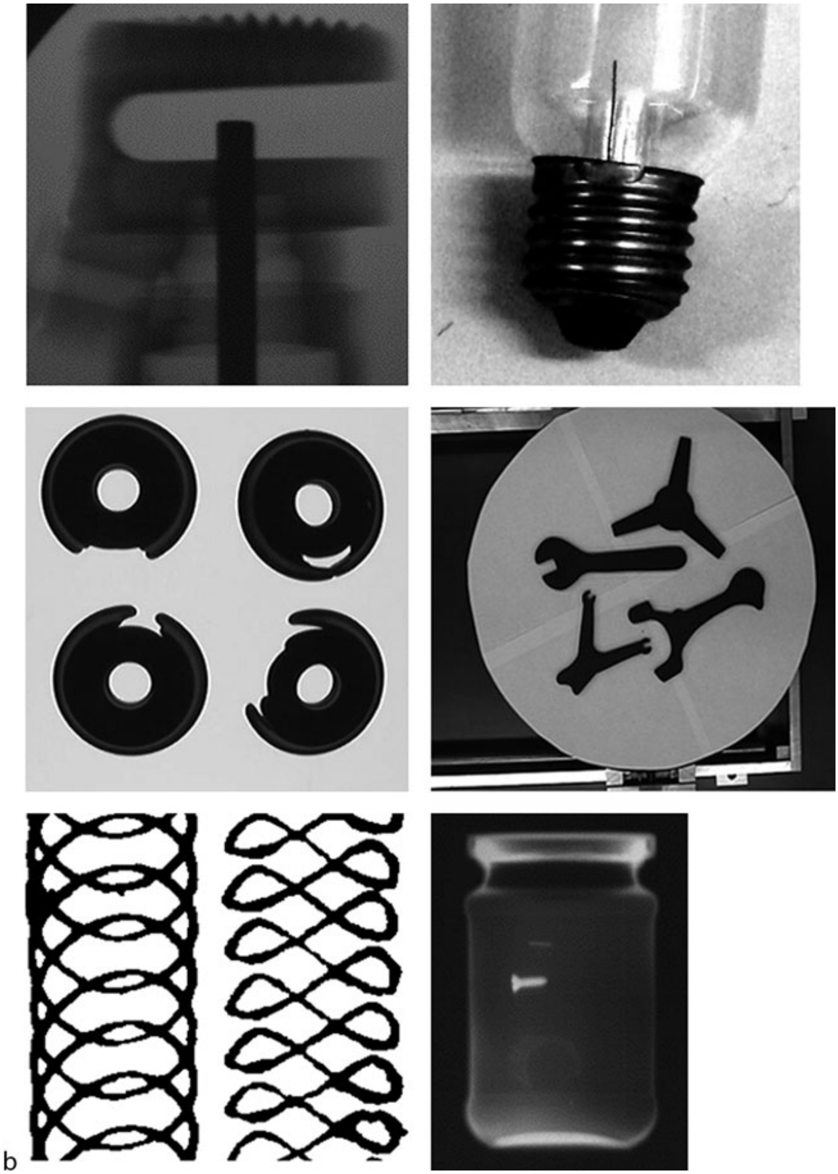


Fig. 1.4 (Continued)

1. Use curtains, boards, or other rigid sheeting to block ambient light.
2. Apply light from stable lamps, driven by regulated power supplies. Feed-back control of the light levels is highly desirable.

Nevertheless, we should, wherever possible, use illumination invariant image processing algorithms as well (see Chap. 13).

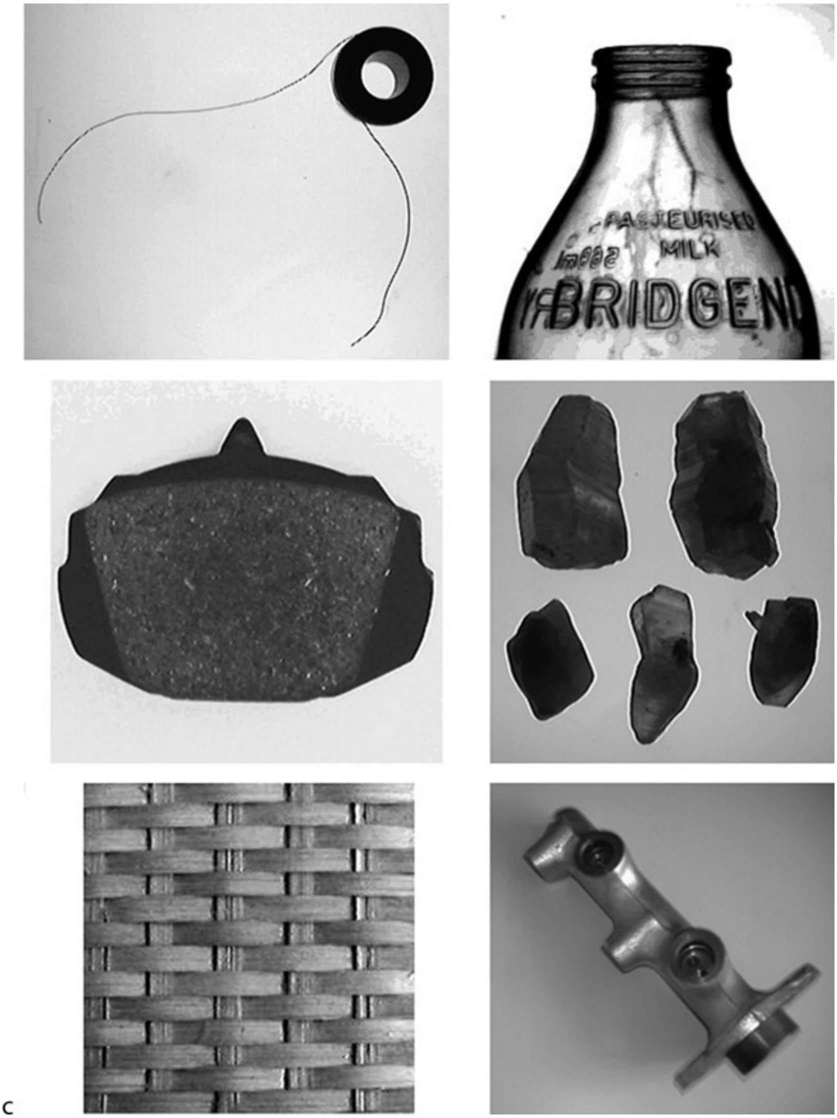


Fig. 1.4 (Continued)

1.3 Machine Vision Is Not Computer Vision

During a recent conversation, the author discussed the meanings of the terms *Machine Vision* (*MV*) and *Computer Vision* (*CV*), with another researcher, who is a very able and active worker in the mathematical and algorithmic aspects of Computer Science. (That is, he works in what the author maintains is *Computer Vision.*) He regards the terms *Machine Vision* and *Computer Vision* as being synonymous. In order to highlight the difference between them, the author

Fig. 1.4 (Continued)

challenged his colleague to design a vision system that is able to decide whether a bright “silver” coin is lying with its obverse or reverse side uppermost. He could not do so! The reason for this is simple: this very intelligent man lacks training in an appropriate experimental subject (e.g., engineering or physics). Since the starting point for this challenge is a coin, not a computer file, the lighting and optical sub-systems must be designed first, before any work on image processing algorithms is contemplated. We will present several further anecdotes to support our contention that MV requires skills and knowledge that CV largely ignores. We do not intend to belittle the intellectual component of Computer Vision. On the contrary, it is our

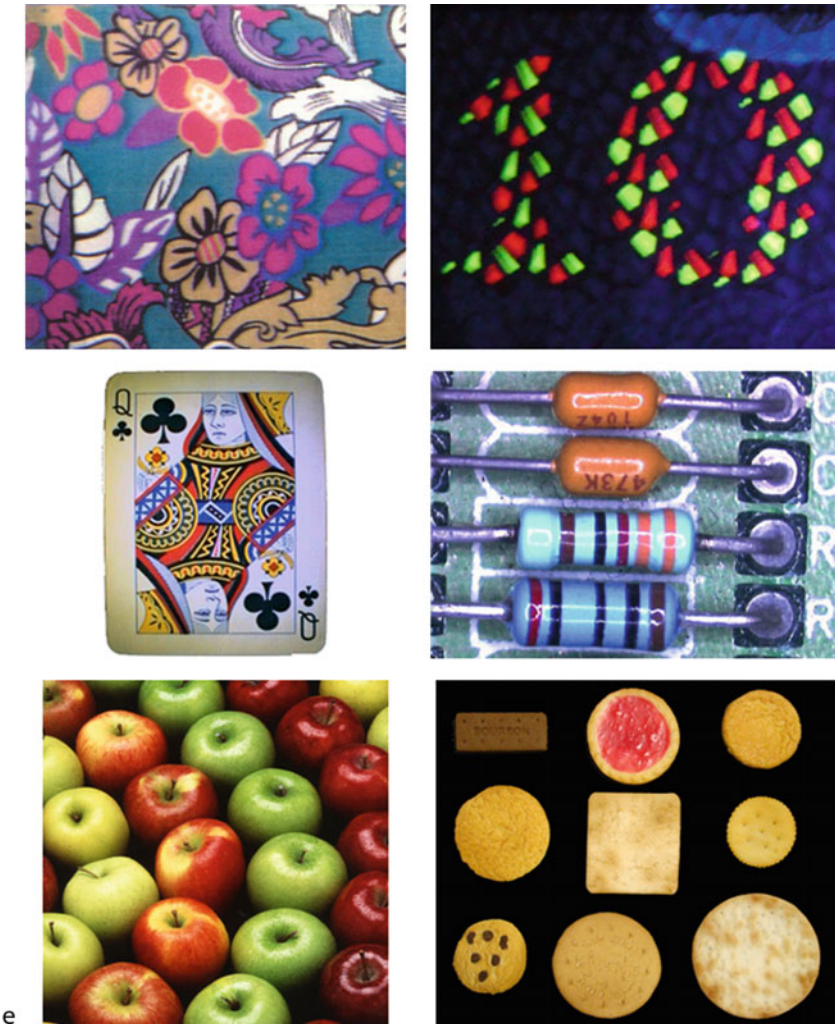


Fig. 1.4
A sample of the industrial applications of Machine Vision that the author has encountered during his career (5 pages). Further details are given in *Table 1.1*

belief that CV is intellectually “deep,” while Machine Vision requires a broader range of skills. There should be mutual respect between the practitioners of these two subjects. Unfortunately, this is not always so. We will argue that MV should be regarded as a distinct academic and commercial subject, deserving its own institutions for organising meetings, publishing results, honouring its leading exponents and with its own funding programme for research and advanced development.

There is a natural assumption among non-specialists that MV and CV are synonymous, because they both refer to Artificial Vision. In addition, many researchers blur the

Table 1.1

Details of the machine vision applications illustrated in ➋ *Fig. 1.4*. The *left-most column* (labelled R/C) indicates the row/column position for the sub-images. In the *right-most column* of this table, safety critical inspection tasks are indicated by ''Yes,'' while possible indirect safety concerns are signalled with ''Maybe.'' For example, PCBs are used in a variety of situations, including safety critical applications. Numerous other examples are given in [1– 6]

R/C	Description	Check/measure/operation	Safety
(a)			
1/1	Integrated circuit, x-ray	Bonding of wires	Maybe
1/2	Bare printed circuit board	Conductor integrity	Maybe
2/1	Turbine/compressor blade	Precise dimensions	Yes
2/2	Hydraulics manifold (casting)	Internal threads, debris, swarf, etc.	Yes
3/1	Automobile brake unit	Correct assembly	Yes
3/2	Thick film circuit	Component assembly	Maybe
(b)			
1/1	Aerosol assembly, x-ray	Complete/correct assembly	Yes
1/2	Light bulb	Loose wires	Yes
2/1	Battery tops	Moulding integrity	
2/2	Robotics (XYθ-table)	Automated handling	
3/1	Cake decoration patterns	Cosmetic appearance	
3/2	Food container (glass jar)	Foreign bodies	Yes
(c)			
1/1	Coil with flying leads	Automated assembly	
1/2	Glass bottle	Cracks, splinters, moulding faults	Yes
2/1	Automobile brake pad	Correct type, integrity of assembly	Yes
2/2	Uncut gems (sapphires)	Sorting/grading/measurement	
3/1	Rattan cane web for chairs	Appearance and integrity	
3/2	Car brake cylinder	Internal scratches on bore surface	Yes
(d)			
1/1	Aerosol spray jet	Measurement of jet	
1/2	Plant, micropropagation	Automated dissection	
2/1	Food containers (6 cans on a tray)	Foreign bodies (internal)	Yes
2/1	Sports shoe upper	Automated stitching	
3/1	Diamond grind wheel	Automated placing of diamonds	
3/1	Shoe components	Sorting	Yes
(e)			
1/1	Dress fabric	Colour imprint alignment	
1/2	Fluorescent security marking	Verify presence/shape	Maybe
2/1	Packaging (representative image)	Colour imprint alignment	Maybe
2/2	Populated PCB	Assembly of components	Maybe
3/1	Fruit	Grading, counting, sorting	
3/2	Baked goods	Size, shape, filling, contamination	Yes

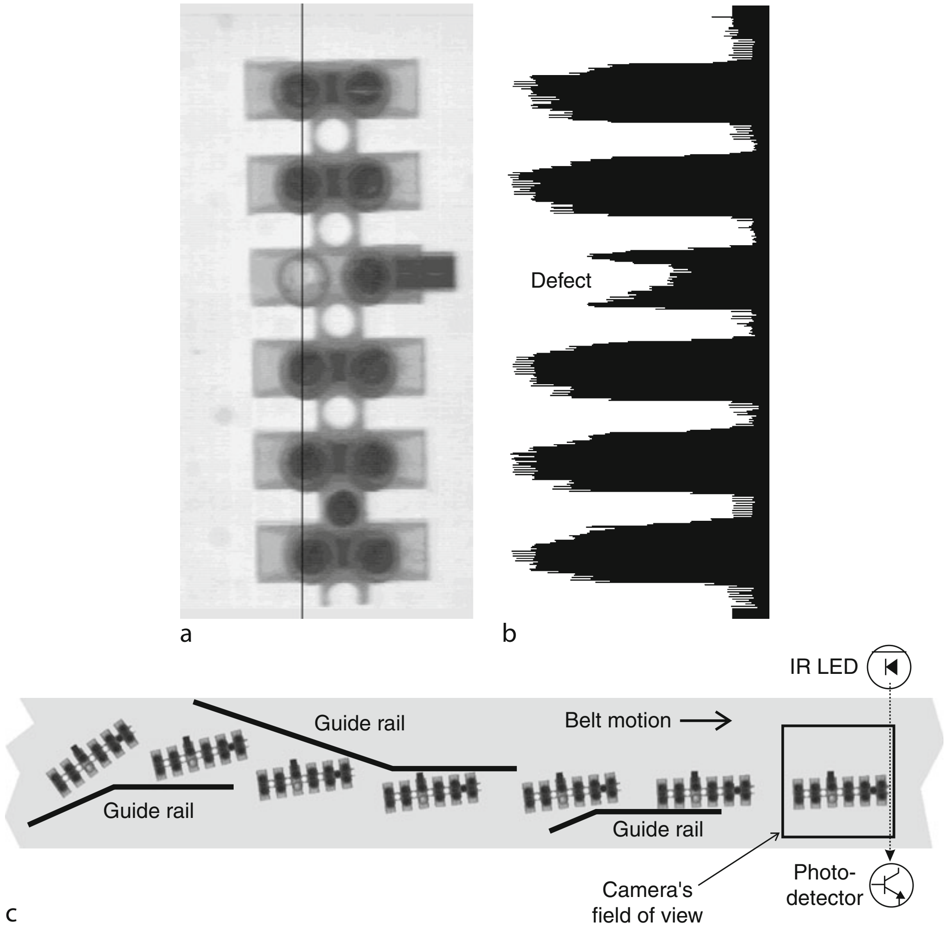


Fig. 1.5
Making inspection easier by standardising the orientation and orientation of an object. The alternative may be some quite slow and/or costly image analysis. (a) Electrical connector block. (b) A simple intensity profile along the vertical black line in (a) is able to detect the defect. (c) Using guide rails on a continuously moving conveyor belt limits the variation of orientation. The horizontal position is "fixed" using the photo-electric proximity sensor which triggers the image inspection process

boundaries separating *Machine Vision* from the subjects of *Artificial Intelligence (AI)*, *Digital Image Processing (DIP, or simply IP)* and *Pattern Recognition (PR*, alternatively called *Neural Networks)*. Of these, DIP is most often confused with MV because it too is concerned with Artificial Vision. Photoshop™ and similar packages perform DIP, although this is not the limit of its scope, which also encompasses tasks such as contrast enhancement of medical x-rays and images from security cameras. DIP is an essential component of both CV and MV, which is the cause of the confusion between these terms. PR and to a smaller extent AI, are sometimes perceived as encompassing aspects of MV, although the distinction is greater than for CV. Several other terms need to be mentioned before we begin our discourse in earnest. *Automated Visual Inspection (AVI)*, *Robot Vision* and *Visual Process Control/Monitoring* are sub-sets of MV. However, the term *Robot Vision* is also used when referring to the theoretical aspects of robot control, as a sub-set of AI.

All of the terms mentioned above are frequently used in an imprecise way, by both writers and speakers at conferences. There is no doubt that the sloppy use of these terms has been the cause of a considerable amount of confusion, in the past. This has often resulted in unsatisfactory service being provided to would-be customers of vision systems. It has also hampered research, and has probably hindered career progression of engineering practitioners.

The problem of nomenclature arises because MV, CV, DIP and sometimes AI and PR are all concerned with the processing and analysis of pictures using electronic equipment. Notice that we did not mention computers explicitly, since not all processing of pictures takes place within a device that is recognisable as a digital computer. Of course, CV assumes that these activities do take place inside a computer. On the other hand, MV does not impose such a limitation, although computer-based processing of images is encompassed by this term. MV allows the processing of images to take place in specialised digital networks, arrays of field-programmable gate arrays (FPGAs, ➲ Chap. 25), multi-processor configurations, optical/opto-electronic computers, analogue electronics, or hybrid systems.

It must be understood that MV, CV and DIP share a great many terms, concepts and algorithmic techniques but they have a completely different set of attitudes and priorities (see ➲ *Table 1.2*). To summarise, the division between MV and CV reflects the separation that exists between engineering and science. Just as engineers have long struggled to establish the distinction between their activities and science, we are trying to do the same for MV and its scientific/ mathematical counterpart (i.e., CV). See ➲ *Table 1.3*.

This confusion of terms is even manifest in the names of learned societies. For example, the *British Machine Vision Association* holds a series of conferences in which almost all of the papers are in what the author maintains is CV. There is certainly a great difference between the articles included in the proceedings of the BMVA on the one hand and those organised by *SPIE – The International Society for Optical Engineering*. Indeed, it is doubtful whether many of the papers that are acceptable to a BMVA audience would be welcomed at an SPIE conference, or vice versa. This fact does not, in any way, detract from the worth, or intellectual content of either "camp"; it merely reflects a difference of approach and priorities. In the case of Machine Vision, the latter are usually manifest in terms of the design criteria imposed by the application.

1.3.1 Non-industrial Applications

In this handbook, we concentrate our attention on industrial applications of vision system, for such tasks as inspecting, measuring, verifying, identifying, recognising, grading, sorting, counting, locating and manipulating parts, monitoring/controlling production processes, etc. Vision systems may be applied to raw or processed materials, piece parts, assemblies, packing, packaging materials and structures, continuous-flow products (webs, extrusions, etc.), simple and complex assemblies, or to tools and manufacturing processes. They can also be applied to complex systems (e.g., complicated physical structures, pipe-work, transport

Table 1.2

Machine vision compared to computer vision. Entries in the machine vision column relate to factory-floor target vision systems (TVS), unless otherwise stated. IVS denotes Interactive Vision System

Feature	Machine vision	Computer vision
Cost	Critical. Cost-effectiveness is central consideration	Secondary importance compared to performance
Dedicated electronic hardware?	Possible. This is becoming less common as CPU speeds increase.	No. Required flexibility and intelligence require software
Engineering or science?	Systems Engineering, practical, pragmatic, experimental	Computer Science, academic, strong research element
Image acquisition and image processing sub-systems designed independently	No; lighting-optics sub-system designed experimentally using an IVS.	Yes; pre-existing image acquisition sub-system cannot be modified after analysing images with IVS
In situ reprogramming to accommodate new applications	Possible. System should be extensible and reusable	Unlikely. Single difficult application solved
Input ''data''	Piece of metal, plastic, glass, wood, etc. to be inspected	Computer files holding digital images
Knowledge of human vision influences system design?	Unlikely but this is not an excuse for seeking inspiration from nature	Possible. Much of CV research is directed specifically to understanding human vision
Motivation	Solving practical problems, to improve safety and/or reduce costs, or for social reasons (e.g., improve environment)	Search for knowledge, or solving complex academic problems
Multi-disciplinary subject?	Yes. Holistic design is essential	No. IP sub-systems designed with scant regard for other modules
Nature of an acceptable solution	Satisfactory performance at affordable price, with acceptable processing speed	Optimal performance of paramount importance
Nature of the subject	Pragmatic. Practical consideration usually take precedence over academic rigour	Academic research. Publications normally contain a lot of ''deep'' mathematical analysis
Operator skill level required	(a) IVS medium/high (b) TVS must be able to cope with low skill level	May be very high
Output data	Simple logical or quantitative signal (e.g., ACCEPT/REJECT, or position coordinates to control robotic equipment)	Complex output signal, may be symbolic qualitative, intended for human being
Practitioner's skills	Multi-disciplinary: Systems Engineering, optics, lighting, computer architecture, software. Practical and experimental skills essential	Computer Science/mathematics, theoretical, software

Table 1.2 (Continued)

Feature	Machine vision	Computer vision
Principal criteria by which a system is judged	(a) Ease of use, (b) Cost-effectiveness (c) Consistent and reliable operation	Performance
Role of image processing	IP is just one of many technologies but is not more important than others	Image processing is dominant
Speed of processing	(a) IVS must be fast enough for effective interaction (b) Speed is critical factor for TVS.	Not normally of prime importance.
System operates free standing?	(a) IVS interacts with human operator (b) TVS must be able to operate free-standing	May rely on human interaction or operate autonomously, depending on the application
User interface	Critical feature for both IVS and TVS but for different reasons	May be able to tolerate weak interface as system is probably used by experts in application area

mechanisms, etc.) and to a wide range of factory-management tasks (e.g., monitoring material stocks, checking the floor for debris, controlling cleaning, ensuring safe working conditions for human operators, etc.)

Machine Vision is also applied to a variety of other areas that have considerable significant commercial importance and academic interest:

- Aerial/satellite image analysis
- Agriculture
- Document processing
- Forensic science, including fingerprint recognition
- Health screening
- Medicine
- Military applications (target identification, missile guidance, vehicle location)
- Research, particularly in physics, biology, astronomy, materials engineering, etc.
- Security and surveillance
- Road traffic control/monitoring (the London Congestion Charge is an example)
- Personal identification using biometric data

It would be possible to analyse each of these non-industrial applications areas, in order to establish a broader horizon for Machine Vision.

Systems Engineering is at the very heart of Machine Vision. For this reason, it is unacceptable to limit our concern to the mathematics and algorithms, or any other single aspect of the system design task. What is unique about industrial Machine Vision is the huge variety of potential applications. While there is a large market for machines that can detect tumours in mammograms, read car registration plates, or identify people from iris scans, these are niche markets. On the other hand, a large manufacturing

Table 1.3
Technologies needed to design a machine vision system

Technology	Remarks
Algorithms/heuristics for image processing	One of the principal subjects in this book. Notice that Machine Vision and Computer Vision are likely to place emphasis on different image processing methods.
Analogue/video electronics	Analogue pre-processing can be a cost-effective way to reduce the data rate and/or improve signal-to-noise ratio. Nowadays, many proprietary cameras generate digital outputs, so modifying the analogue preprocessing is impossible.
Communications/ Networking	TVS may be networked to stock-control, work-control, and management computers. May also be interfaced to real-time factory control systems, PLCs, etc.
Digital electronics	Digital pre-processing can provide a cost-effective way reduce the data rate and improve signal-to-noise ratio.
Industrial engineering	TVS must be robust and provide reliable operation in a hostile factory environment
Lighting	Lighting is a critical component for any Machine Vision system. (N.B. Shrouding is needed to eliminate ambient light and is just as important as the applied lighting.)
Mechanical Engineering	• Good mechanical handling is essential for presenting objects to the camera in a controlled manner (N.B. Defective parts must not jam the handling unit) • Vibration-free mountings are needed for widget, lights, optics and camera
Optics	Numerous special effects can be provided by lighting and optics and can often convert a very difficult problem into a trivial one.
Quality Control and Production Engineering	TVS should comply with current operational and environmental working practices, QC protocols, etc.
Sensor	• The imaging sensor may detect 1-D, 2-D, or 3-D spatial patterns using one of the following physical sensing media: gamma ray, infra-red, magnetic field, microwave, electron beam, pressure, surface profile, ultra-sonic, ultra-violet, visible, x-ray, etc. • Various image formats are possible provided by: array-scan camera, line-scan camera, flying-spot scanner,
Software	The software should be well engineered as it is likely to be reused for new applications
Systems Engineering	It is essential to maintain a good overall "balance" throughout the TVS; no component should be under stress because other parts have been neglected during the design.
User interface	An ergonomic interface is essential as users may be untrained.

company may introduce many hundreds of very different products every year, each with a life-time of no more than 2–5 years. Designing reliable, versatile and reusable inspection equipment for these presents a challenge that is unique to industrial applications.

1.3.2 Does It Matter What We Call Our Subject?

The answer is most emphatically *YES*! There are several reasons why MV deserves to be identified as a subject in its own right.

- To ensure proper refereeing of academic papers
- To promote appropriate funding of research
- To protect potential customers of vision systems from claims of expertise in MV, by workers who have a narrow field of expertise and are therefore not competent to design a complete system
- To protect the careers of its practitioners

Most technical papers in Machine Vision are not intellectually "deep," as those in CV usually are; papers on MV are usually much broader in their scope. It is usual, for example, to find a rather small amount of detailed mathematical analysis in a paper on MV. If mathematics is present at all, then it is almost certainly directly relevant to the solution of a practical problem. On the other hand, many papers in CV seem to be written with the deliberate intention of including as much mathematics as possible, seemingly to impress the readers with the author's erudition. The author of this chapter has on several occasions received quite critical comments from referees of academic papers who are clearly unaware of the distinctive nature of MV. (It should be made clear that, in several cases, the same article was subsequently published in a forum that is more sympathetic to "systems" issues.) There is a distinct tendency for workers in CV to dismiss papers in MV as being trivial, or shallow, because they do not contain a lot of analysis. On the other hand, a well-written paper on Machine Vision is likely to cover a wide range of practical issues, such as object transportation, lighting, optics, system architecture, algorithms, human factors, etc. Hence, papers that concentrate solely on the mathematics and algorithms of image processing may not be acceptable in a journal or conference on Machine Vision.

In similar way, the assessment of applications for grants or promotion can be biased, if the referee thinks his or her skills are appropriate when in fact they are not. For this reason, the distinction between MV and CV is crucial. A mistaken impression of a person's total contribution to a project can be gained by someone who does not understand the full scope of his work. Just as it would be unsuitable for a person trained in, say, mathematics to judge a hardware engineer, so it is inappropriate for a CV *specialist* to make categorical judgments about an MV *generalist*. (The significant words are set in italics.)

There is good reason to believe that research grant applications are sometimes rejected because the referee fails to understand the full significance of a new approach. Referees who look for significant new techniques in algorithms will not normally be impressed by grant applications that concentrate on "systems issues," such as the inter-relationships between the choice of mechanical handling device, optics, camera, and computing system architecture. As a result, perfectly sound MV grant applications are sometimes rejected. Of equal significance is the problem of inferior grant applications being accepted because the referee is unaware of broad "systems" issues, or new developments in MV thinking that make the proposed approach less than optimal. For example, a new kind of camera may eliminate the need for a certain type of image processing operation. Or, a new type of lamp, or optical filter, may enhance image contrast to the point where only the very simplest type of processing is necessary.

Perhaps the most important single reason for needing to protect the identity of MV is to avoid the damage that can be done to its reputation by charlatans. Indeed, a great deal of harm has already been done by specialists in CV, DIP, PR and to a lesser extent AI *dabbling* in Machine Vision when they are not competent to do so. Ignoring image acquisition issues is a sure way to generate an over-complicated image processing solution to a problem. Often, the requirements of an application could be satisfied better by a simple adjustment to the lighting or optics, rather than adding sophistication to the image processing. The classic problem of this type is that of thresholding an image. There are *many* ways to achieve high-contrast images, some of which do not require the use of conventional back-illumination (➔ Chaps. 8 and ➔ 40). Ignoring such a possibility and then resorting to an over-complicated dynamic thresholding technique is simply poor engineering practice. There is no doubt that many installed vision systems use excessively complicated algorithms to compensate for poor lighting. From time to time, over-zealous and ill-informed salesmen have claimed that their vision product is able to operate in uncontrolled ambient light. This is simply not true. Given the present state of our knowledge of the subject, claims of this kind are fraudulent! Many other vision systems are less reliable than they could be, had they been designed from the outset with a good appreciation of all of the systems issues involved.

To summarise, there is no excuse for fooling a customer who has no detailed knowledge of our subject, into thinking that a specialist in CV is competent to design a system for use in a factory. Unfortunately, intellectual honesty is not a universal virtue and the consequences of misleading customers like this can have a major impact on the wider, long-term public perception of a maturing technology such as MV. There are subtle ways in which erroneous views can be propagated and perpetuated. Every MV engineer knows that very subtle points can sometimes lead to the undoing of an otherwise successful application. For example, static electricity may make it very difficult to keep optical surfaces clean. This can reduce image contrast sufficiently to render a certain algorithm unreliable. Another example is the potential health hazard that can arise when inspecting objects transported on a fast, continuously moving conveyor. In order to "freeze" the object motion for viewing with an array-scan camera, stroboscopic illumination could be used. However, flashing lights can induce migraine attacks, epileptic fits and cause rotating shafts to appear stationary. On the other hand, using a line-scan camera, or an array-scan camera with a fast electronic shutter, avoids this altogether.

1.4 Four Case Studies

Lest the reader is still unconvinced, we will now consider four practical applications which highlight the essential identifying feature of Machine Vision: the need for a Systems Engineering approach to design. We have attempted to encapsulate the experience that is needed to design a successful Machine Vision system in a series of proverbs (➔ Appendix C). Five of these relate specifically to the need to define the intended function of a vision system.

- The specification of the vision system is not what the customer wants.
- No machine vision system can solve the problem that the customer forgot to mention when placing the order.
- The specification of a vision system should indicate its functionality and performance.
- Beware when the customer says "By the way! . . . We would like to be able to inspect some other objects as well."
- The customer said his widgets were made of brass. He did not think to state that they are always painted blue and are oily.

Another Proverb has gained particular resonance with the author's colleagues working in MV and summarises a danger that is peculiar to the design of vision systems:

- Everybody (including the customer) thinks he is an expert on vision and will tell the vision engineer how to design the machine.

That is totally wrong! Such an opinion has to be firmly corrected and then reinforced over and over again.

In order to emphasise that MV is an aspect of Systems Engineering, we will trace the history of four very different industrial projects in which vision systems were commissioned:

1. The project specification was drawn up without reference to what is technically possible, or economically feasible. No experienced MV engineer was involved.
2. The vision system was designed successfully by an experienced team working full-time in MV. An unusual image-acquisition sub-system was designed in a series of experiments using very simple equipment. This greatly simplified the image processing required.
3. The client specified the problem in a totally inappropriate way. Good engineering practice detected the problem before it was too late.
4. The specification of an AVI system evolved during the preliminary contract negotiations and the final machine was very different from that first envisaged. This dialogue would never have taken place without an experienced MV engineer being involved.

In each case, the engineers working in the client company had no previous experience with Machine Vision technology. They differed in their willingness to learn and accept the advice given by an experienced vision engineer. As a result, these four projects reached very different conclusions.

1.4.1 Doomed to Failure

The first project epitomises an approach that almost inevitably leads to failure and disillusionment with MV technology.

A certain company makes complex, high-precision components for domestic goods, by a process involving several stages of pressing and punching sheet steel. Some of the products display a variety of faults, including both functional and cosmetic defects. The company, in association with a well-respected university department, set up a joint project. The investigator was a young graduate in Production Engineering, with no prior experience of Machine Vision. In addition, his supervisors had very limited experience in this technology. Inexplicably, the university-company partnership succeeded in gaining funding for this project from both government and board-level company sources. The university also had an established link with a major electronics company that lead to a decision, quite early in the course of the project, to purchase a certain (old-fashioned) turn-key image processing system. This decision was made for political rather than sound technical reasons. The specification of this system was far inferior to that available from a modern general-purpose computer costing about 25% of the price. At the stage when this decision was made, the lighting-optics-viewing arrangement had not been considered seriously and no image processing algorithm had been designed or selected. There was no evidence at all that this turn-key system would be able to implement the necessary image processing functions, since the requirements at this stage were simply not known. It is interesting to note that, during the middle phase of the project, two highly experienced MV engineers were given privileged access to the work of this group and counselled strongly but unsuccessfully against the approach being adopted. (Recall the Proverb which states that everyone thinks that he or she is an expert in vision.) The system specification was drawn up without any reference whatsoever to what is possible in technical, financial or human terms. The outcome in such a situation is inevitably failure and disillusionment with MV technology. The project failed!

1.4.2 A Successful Design

In the mid-1980s, engineers in a certain company were in the late stages of building a fully automatic flexible-manufacturing system, designed to produce over 50 different products, when they realised that they would be unable to complete the task without a non-contact parts-location system. The requirement was well-specified and was highly suitable for a vision system. The client company contacted a company specialising in designing and building Machine Vision systems. The latter company could not, at first, see a path leading towards a successful solution and, in turn, consulted a very experienced MV researcher. He devised an experiment involving a plastic meal-tray, a dress-maker's tape measure, a desk lamp and a standard surveillance-type video camera. (He needed to simulate a motorised turn-table, a line-scan camera and oblique lighting.) Acquiring each image took over an hour, with three people participating. Nevertheless, a series of test images was obtained and a suitable image processing algorithm devised, using an interactive Machine Vision system of the type described in later pages of this book. On the basis of these seemingly clumsy experiments, the vision company was able to design, build and install a well-engineered target system. It worked very well, except that sun-light caused intermittent optical interference. However, a solution was soon found. This consisted of a metal screen placed around the camera and object to be inspected, to block out ambient light. The final vision system was highly successful and worked reliably for several years. The important point to note is that consulting a suitably experienced Machine Vision engineer early in the project lead to a highly successful solution. Although the resources used in the early experimental stages of this project were severely limited, they were appropriate for the purpose and a good system design ensued.

1.4.3 Mouldy Nuts

A highly experienced and well-respected MV engineer was commissioned to design a machine to inspect nuts (the edible variety), to detect patches of mould on the surface. He was supplied with a large bag of nuts, to enable him to design the inspection machine. He expended a lot of effort on this exercise. When he had completed the task to his satisfaction, he contacted the client, to arrange a demonstration. When the engineers from the client company arrived for the demonstration, they brought with them another large bag of nuts, hoping to test the vision system on previously unseen samples. (This idea is, of course, very sensible, in principle.) The vision system did not respond well in these tests and failed to satisfy the client's expectations, for one very simple reason: the new sample of nuts had fresh mould which was *green*. The samples originally supplied to the vision engineer to help him in the design process had older *brown* mould. The client had omitted to tell him that the green mould quickly changes colour to brown. This fact was obvious to them but they had not thought it important to tell the vision engineer. Since brown mould patches can be identified by eye, it was "obvious" to the client that the green mould could be detected just as easily by a Machine Vision system. It is a universal truth that, whenever the client tries to tell the vision engineer how to do his job, trouble is about to occur.

The lesson for us is that although the system was not designed appropriately in the first instance. The good engineering practice of testing the system on data independently of that used during the design highlighted a serious problem. The result was annoying but not disastrous, as it was fairly easy to redesign the system to recognise the coloured mould. In fact, the revised design was considerably simpler than the first one!

1.4.4 A System That Grew and Grew

In this section we describe how one application, which was initially conceived as requiring a single-camera vision system, grew to much more massive proportions. As we will see, the original concept grew very considerably, during the dialogue between the customer and the vision engineer. Of course, this is the best time of all for the specification to change! In this project, the vision engineer, was encouraged by the client to explain how Machine Vision could form part of a much larger system. In this case, the system that was eventually built was a complete parts-handling machine, which just happened to incorporate Machine Vision, among several other functions. In the machine that was finally delivered to the customer, the vision sub-system was fully integrated into the larger machine and was no more, nor less, important to the overall function than that of the other equipment. In other words, the vision engine formed just one of many components of the larger system. The specification of the system was developed during several conversations, involving the MV engineer and his client. Here is a brief history of the project:

1. The MV systems engineer was invited to provide advice about the provision of an Automated Visual Inspection system for components used in the automotive industry. At first sight, the task seemed straightforward: the customer wanted to ensure that certain safety-critical components met their specification, but wanted to avoid the long and tedious tests then being performed manually. A significant speed increase (greater than 100%) was also expected. The initial specification was for a machine that could locate four fixing holes at the corners of the component, which consists of a flat, nearly-rectangular, metal plate, with a central cylindrical spindle, normal to the plate (➋ *Fig. 1.6*).

   In the first instance, measuring the positions of the centres of these holes appeared to be a straightforward task. However, the project became ever more complex as successive meetings took place between the MV engineer and the client.
2. The *Production Engineer* had ambitious ideas! He reasoned that, if the inspection system could perform this task, then why not others, as well? So, he added the requirement to check that the holes are tapped. Clearly, the camera used to locate the holes could not be used to check these female threads, as the latter task requires an oblique view and a different lighting arrangement.
3. The third requirement, also suggested by the Production Engineer, was for another camera, with its associated image processing sub-system, to check and measure the male thread at the end of the spindle.
4. Then, a requirement was drawn up for yet another camera, to check that the heat treatment that this part receives during its manufacture has been completed. (Heat treatment causes perceptible colour changes.) Up to now, the object to be examined could be inspected while it was static.

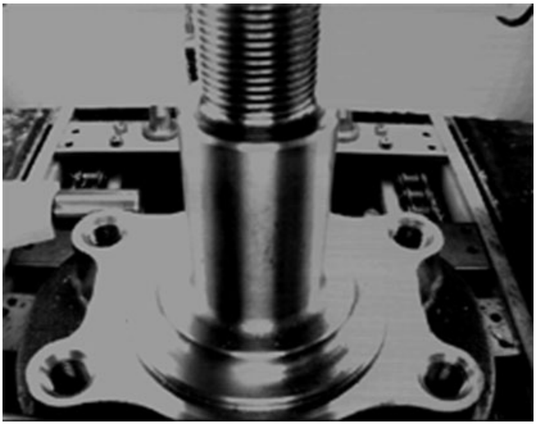


Fig. 1.6
Subject of a simple task that grew and grew as more inspection functions were added

5. Since the spindle diameter is critically important, yet another requirement was introduced. However, this one was satisfied without the use of vision. Four swinging arms, each carrying a digital micrometer, were used to measure the spindle diameter. The result is more accurate than a simple vision system could achieve. Requirements 1 to 5 were all suggested by the Production Engineer.
6. Next, came the requirements defined by the *End-of-line Packer*: The parts are sprayed with an anti-corroding agent, prior to shipping out of the factory. Before they are sprayed, the parts must be cleaned. This involves an all-over spray, with particular attention being paid to the four fixing holes. (These might contain swarf and other particulate debris.) The cleaning process was to take place in an on-line washer, with the parts being transported on a continuously-moving conveyor belt. The cleaning process also imposed a need for a component drying station.
7. The *Quality Control Engineer* was the next person to participate in specifying the system requirements, by suggesting an additional function for the new machine. He pointed out that there is a need for a marking system, enabling defective components, the machine and shift responsible to be identified later. Since the parts are painted shortly after they have been made, the marking must be engraved. In order to accommodate this function, the conveyor belt had to be lengthened. This had an unfortunate effect on product safety: objects that have already been rejected as faulty could mistakenly be returned to the *ACCEPT* region of an unguarded conveyor. To prevent this, faulty parts should pass into a secure cabinet that is accessible only to a trusted member of staff.
8. The *Line Supervisor* provided the last suggestion for embellishments to the planned machine. With only a little additional complication, the machine can be made totally automatic, with no operator intervention whatsoever. This required the provision of loading and ejection bays.

So far, we not even begun to discuss the attitude of the accountants to all this! That is another story that need not concern us here. Let it suffice to say that the accountant did finally agree that the (enlarged) machine should be built. To a novice, this story may seem exaggerated. However, it is a real-life experience and, in general terms should be regarded as typical, rather than exceptional. Notice how much of the specification of this machine is concerned with nonimaging tasks. Of considerable importance in the negotiations is the improvement in confidence that results from the vision engineer being able to demonstrate each step in the developing inspection strategy. (We will return to this particular point many times later in this handbook.)

1.5 Machine Vision Is Engineering, not Science

The experience outlined in these four case studies is not unique. They all emphasise the importance of proper initial planning. One crucial part of this is reaching a clearly defined agreement between the customer and the vision engineer about the real objectives of the exercise. In ➜ Appendix B, we include the Machine Vision System Design Questionnaire. This was specifically designed to assist during the negotiations that will take place in the initial phases of a Machine Vision project. It does so by high-lighting a broad range of issues that might otherwise have been overlooked in the discussion between the vision engineer and his client. The use of a pre-defined structure for the system and contract specifications should not be taken as an excuse to stifle dialogue between the vision engineer and his customer. The specification for the system described in ➜ Sect. 2.4.4 evolved during several weeks of detailed discussion between these parties working closely together. Dialogue is an essential feature of a successful design, since neither the vision engineer nor the client has access to all of the information necessary to complete the task without the involvement of the other.

Another point to note is that an experienced vision engineer may not always give the answer that the customer would like to hear but it may save a lot of problems later. One of the most valuable pieces of advice that a vision engineer can give is "*Don't use vision.*" Blunt words are sometimes necessary, to counter the prejudiced opinions of technically naive people who have an over-optimistic view of what Machine Vision can accomplish. Remember the maxim

- ▶ Wanting something does not mean that it is both technically possible and economically realistic

Excessive optimism arises whenever people try to translate what they *think* they see into terms relating to a machine. When they first perceive the possibility of using MV technology, would-be customers often expect miracles, because they think of themselves as being experts in vision. As a good working rule, it may be assumed that this is *never* the case, since introspection is a very unreliable method of determining how we perceive the world around us. Would-be customers are almost always totally unaware of both the limitations and potential of modern vision systems. Even if they have read about the subject, they are unaware of many of the subtle points of detail that can destroy a vision system's chances of ever working effectively and reliably. Very often during the course of the initial discussions between the customer and vision engineer, various requirements, not directly related to image processing, become apparent and from thereon form an integral part of the specification of a system with a far broader function than was at first imagined.

1.5.1 Systems Engineering

➜ Appendix D summarises the inter-relationship existing among those areas of study that a vision engineer must expect to encounter some time in his work. He should have, at least, a broad understanding of each one. ➜ *Figure 1.7* shows the general structure of two vision systems intended for use on the factory floor. ➜ *Figure 1.7a* relates to area-scan sensors and ➜ *Fig. 1.7b* to line-scan sensors. These are perhaps the most important diagrams in the whole

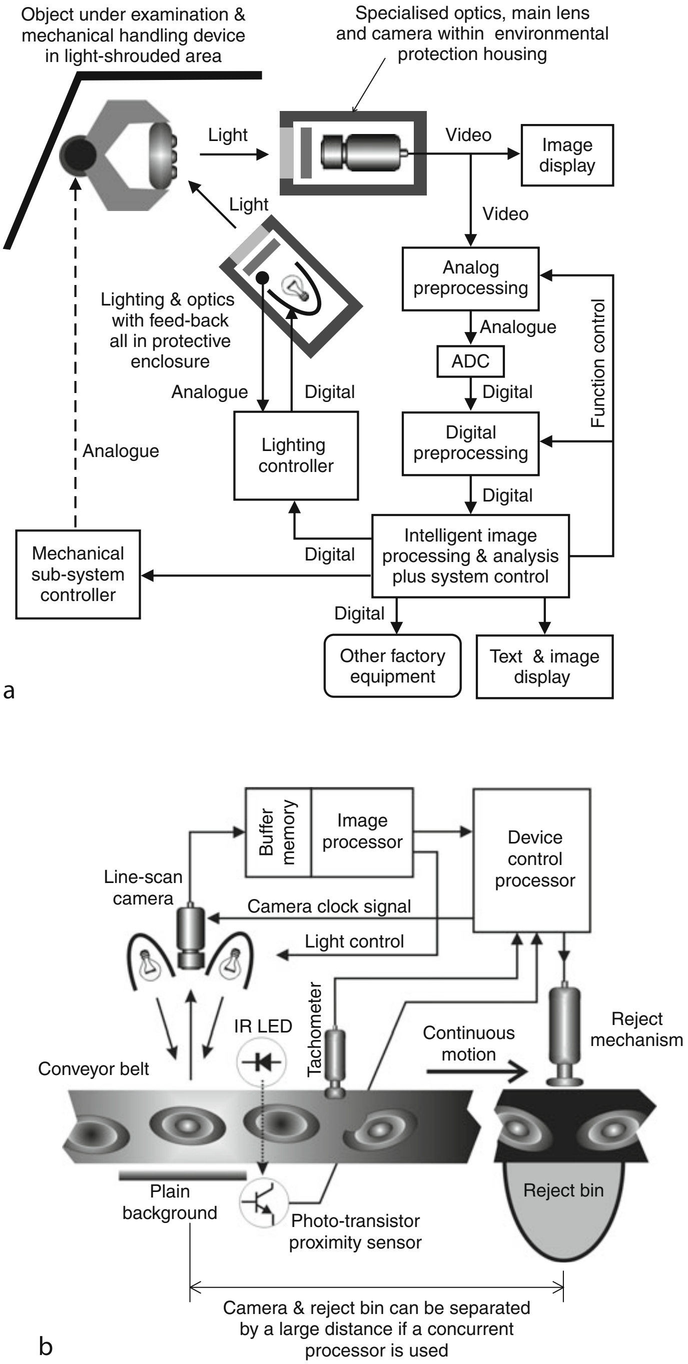


Fig. 1.7

Archetypal Machine Vision system (a) General scheme, applicable to a wide range of industrial applications and using an area-array sensor. (b) Typical on-line Automated Visual Inspection system based on a continuously moving conveyor belt

book. Like ➲ Appendix D, they emphasise that Machine Vision system requires a multi-disciplinary approach, combining and harmonising skills and knowledge of several different areas of engineering: mechanical, video, electrical, analogue and digital electronics, control, illumination, optics, software (not visible in either diagram), ergonomics (human-computer interfacing), production/Quality Assurance and, of course, systems integration. A machine intended for the factory floor will be termed a *Target Vision System* (*TVS*). (This name was chosen because a TVS is the end result of the design process.) Some systems may not require all of the modules depicted in ➲ *Fig. 1.7a* and there may be other minor modifications to take account of application specific issues. For example, when low-speed processing is acceptable, a standard computer may be able to perform all of the image processing operations required. When high speed is essential, dedicated image processing hardware may be necessary (➲ Chap. 25). In many instances, a separate intelligent processing module is not needed. ➲ *Figure 1.7* encompasses systems that employ visible light (VIS), microwave, infra-red, ultra-violet, x-ray and gamma-ray sensing and requires only minimal changes to cover neutron-beam, electron-beam and ultra-sonic imaging systems as well. The same diagram also covers systems that exploit fluorescence and phosphorescence.

No diagram can properly convey the critical point that it is the *interplay* between the different parts of a vision system that determine whether it is successful or not. Designing a system requires a mixture of experience, knowledge, intelligent analysis and experimentation. (➲ Appendices C and ➲ D) How the design process is performed is discussed in detail in these pages. There are two crucial parts to the design process: choosing the image acquisition sub-system and analysing the image data that it generates. All of this must be done with the sole objective of producing the design for a robust, reliable machine that serves the agreed purpose. In ➲ Chaps. 8 and ➲ 40, we list and illustrate about 130 different lighting-viewing configurations. These can be used singly, while two or three may be combined for some applications. There is no algorithm for choosing the most appropriate one, or combination. Neither is there any prescription for analysing the images so generated. Moreover, the choice of the image acquisition hardware and the image analysis software/hardware are intimately bound together. In these pages, we describe how optical components, cameras and image processing procedures work. We also explain how experimentation helps in the design process. We cannot fully encapsulate the experience, nous and low cunning of an experienced vision engineer in a computer program. The best we can do is make good use of human intuition by building appropriate tools to assist in the design process.

1.6 Design Tools

Despite the rapid progress that has taken place in Machine Vision technology in recent years, there remains a significant and pernicious problem: the very large amount of highly skilled engineering effort needed to develop systems for each new application. It has been claimed many times, by numerous salesmen, authors and conference speakers, that Machine Vision is a very flexible technology, ideal for sensing defects in a very wide range of industrial products and manufacturing processes. The fact that Machine Vision is potentially able to fulfill the needs of a very diverse range of industrial inspection, monitoring and control tasks has already been established in numerous laboratory-based feasibility studies. However, this oft-claimed versatility is merely an illusion, if there are not enough people with the necessary systems engineering skills to design and build machines that will continue to operate effectively and reliably on the factory floor. Bearing these ideas in mind, it becomes clear that a variety of computer-based design tools for Machine Vision would be valuable. If properly implemented, such tools would reduce the skill level for many of the sub-tasks in the design process. In this way, we hope to improve the overall effectiveness of the available pool of skilled labour.

The major tasks for which help is most needed may be included under the following headings:

- Overall planning of a vision system, given a range of general application constraints
- Configuring the optical sub-system (i.e., placing the lighting, optical elements and camera)
- Calculating field-of-view, depth of focus, camera/image resolution, etc.
- Choosing an appropriate image sensor (camera)
- Designing the optical sub-system (i.e., choosing the lens and mirror configuration, coatings, mountings and performing the detailed design of multi-element systems, etc.)
- Selecting and designing an appropriate image processing algorithm.
- Designing and programming a target system, for use on the factory floor.

1.6.1 Image Acquisition

➋ Chapter 40 contains a catalogue of over 150 different lighting/viewing methods (read ➋ Chap. 8 first). These are presented in the form of descriptive text, optical layout diagrams and in some cases sample images.

Another important development that is intended to assist in the design process is prototyping systems for lighting and viewing. One of these, called Micro-ALIS, consists of an optical bench and a range of high-precision fibre-optical illuminators, with rapid-release clamps to hold them in situ. The other system, called ALIS 600, is a light-proof cabinet containing a range of different light sources, which are operated under computer control. [*ALIS600* and *MicroALIS* (Application Lighting Investigation Systems) were previously products of Dolan-Jenner Industries, Inc, URL http://www.dolan-jenner.com/ They are no longer included in the product list of that company.]

1.6.2 Algorithm Design

The design process for a TVS has been summarised in the form of a flow chart (➋ *Fig. 1.8*). It is not possible to design a vision algorithm as one would, for example, write a program to perform statistical data analysis. The reason is exactly the same as that which prevents a person from becoming an expert on vision simply by introspective analysis of his or her own thought processes. A person writing a vision program needs to see the results of each step in the procedure. With very few exceptions, a person cannot simply look at a picture, mentally analyse it as a computer would and then decide what algorithmic steps are needed to extract a given item of information from it. Interactive image processing was devised in the mid-1970s and has been found to be a valuable tool for designing/choosing a vision algorithm. Its rôle in designing an industrial vision system is illustrated in the following anecdotes, which describe actual projects in which the author has been involved.

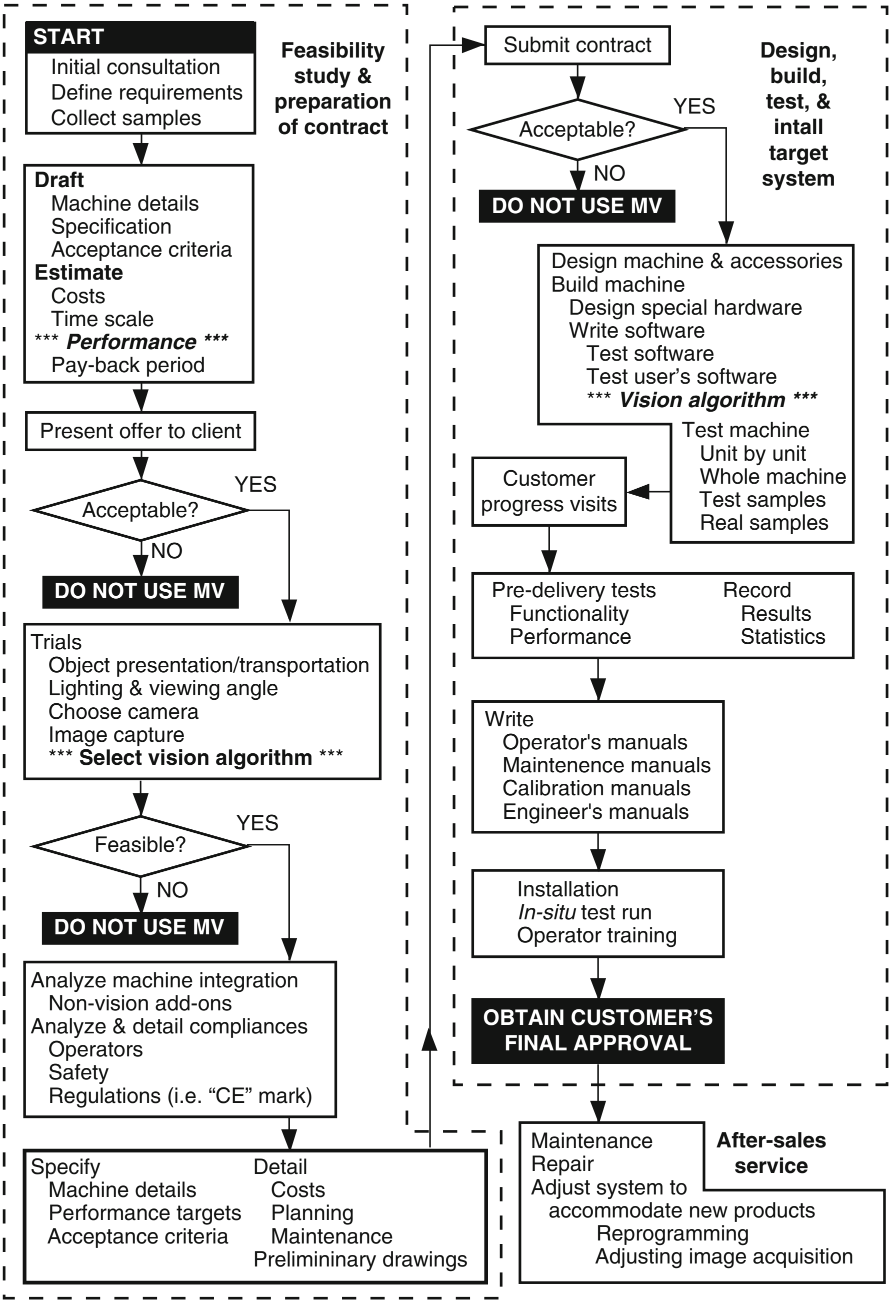
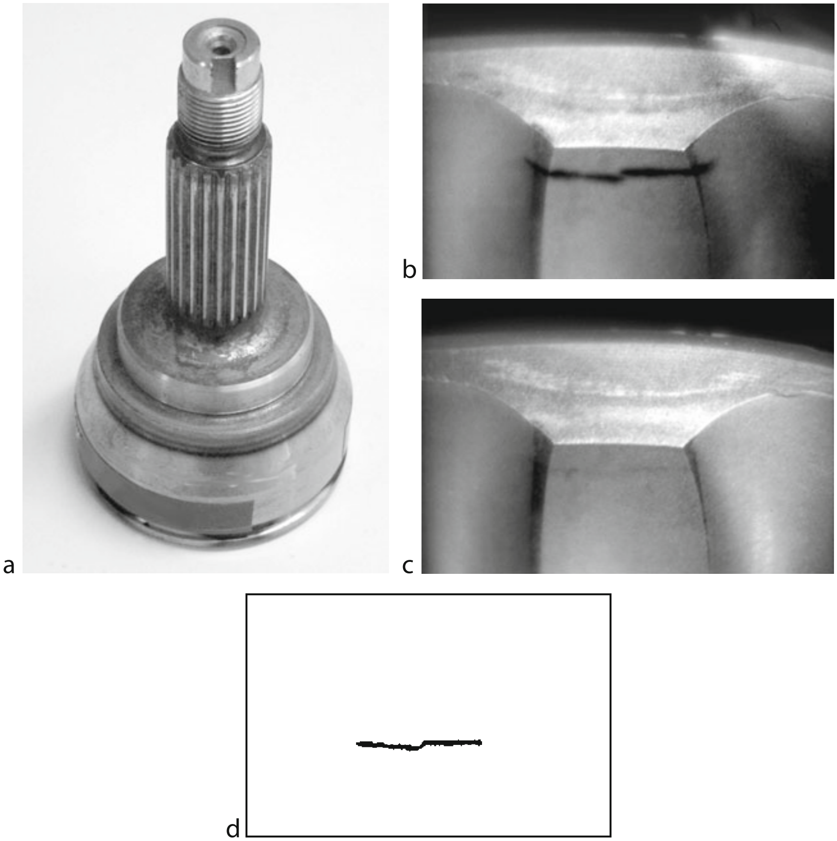


Fig. 1.8

Designing a target vision system for use in a factory

1. A certain company wished to build a system for checking the characteristics of an aerosol spray. Among other parameters, the spray-cone angle was to be measured. The particular spray material that interested the client would have been rather unpleasant to use. Instead, the client was prepared to study the application by using a less noxious material: furniture polish. A simple set-up was arranged, consisting of a light immediately above the spray cone. A black board, located behind the spray, acted as the visual background (➋ *Fig. 1.4d*) shows the negative image. Using this simple arrangement, several images were digitised and then processed, using an interactive vision system (IVS), which was used to explore various processing options. On this occasion, it took the author only a few minutes to derive a suitable image processing algorithm that could estimate the spray-cone angle and derive a measure of its symmetry. On the basis of this quick test, the client was convinced of the merit of using vision for his application. A *TVS* was then commissioned, designed and built.
2. A company wished to detect cracks in forged steel connecting rods for automobiles. In this instance, the client already employed a sophisticated method for detecting cracks, the so-called *Magnetic Particle Indicator* method. This involved magnetising the component, then immersing it in a suspension of very fine ferro-magnetic particles that are also fluorescent, within a kerosene base. The components are then shaken to remove excess "ink" and are dried in hot air. They are then illuminated by ultra-violet light, but with no applied visible light. The cracks fluoresce (in the visible wave-band) because the magnetic-fluorescent particles tend to adhere to the surface in regions where there is a high remnant magnetic field. The cracks highlighted in this way were then viewed using a video camera. The images were analysed using an IVS and an effective algorithm for detecting and measuring cracks was found within a few hours. On this occasion (early-1980s), an entirely new algorithm was developed and has subsequently been employed in many other industrial vision systems. Another technique for highlighting crack visibility is called the *Dye Pentrant Method.* In this case, a non-magnetic "ink" is used that has a high surface tension and low viscosity. ➋ *Figure 1.9* shows how effective the dye penetrant method is. Clearly, to ignore this type of "trick" is sheer folly!
3. A flattened toroidal alloy casting (➋ *Fig. 1.10a*) was presented to the author and his colleagues as a sample of a range of circular objects that were to be handled by a robot vision system. The client asked for a vision system that could measure the orientation of these components to a tolerance of 0.5°. An experiment was devised, simulating a line-scan camera observing the intensity profile along a radius of a turntable. As the component, located nearly concentrically on this turntable, was rotated (by hand), the virtual line-scan camera was sampled repeatedly, enabling a polar-coordinate intensity map to be created (➋ *Fig. 1.10b*). High contrast images were created by using low-angle illumination. The images were analysed by an IVS and an algorithm was devised in a few minutes. On this basis of this experiment, the client ordered a TVS to be built. The interesting feature of this exercise was the crude experimental rig, which had to be set up in a hurry, at minimal cost. The turn-table consisted of a plastic meal tray, with a dress-maker's tape measure stuck around the rim. This enabled the make-shift turntable to be rotated manually, by a fixed amount, after every line-scan. The virtual line-scan camera was, in fact a standard area-scan camera but only one scan line was kept. Since more sophisticated equipment was not immediately available during the afternoon when the hoped-for clients were available, a great deal of time and effort was saved by this exercise.



■ **Fig. 1.9**

Cracks in a forged-machined ferrous component. ("Tulip" for an automobile constant velocity joint) (a) "Tulip." The cracks are located just below the rim of the cup and are not visible from this view. (b) With a narrow band filter in the optical path. Some ambient light was deliberately tolerated here, to produce an image that is able to show off the power of the image processing. The ambient light reduces the image contrast. It is also possible to obtain a very high contrast using fluorescence: UV illumination and viewing in the visible wave-band. (c) Normal view, i.e., with no filter present. (d) Crack detected, using the QT command sequence *[crk(5), thr(145), kgr (50), dil(4), ero(4), rbe]* applied to (b) (see ➲ Chap. 21 for details)

1.7 Overview of This Book

Throughout this book, algorithms are frequently expressed using command sequences for the QT image processing software introduced in ➲ Chap. 21 and described in more detail in ➲ Chap. 41. This is done so that the reader can repeat the experiments. This will allow him to become much more familiar with image processing concepts. The reader is urged to experiment: adjust operating parameters, process new images, combine different functions. The best way to learn about QT is to "play" with it! QT's history and rationale are described in previous books by the same author [1–6].

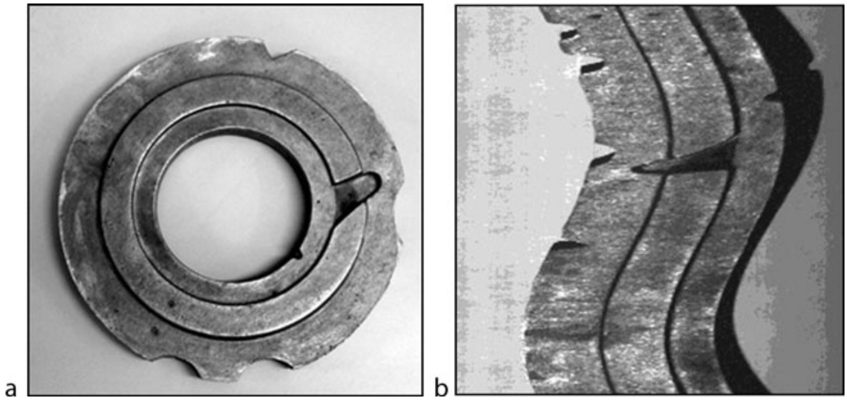


Fig. 1.10
Even simple (crude!) experiments can sometimes yield valuable insight. (a) Alloy toroidal component. (The reverse side is shown in ➲ *Fig. 1.4*, row 1. column 4.) (b) Simulated line-scan image

1.7.1 General Principles

1.7.1.1 Chapter 2: Inspecting Natural and Other Variable Objects

Many chapters of this book concentrate on inspecting, or manipulating, mass-produced engineering products, such as parts for automobiles, washing machines, computers, etc. Their components are typically manufactured to close tolerance. There is another large and as yet largely undeveloped area of potential vision applications, including the inspection, grading and sorting of vegetables and fruit, finding foreign bodies in food materials and monitoring the manufacture of food products. This group is typified by applications in which tolerances are normally large, compared to engineering components. Nuts, apples, carrots and potatoes are not made in a mould. Peas, even in the same pod, are very variable, in size, shape and colour. Pizzas, pies, loaves and cakes are all manufactured on an industrial scale but they may well show significant variations in size, shape and general appearance, compared to the equivalent parameters for plastic mouldings, metal stampings, bottles, nuts, bolts, springs, etc. The high variability of natural products and manufactured food products introduces challenges for the designer of an inspection system that is unknown when tolerances are tighter. Furthermore, it is often difficult to standardise the presentation of a natural object to the camera for inspection. If the shape is variable, it may be difficult to find reference points with which to establish a safe grasping position. Grading fruit and vegetables is a task that is often defined in terms of rules originally written for human inspectors. These must then be translated into a computer program. This suggests the need for a different type of decision-making from that required for close-tolerance engineering components. Prolog-style reasoning is often appropriate in this type of situation (➲ Chap. 23) [1, 3, 5]. This is significantly different from, for example, the image filtering techniques introduced in ➲ Chap. 14 and implemented in software packages such as NeatVision (➲ Chap. 22), or QT (➲ Chaps. 21 and ➲ 41). Parts of some engineering components/assemblies show significant variability. Examples include welding, rivets, wiring, flexible tubing, solder joints, etc. During the manufacturing process high levels of variability may exist temporarily, for example when extruding dollops of semi-fluid material (adhesive, molten glass, clay), or when removing a casting from its mould. To monitor a material-removal process, we might usefully monitor the debris produced (swarf, dust, sprue) as well as the product itself. Waste material can be a useful indicator of the state of the tooling used in the production process. These are also situations in which we might apply the lessons of this chapter. In many applications where there is high variability, it is convenient to use fast hardware to obtain a standard set of measurements that are passed, in parallel, to a Pattern Recognition module (➲ Chaps. 19 and ➲ 25). The latter learns from its own experience to perform the required discrimination task, which cannot be defined explicitly by a human being (➲ Chap. 24 and the supplement to ➲ Chap. 16). Self-adaptive learning using feedback guided by a human teacher is a more effective alternative to rule-based methods for tasks such as colour recognition (➲ Chap. 16) and texture analysis (➲ Chaps. 30 and ➲ 38).

1.7.1.2 Chapter 3: Human and Animal Vision

Since this is a book about artificial vision, it may seem strange that we include an essay on vision in the natural world. The reason is simple: we need to convince the reader that there is no single model for vision that is ideal for all circumstances. Our own visual system, sophisticated though it is, would not suit a housefly, eagle, owl, chameleon, or even a horse. Each of these animals has managed to thrive within a specific type of environment, for far longer than human beings have existed. Even though it is very limited in its visual capability compared to human beings, a frog has a vision system that has served it well for many millions of years. In many instances, an industrial Machine Vision system is like a frog: both can do certain well-defined tasks, without attempting to achieve the subtlety of discrimination and the high level of intelligent scene analysis that a human being can accomplish. In nature, there are four scenes that nearly every organism has to recognise: food, threat, mate and shelter. This is not so very different from an inspection system that is merely intended to distinguish between "good" and "faulty" products. Animals do, however, have to cope with highly variable environments and cannot change the viewing conditions as a vision engineer can do to make his/her creation cheaper, faster, or more reliable. So, our task is usually simpler than designing a machine as complex as a frog. We must never make the mistake of thinking that only we see the world as it really is. We are only sensitive to a narrow range of optical wavelengths. We cannot sense polarisation, whereas bees and appropriately designed machines can. We cannot sense very low light levels like some animals. Why should we want to replicate a system like ours that is confused by numerous visual illusions? The natural world has produced many different models for vision. For example, there are over 40 different types of eye in the animal kingdom. The compound eye of insects and our own are obviously very different. Even the eye of a relatively close relation, the cat, is significantly different from our own, having a vertical pupil in bright light. Some animal eyes move the lens to focus on objects that are moving along the line of sight, rather than alter the shape of the lens, as our own eyes do. There is nothing particularly special about our colour vision either. Mantis shrimps for example have 12 different colour pigments in their eyes. Spiders are not content with just two eyes as we are; they have eight! This variety does more than present us with a collection of interesting curiosities; it provides inspiration! Vision engineers can benefit from learning about non-human vision systems. Perhaps the most important point of all is that vision engineers should have the confidence to be different; we should not be constrained always to try and emulate human vision. Nature isn't! Although it is immensely powerful, human vision is sometimes ineffective, and in other cases grossly over-sophisticated for the humble tasks required of industrial vision systems. It is always vain to do with more what can be done with less. So, if frog-vision works, use it!

1.7.1.3 Chapter 4: Colour Vision

Colour vision has been studied extensively by psychologists, physiologists and physicists. Philosophers have also pondered over the nature of colour. Photographers, printers and video engineers try to reproduce colours encountered in nature. The result is a large but still incomplete body of knowledge termed Colour Science. When the editor was initially considering the content of this Handbook, he felt that a chapter on the science and practice of colour reproduction was essential. However, on reflection, he modified this view, since Colour Science is concerned primarily with understanding colour vision in humans for the specific purpose of *reproducing* colours. That is not our interest here! Our preoccupation in Machine Vision is *verifying* that coloured features are of the correct size, position and shape. To do this, we need to be able to segment an image according to the colours that it contains. For example, we might need to distinguish "*red*" printing from "*blue*" printing; identify "*yellow*" tablets lying on a "green" surface (to count them); or measure the area covered by red raspberry jam on a cake. Performing tasks such as these does not require detailed knowledge of many of the subtleties that have been discovered about human colour vision. We are faced with stark questions, such as "*Is the feature XXX red, or is it green?*" But, what is "*green*"*?* My wife and I often disagree, in the nicest possible way, about the names of colours, particularly in the blue-green part of the spectrum. Who is correct? My concept of "green" is different from hers – and yours! So, we can all be correct, because human understanding of the term "green" is different from person to person. Having deliberately ignored much of Colour Science, we proceed by discussing some of the different colour models that have been developed. However, even this is not straightforward, because Machine Vision must use the available technology. Colour cameras operate by sensing the optical energy in three overlapping wavebands and thereby generate three separate signals, RGB. There is little point, therefore, in over-reliance on other colour models, since they have to be calculated in terms of the RGB signals coming from a video camera. One other model does stand out as worthy of detailed attention: HSI (hue, saturation intensity). This chapter lays the foundation for ❯ Chap. 16 where the RGB and HSI models are used extensively and should be read in conjunction with ❯ Chap. 3.

1.7.1.4 Chapter 5: Light and Optics

Photographers and video engineers have long been aware of the critical importance of optics. Since forming and conditioning images is an essential feature of all camera-based Machine Vision systems, the ability to produce a sharply focussed image, without significant aberrations, or vignetting, is critical. Poor image quality degrades overall system performance. While it is possible to use software to compensate for mis-focussing, it is far better to design the optics so that this is not necessary. Deconvolution, the computational process used to improve focus, is relatively slow and fails to produce results that are of comparable quality to those obtained by selecting a good lens and using it in the most appropriate manner. While programmable colour-selective filters are very versatile, they cannot compete for speed, or cost-effectiveness, with a simple piece of coloured glass, or for more sophisticated colour separation, an optical interference filter. Elsewhere in this book, a Machine Vision system is pictured as an electronic image processing engine, interfaced to an image acquisition system (i.e., the optical "front end"). Let us instead think of an MV system as an integrated hybrid processor, incorporating the following "computing" elements:

- Optical signal generator (lamps, Chap. 7)
- Optical signal conditioning (illumination optics, Chaps. 6 and 7)
- Data input (mechanical handling system, moving the widget to/from the inspection area)
- Optical processing (imaging optics, Chap. 5)
- Translating the information domain from light to electrical form (cameras and scanners, Chaps. 9–12)
- Signal conditioning (Chaps. 10, 11 and 13)
- Conversion from analogue electrical to digital image format (Image formats Chap. 11)
- Digital image processing (Chaps. 14–19)
- Software for image processing and interpretation (Chaps. 21, 22 and 41)

The data processing rate at the front end of this hybrid processing chain is very large, compared to that in the later stages. An optical processor offers a limited repertoire of available functions, compared to software but is able to handle truly huge volumes of data. Considerable cost/speed benefits can therefore ensue if we can perform most of the processing in the optical domain. Among the limited set of processing steps that can conveniently be performed optically, we can include:

- Image size adjustment
- Image aspect-ratio adjustment
- Trapezium and parallelogram image warping
- Compressing an image into a single line
- Rotating an image
- Inverting one, or both, image axes
- Adjusting the balance of signal levels spatially and temporally (LCD modulators, LED illuminators, Chap. 7)
- Fourier transform (performed on transparent objects/media)
- Generating variable waveband lighting patterns (LED illuminators, Chaps. 7, 8 and 40)
- Filtering images on the basis of wavelength (optical filters, also see Chaps. 5 and 40)
- Filtering images on the basis of polarisation.

The catalogue of lighting viewing methods in Chap. 40 is built on an understanding of the optical components described in this chapter. The lesson here is simple: good optical design will greatly improve image quality and can achieve some valuable special effects that can lead to greatly improved system performance and cost/effectiveness.

1.7.1.5 Chapter 6: Telecentric, Fresnel and Micro Lenses

In our everyday experience, looking into a deep well produces two concentric rings: the outer one corresponds to the top of the well and the inner one to the well bottom. An ideal telecentric lens superimposes these contours. This can only be approximated using a conventional lens when the well is viewed from far away. Telecentric lenses do not conform to our normal expectations. They may not be familiar to some readers, because they do not have the same importance for general photographic, or video, applications as they do for Machine Vision. Within the restricted world in which industrial vision systems normally operate, telecentric lenses offer important advantages compared to conventional lenses: size constancy as the object-lens distance varies, freedom from parallax, high measurement accuracy and the ability to detect subtle local defects in transparent sheet materials and on plain mirror-like surfaces. Telecentric lenses have been central to the success of many installations. Gauging has benefitted particularly from their use, since they provide a much better estimate of the real position of the edges of an object than a conventional lens can. For the greatest precision, a collimated light beam is used to back-illuminate shiny objects. A collimator is a telecentric device (➲ Chaps. 5 and ➲ 7). Although not uniquely associated with Machine Vision, telecentric lenses have taken a pivotal role in our subject. Without their use, the algorithmic techniques for measuring object size and position from an image would be more complicated. The cone of visibility for a telecentric lens is a cylinder, while that for a fish-eye lens is a diverging cone. (Its apex is at the camera.) Hypercentric lenses have a converging cone of visibility; the apex is near the object. Such a lens allows the top and sides of a cylinder, or cube, to be viewed simultaneously, albeit with considerable distortion of perspective. A certain type of hypercentric lens can be used to view through a small aperture, for example, through a letter-box to obtain a wide-angle view of the side walls of the lobby inside. Hypercentric lenses are well suited for tasks such as inspecting the side-walls of sewers and pipes. The lens and camera do not have to be inserted into the bore. To be effective, both telecentric and hypercentric lenses require large front (objective) lens elements. This can be a problem when big objects are to be imaged, as large conventional lenses are expensive. Fresnel lenses provide an attractive alternative. This type of lens is made in the form of a flat plate with numerous specially shaped grooves on its surface. These act as tiny prisms and collectively focus a parallel light beam just like a conventional lens. Apart from being used in telecentric and hypercentric assemblies, Fresnel lenses are used in projectors, and wherever a large or light-weight lens is needed. They can be made inexpensively in bulk. A microlens array resembles a Fresnel lens and consists of a flat plate with numerous individual moulded lenslets covering its surface. The lenslets are normally very small but are large enough to avoid creating diffraction effects. Such arrays can perform some remarkable functions, and are used in diffusers, beamsplitters and for anamorphic imaging. Many solid-state photo-detector arrays have single lenslets placed over each photo-sensitive site (➲ Chaps. 10 and ➲ 11). This increases the light-collecting power of the camera and avoids blurring due to "spill-over" of light scattered within the sensor chip.

1.7.1.6 Chapter 7: Illumination Sources

Most of the other chapters in this book emphasise the critical role that lighting has for Machine Vision. In order to obtain the benefits of detailed optical design (➲ Chaps. 5 and ➲ 6) and careful camera selection (➲ Chaps. 10 and ➲ 11), equal consideration must be given to the choice of lamp. *A desk-lamp is never the answer!* Paradoxically, to many people, lighting is a "black art"; our everyday experience does not alert us to the subtleties that we need to study. For example, in ➲ Chap. 3, the ability of the human eye-brain complex to compensate for changes in the colour and intensity of ambient lighting was emphasised. Most people are unaware of the slow degradation of the room lighting at home, as lamps get older,. Many people change a light bulb and then remark with surprise how bright the room suddenly appears. When trying to discriminate colours using a machine (➲ Chap. 16), precise control of the colour of the lighting is crucial. Until the end of the last century, most people were aware of just a few types of electric lamps. Incandescant filament bulbs and fluorescent tubes were used in the home, while mercury discharge lamps were common in places of work, such as factories and warehouses. Street lighting used mercury and sodium lamps, while advertising relied heavily on "neon" displays. (These devices use other gases, in addition to neon, to produce a range of colours.) Nowadays, there is a greater variety of lamps available, including a wide range of LED-based lighting units, cold cathode tubes, compact fluorescent bulbs and plasma panels. For a long time, arc lamps have been used in specialised scientific and industrial applications. To most people, lamps have just four important properties:

- Brightness, with no regard for the spatial radiation pattern
- Efficiency, measured by the ratio of the optical emission power to the electrical power consumed
- Life-time, measured simply by how long before it becomes so dim as to be useless, or stops emitting altogether
- Colour. This is usually just a vague notion, based on descriptions such as *soft*, *harsh*, *cold*, *warm*, etc.

For Machine Vision, a more scientific specification is needed and must take into account

- 3D radiation pattern
- Ability to control brightness
- Driver (AC/DC, frequency, closed-loop control)
- Electrical noise generated
- Heat produced
- Infra-red content
- Long-term degradation of emitted power
- Shape
- Short-tem stability of emitted power
- Spectrum (This requires a much more detailed analysis than simply stating the "colour.")
- Ultra-violet content
- Power-line ripple

In recent years, there have been major advances in the design of LEDs specifically for Machine Vision. It is possible to build arrays of any given shape, consisting of numerous individual LED "point sources," each of which has precisely known spectral characteristics. They are efficient and stable over a long life-time. The lighting pattern can be adjusted dynamically, under software control, thereby providing a powerful new dimension to Machine Vision system design that was previously unavailable.

1.7.1.7 Chapters 8 and 40: Lighting-Viewing Methods

The image acquisition sub-system consists of one, or more, light sources (➲ Chap. 7), lenses and possibly other optical components (➲ Chaps. 5 and ➲ 6), as well as one, or more, opto-electronic transducers (cameras, ➲ Chaps. 10 and ➲ 11; or scanners, ➲ Chap. 9). Simply knowing what "building bricks" are available is not enough: we must be able to build them into an integrated system. There is another facet to be considered: object presentation and transport. This requires that we address issues such as holding the object and moving it before, during and after image capture. The type of viewing background is also important, as are seemingly mundane issues, such as eliminating ambient light, building vibration-proof optical rigs and keeping optical components clean, dry and cool. In many applications, the placement of the camera relative to the object under inspection and the lights is critical; the image quality derived from the same object may vary from very poor to excellent under different lighting-viewing conditions. The chapters just cited are concerned with "component level" factors, while this one deals with high-level issues. On the other hand, ➲ Chap. 11 and the Machine Vision System Design Questionnaire (➲ Appendix B) present practical advice for optical system engineers. This is based on common sense, experimentation and experience. ➲ Chapter 8 introduces the much larger catalogue of Lighting-Viewing Methods (LVMs) presented in ➲ Chap. 40. To understand the importance of choosing an appropriate lighting-viewing method, consider the task of inspecting objects on a continuously moving conveyor belt. (This is a popular method for transporting objects around a factory, since it avoids building a more complicated, less efficient and less robust parts-handling system that would be needed if an indexed image acquisition system were used.) This allows various options, possibly based around line-scan cameras, progressive scan (array) cameras, cylindrical optics, strobed lighting, etc. Questions regarding system synchronisation, conveyor-speed measurement, parts-present detectors, and visual background also arise. The LVM catalogue addresses these issues. often necessarily presenting information in qualitative, pragmatic form. There can be no rigid formula-based approach to optical system design. This does not, in any way, diminish the need for sound mathematical analysis, where it can be applied appropriately, as for example, in lens design. There can be no equation to tell us when it is sensible to use, say, diffuse, or dark-field illumination, monochromatic or polarised light, or a combination of these. The author has long contemplated trying to embody information of this kind in an expert system but still believes that it would not match human intuition, and intelligence.

1.7.1.8 Chapter 9: Laser Scanners

Although solid-state digital cameras are the most frequently used image sensors in Machine Vision systems, scanners have much to offer, particularly in certain specialised applications. Scanners provide an attractive alternative to solid-state cameras for detecting changes in reflectance, colour, surface roughness and polarisation. Of course, scanners are familiar in our everyday lives, as they are used in retail bar-code readers. They are typically used for inspecting flat, web (sheet) materials, such as rolled steel, plastic film and paper. They have been used to examine large internal bores, such as hydraulic cylinders, pipes, tunnels, sewers, etc. Scanners are able to generate high resolution images. A laser scanner is able to measure surface reflectance over a single very narrow and precisely defined wave-band. It can also be used to obtain a number of exactly registered images simultaneously, each derived from a different narrow spectral band. These may have wave-lengths below, within, or above the visible part of the spectrum (VIS). It is also possible to obtain several images, again aligned exactly with one another, and derived from specular and near-specular reflections. Moreover, it is possible to separate the light reflected from the surface under investigation into two orthogonally polarised beams. This is useful, for example, to detect defects in transparent coatings.

The scan pattern can be varied; it may be linear, circular, spiral, helical, or radial. It may even be random, or follow a complex path, such as a Hilbert curve, although the data rate is rather low in these last two cases. The same optical arrangement might be used in some applications for both sensing, and marking. When used with light-sensitive coatings, scanners can be used to create complex patterns, such as those found on printing plates and printed circuit boards. The same scanner might then be used to inspect them, or burn patterns. Laser scanners have even been used to examine photographic film! There are two "tricks" to this: select the scanner wavelength carefully and keep the laser spot moving very rapidly. Laser scanners are ideally suited for surface-inspection applications, such as examining floor tiles, paper, plastic sheeting, plain wooden surfaces, animal hides, fabric, products made in the form of a flat strip and those where it is not possible to put a camera close to the object undergoing inspection. They are ideal for examining red-hot metal and working in areas where there is a high level of radioactivity and where there is a toxic, explosive or biological hazard.

Scanners can be made highly specific to suit particular applications. It is possible, for example, to use UV or VIS light with a narrow spectral peak, to excite fluorescent, phosphorescent and photochromic materials. (The last mentioned switches colour in a reversible manner.) This has potential applications in security applications, such as checking document/product authenticity. The scanner light source (laser) can be pulsed, which is particularly useful for detecting phosphorescence. A scanned laser beam can be projected through a very narrow slit, making it possible to keep light levels low everywhere, except at the immediate point of inspection. It is possible to build a laser scanner that can examine objects that are a long way away. Hence, it is possible to use a scanner to detect oil leaks in remote and complex pipe-work. Since organic substances often fluoresce, laser scanners can detect infection by micro-organisms on food-materials and products. Scanners can also be used to measure range and 3D shape (➋ Chap. 15). There are two distinct ways to do this: by trangulation and by measuring time-of-flight of the light beam. (This is the principle of operation of Lidar.) Laser scanners are often used in automatically guided vehicles to detect obstacles.

1.7.1.9 Chapter 10: Cameras

A solid-state camera forms the very heart of most modern vision systems. Huge advances have been made in sensor technology since the cathode ray tube (CRT) was the only realistic way to acquire image data. CRT cameras are large, heavy, physically fragile and susceptible to defects such as blooming, comet-tail, image burn-in and non-linear spatial sampling. They require high power to heat the filament and use dangerously high operating voltages. They are also sensitive to magnetic fields. On the other hand, solid-state sensors are much smaller, lighter in weight, physically robust and provide very accurate spatial sampling. They require less power and use safe, low operating voltages. Modern cameras are reasonably tolerant of optical overload but they do not tolerate high temperatures so well as CRTs and are more easily damaged by high-energy particle beams.

Although there are several different types of solid-state sensor, nearly all rely on the generation of hole-electron pairs in silicon as a response to incident light. There are several different ways to access the charge pattern developed by the photo-detector array and convert it into a video signal. Photo-detector elements can be arranged in a single row (line-scan camera), multiple rows (time delay integration camera) rectangle/square (area-scan camera), circle (circle-scan camera), along the "spokes" of a wheel-spoke, or around concentric circles. Colour filters can be built into a camera, making it sensitive to different wave-bands. These can cover the near infra-read (NIR) and near ultra-violet (NUV) wave-bands (Chap. 5 and Appendix I), or generating the familiar RGB signals from the visible (VIS) band.

This chapter concentrates on the physics and construction (architecture) of solid-state sensors, while Chap. 11 discusses their characteristics and the implications for their application in Machine Vision. A camera is not nearly as sophisticated as the human eye (Chap. 3). In particular, it does not possess such a wide dynamic range. For this reason, algorithmic techniques that are relatively insensitive to varying light levels are important (Chap. 13). Cameras are always fitted with lenses and these are, of course, an integral part of the lighting-optical sub-system (see Chaps. 5–8, 40).

Since they are sensitive to heat, cameras must be kept cool. This can be achieved using a hollow-walled metal box with cold air, water or even liquid nitrogen circulating within its cavity (Chap. 40). Chapter 9, which describes laser flying spot scanners, alerts the reader to other possible ways to obtain image data. These devices and more conventional cameras provide solutions for different niches, scanners being more suitable for high-speed inspection of fast-moving web-like materials. Modern cameras have contributed a great deal to the development of Machine Vision; without them, it is doubtful whether it would be worthwhile writing a book like this!

1.7.1.10 Chapter 11: Selecting Cameras for Machine Vision

Modern solid-state cameras are tiny devices that rely on detecting quantum-level energy changes occurring when photons cause the creation of electron-hole pairs in a semiconductor. The physical basis of this process is described in Chap. 10, while this chapter discusses the practical issues that impinge on the design of Machine Vision systems. Cameras have many subtle features that should be understood in order to obtain the best possible results. For example, the generation of electrical noise is an inevitable consequence of the photo-detection process. Its effects can be minimised by operating the camera at the optimal light level. This must be low enough to avoid saturation, blooming and "bleeding" of charge from one photo-detector site to another. While the geometry of the photo-detector array is defined very precisely during manufacture, its sensitivity varies slightly from one pixel to another. A simple algorithm can compensate for this. Furthermore, a fixed noise-like pattern is effectively superimposed on every image that the camera "sees." This can be removed very effectively by the simple expedient of storing and subtracting an image of the noise pattern. Many other subtleties about camera design and operation are discussed. Taking them properly into account can make a big difference to the quality of the digital images that ensue. Camera designers have done much to compensate for their variations and quirks. Many of the innovations in solid-state photo-sensors have been directed towards improving personal digital cameras and camcorders but we must not take it for granted that these are always appropriate for Machine Vision applications. Automatic Gain Control (AGC) is one example of an " improvement" that rarely suits our particular needs. The reason is fairly obvious: in a typical inspection cell, the camera, lights and object being examined are all in fixed position relative to one another, the products are very much alike and the lamp brightness does not (should not!) vary much. Viewing under nearly constant conditions, like this, makes

AGC unnecessary. Indeed, it can actually make matters worse, because it can obscure changes, such as lamp aging, that we need to know about. In the early days of Machine Vision, the advantage of having pixels that are square, not rectangular, was not universally understood. This is widely appreciated now and is reflected in the layout of nearly all photo-detector arrays. This is such a simple change that its real significance could easily be lost again. Line-scan cameras offer some important and distinctive features, particularly for inspecting continuous webs and objects on a conveyor. However, to realise these benefits, it is necessary to know the relationship between the frequency of the camera clock signal and the speed of the conveyor belt. This is needed to ensure that the pixels are square. If they are not, the image processing algorithms become more complicated. Subtleties and interactions like these are the very essence of Machine Vision. Line-scan cameras allow cylindrical lenses and mirrors to be used (➲ Chap. 40). It is possible to build very large 1-dimensional optical systems quite cheaply. For example, it might even be possible to use very cheap, extruded Fresnel lenses (➲ Chap. 6). Individual line-scan cameras can be butted end to end, effectively to build very long line-scan sensors. In this way, it is possible to build a very high resolution but modestly priced line-scan system that is able to inspect wide webs and continuously moving objects.

1.7.1.11 Chapter 12: X-ray Inspection

Automated X-ray Inspection (AXI) is particularly useful for two types of application: detecting foreign bodies in food and examining the internal detail of safety-critical, high-value products. It may seem that AXI is simply a straightforward extension of Machine Vision techniques as discussed elsewhere in this book. Most x-ray systems project a beam through an attenuating medium. This appears to be very similar to viewing a transparent object with back-lighting (➲ Chap. 40). For most materials, there is very little deflection of the x-ray beam due to refraction (➲ Chap. 5). However, there are practical issues that complicate this naive view. There is only a limited scope for altering the energy spectrum (wave-band) of the beam nor is it easy to build x-ray mirrors and lenses. Since the x-ray source, a specially designed cathode ray tube, generates a divergent (conical) beam, the images are not quite as simple as we might imagine. This introduces parallax errors. Moreover, objects near the source appear to be bigger than those further away. Features along the system axis are superimposed, often creating a jumbled picture that is difficult to analyse so that we can associate object edges with image contours. A useful approach to understanding x-rays images is to construct a synthetic image by emulating the x-ray source and target object. We may produce a crude approximation so that the vision engineer can associate object and image features together. A more refined model can generate an image that is close to that produced by an ideal (perfect) object. This image can be used as a reference for comparison with the images produced by test samples. If the objects being inspected are manufactured to close tolerance and are accurately registered each time, a point-by-point comparison of the test and reference images may suffice as a basis for the inspection algorithm. This is known as template fitting, which is normally deprecated in Machine Vision (➲ Chaps. 14 and ➲ 19). In some cases, it is necessary to warp one of the images slightly, so that it will fit onto the other. The warping and fitting of a rubber template can be quite complicated, compared to most of the algorithms described in ➲ Chaps. 14, ➲ 18 and ➲ 19. More sophisticated image processing procedures are probably needed when the objects being examined exhibit even a modest level of variability. To summarise, AXI has its own set of issues that distinguish it from other areas of Machine Vision. There is one particular point that emphasises this more than any other: the danger that ionising radiation presents. Subsequently, there is a need for good radiation screening and safety features, such as cut-off switches on equipment doors. This is not an option; it is a legal requirement!

1.7.1.12 Chapter 13: Illumination-Invariant Image Processing

Machine Vision is necessarily concerned with four different media in which we represent the appearance of an object: optics, analogue electronics, digital electronics and software. Thus, there are several ways in which significant information can become corrupted. The importance of the lighting-viewing configuration cannot be overstated; the maxim "*Rubbish in, rubbish out*" should always be remembered. For this reason, there can be no excuse for taking a casual approach to image formation and acquisition. First, we must choose the most appropriate light source (➲ Chap. 7). Then, we should experiment with the lighting-viewing arrangement (➲ Chaps. 8 and ➲ 40), before adopting a proper scientific approach to the detailed design of the optical sub-system (➲ Chap. 5) We then choose the most suitable camera (➲ Chaps. 9–11 and ➲ 14). The output of the camera is digitised and the resulting image array is ready for processing (➲ Chaps. 14, ➲ 21 and ➲ 22). However careful we are in the design process, the result will be a series of digital images varying over time, as a result of ambient light fluctuations, lamp ageing, variations in lamp drive current, accumulation of dirt on optical surfaces, variations in the gain of the camera, changes in the reflectivity of the object under inspection and its background. All of these can give the same effect as changing lamp brightness. It is possible to compensate for these effects, by judiciously choosing the image processing strategy. It has been emphasised repeatedly that, despite its obvious attraction, simple fixed-level thresholding is not reliable as a way to segment a grey-scale image (➲ Chap. 14). To overcome the problems, several methods for improving thresholding are used extensively elsewhere in this book: contrast enhancement [QT: *enc*], histogram equalisation [QT: *heq*], morphology (➲ Chaps. 19 and ➲ 20) and high-pass filtering (➲ Chap. 14). This chapter investigates the general problem in more depth and in a formal manner. Human beings are normally insensitive, even to quite large variations in lamp brightness (and colour). Being able to compensate for diurnal fluctuations in natural light levels (➲ Chap. 8) has obvious advantages for an organism but does mean that we are probably unaware of the need for the methods described here. However, a few minutes working with an image processor (➲ Chaps. 21, ➲ 22 and ➲ 41) will clearly demonstrate that many of our prior conceptions are wrong.

1.7.1.13 Chapter 14: Basic Machine Vision Techniques

The ability to process digital images within a computer is the youngest of the enabling technologies needed for Machine Vision. Twenty years ago, the concepts of digital image processing were almost unknown, except to a small number of researchers. With the advent of digital cameras and photo-editing programs, such as Photoshop™, there is less mystique – but there is still a lot of mist! A very common mistake that newcomers to our subject make is to expect that Machine Vision will necessarily emulate human vision as they perceive it to be. As we have already seen in Chap. 3, there is a wide diversity of vision paradigms in the biological world; human vision is not the only success story. There is no fundamental reason why we should constrain Machine Vision to emulate human ability. It could be inspired by other biological systems, such as those of a fish, frog or insect, or it may owe nothing to nature. This chapter begins with a discussion of ways of representing images in a form suitable for manipulation within a computer. The most commonly used format is that of a 2- or 3-dimensional array, although others present advantages in certain specific situations. Sometimes a conventional digital computer is simply not fast enough and dedicated electronic systems are required. The array format is often suitable in that situation too. The real "value" of an image processing operator depends on its ability to perform some useful function *and* its cost/speed of implementation. A large range of image processing calculations can be performed in an array of logic gates, registers and accumulators. These are commonly referred to as "low-level" functions and are discussed in this chapter. On the other hand, some functions require abstract logical reasoning. This lies within the subject domain commonly called *Artificial Intelligence*. Discussion of this topic is deferred until Chap. 23. Chapters 18 and 19 provide more detailed coverage of two important topics introduced here, while Chaps. 17 and 20 explain how algorithms can often be reformulated to improve speed, or replaced by faster but approximate methods (heuristics). Nearly all of the low-level operations mentioned in this chapter can be implemented using the QT software (described in Chaps. 21 and 41), or NeatVision in Chap. 22. A great deal of time, effort and money has been devoted to building special hardware for low-level image processing. This includes designing special-purpose integrated circuits for tasks such image filtering and edge detection. The design of Field Programmable Gate Arrays (FPGAs) for high-speed image processing is discussed in Chap. 25. A popular approach has been to provide image processing board that can be installed in a general-purpose computer. A decade ago this was of crucial importance but, with the advent of high-speed general-purpose computers, there is less need for add-on hardware. Of equal importance is the fact that a wide range of cameras can be connected to a general-purpose computer, via a USB, Ethernet or Fire-wire interface.

1.7.1.14 Chapter 15: Imaging and Range Image Processing

The ability to acquire, process and analyse 3D range information is important for a great many practical applications, including industrial parts inspection and robotic parts handling. The creation of so-called *range images* (also called *range maps*, or *depth maps*) cannot be achieved by simply pointing a camera at an object; special optical-illumination arrangements are needed. Several different methods have been devised for acquiring range information, including: measuring time of flight of a laser or microwave beam (mm/cm wavelengths), triangulation using one/several projected white or coloured light stripes (Chap. 40), triangulation using active pattern projection, Moiré patterns, Talbot fringes and optical interference fringes. Time-of-flight range-finding methods are faster but are built around a laser scanner (Chap. 9). Whichever method is used, the resulting range image is likely to be quite noisy. Hence, noise removal in range image data is an important first step in the analysis sequence. Some 3D sensing methods do not differentiate between positive and negative range changes, which leads to complications in the analysis of depth maps. Another difficulty is caused by occlusion near step-like edges in the object surface. These occur quite frequently in depth maps generated using triangulation as a result of parallax. (The illumination path and camera axis are at about 45° to one another.) Depth maps have potential uses in a very wide variety of practical applications, including grading fruit, weighing meat, inspecting baked goods, inspecting car-body panels, measuring aircraft turbine blades, guiding autonomous vehicles and robots, reverse engineering (starting with hand-carved prototypes) and manufacturing prosthetic devices (e.g., fitting artificial limbs for amputees). Some applications require only low-precision measurement of range. For example, the manufacture of food products, such as loaves, cakes, pies, etc. is quite satisfactorily served with millimetre-level precision. Typical objectives in the baking industry include weighing, estimating product volume, measuring size (to ensure that the product will fit into its wrapping) and simply judging aesthetic appeal. Examining agricultural and horticultural products does not normally require high precision. This is in marked contrast with the automobile and aircraft industries, where high-precision measurements (micron level) of surface form are needed. A depth map can be processed as if it were a standard monochrome image (➋ Chap. 14). Isophotes in a depth map correspond to height contours on the object under examination. Although the relationship between surface height and depth map intensity is monotonic, it may not be linear. Once a ranging system has been calibrated, the mapping function between these variables is fixed and can be stored in a look-up-table. Simple thresholding provides a convenient means of locating surface features, while edge detection and morphology (➋ Chaps. 19 and ➋ 20) are important for feature location.

1.7.1.15 Chapter 16: Colour Recognition

Despite its many references to the firmament, the Bible does not mention the fact that the (cloudless) sky is blue. Nor does Homer say so in the *Iliad*. Everybody with reasonably good sight knows the colour of the sky at noon, so saying "*The sky is blue*" makes a point about the colour term "*blue*" but does not tell us anything about the sky – except that there are no clouds. Saying that a banana is "*yellow*" indicates that it is ripe and ready to eat but it does not tell us precisely what a banana is really like. A sulphur-coloured banana would appear very odd but it would still be "yellow." There is no absolute truth involved in the naming of colours. In English, we say that the sky is "*blue*" and in Welsh "*glas*." However, "*glas*" can also be used to refer to grass and trees. In other words, "*blue*" and "*glas*" do not have the same meaning. I was taught at school that the French word for "*yellow*" is "*jaune*," which is not strictly true. To summarise, the names of colours have no fixed immutable meaning, so we can only build machines that recognise colours by emulating a human being, whom we trust to be acting reasonably. The ephemeral nature of colour must not deter us from using what is a very powerful way to describe the world. Colour is an important feature of many artifacts and natural products, as well as a large proportion of the packaging materials used for them. Verifying that an industrial artifact has its coloured features intact and correctly placed is, of course, important commercially. The ability to segment an image on the basis of the various colours it contains allows standard (binary) image analysis software/hardware to be used for inspection. For example, measuring the areas and aspect ratios of each of the various "yellow" features on the printed labels for cartons containing pharmaceutical materials, may be a useful way to guarantee that the product has been packaged properly. This is typical of what the author calls "*coarse colour discrimination*." This may be as simple as deciding whether each pixel is "*red*," "*blue*," "*yellow*" or "*something else*." Isolating "*yellow*" features enables us to generate a binary image, which can then be analysed using the binary image processing operators described in ➲ Chap. 14. Colour recognition can be based on standard Pattern Recognition techniques (➲ Chap. 24), or on interactive adjustment of a so-called *Programmable Colour Filter* (➲ Chap. 16). The latter relies on the use of a look-up-table (LUT) to associate RGB or HSI vectors with symbolic colour names. Finding appropriate entries for this LUT relies on human intuition and intelligence, rather than mathematical formulae, so skill and experience is needed to design a PCF. This chapter should be read in conjunction with ➲ Chap. 4.

1.7.1.16 Chapter 17: Algorithms, Approximations and Heuristics

Just as there are numerous ways to describe something in English, there are many options when defining an image processing algorithm. Ideas outlined in ➲ Chap. 14 do not necessarily indicate the optimal implementation. Of course, that depends on the hardware-software platform that is being used. This is a choice that has to be made early on in the design process and probably has little to do with Machine Vision *per se*. It might, for example, be based on commercial/managerial criteria, such as long-term stability of equipment supply. During the early stages of an application study, a prototypical algorithm might be devised, using an interactive image processing system, such as QT (➲ Chaps. 21 and ➲ 41), or NeatVision (➲ Chap. 22) The obvious approach is to recode that algorithm directly in whatever language is currently preferred by the company: C, C++ and Java are obvious favourites. Another possibility explored in this chapter is to reformulate the algorithm itself. This may be a trivial change, such as using logarithms and addition, rather than multiplication. In some instances, we might decide to use a simple look-up table to perform multiplication instead. This is perfectly feasible for short-integer (8 bit) arithmetic. Since digital memory is inexpensive nowadays, this might be a preferred option even when using 16 bit arithmetic, perhaps when building dedicated hardware. Imagine that we needed to square a large number of integers (X^2) on a regular basis. Clearly, we could multiply, use logarithms or a look-up table (LUT). Another option is to use an approximation, such as the modulus function ($|X|$). In some cases, this is perfectly reasonable as, in many calculations, there is no fundamental reason why X^2 is any better than $|X|$. Sometimes, a more fundamental reformulation is needed to achieve an efficient implementation. A prime example of this is the SKIPSM algorithm described in ➲ Chap. 20. Making appropriate approximations is an integral part of almost every engineering design exercise. In MV, this may involve substituting one image filter, such as edge detector (➲ Chap. 14), for another that is easier to implement on a given platform. The term "algorithm" is often used even where there is no mathematical justification for believing it to be anything more than a "good idea"; the word "heuristic" would be more appropriate. A heuristic is a rule of thumb (an informally defined procedure) that usually provides the result we desire – but it may not always do so! A heuristic may actually be optimal (algorithmic) without our being able to prove it. A heuristic may be used directly to calculate the result we need, or it may be used to decide which computational process is most appropriate to apply. An analogy may help to highlight the lesson of this chapter: to fasten two pieces of material together, we might use a rivet, nut and bolt, welding, pop-rivet, a clamp or glue. Similarly, in MV, there are often many procedures that can achieve results that would be "good enough." (Note the use of that term; we did not say "optimal." After all, this is engineering, not science, or pure mathematics!)

1.7.1.17 Chapter 18: Object Location Using the Hough Transform

A broad range of image processing operators was introduced in Chap. 14, while a detailed discussion of morphology to follow in Chap. 19 (also see Chap. 20). In view of its importance for Machine Vision, another important family of functions is discussed in depth here. One of the prime uses of the Hough Transform (HT) is to identify and reconstruct "broken" lines. When an edge detection operator is applied to a noisy monochrome image, the contours that it highlights are unlikely to be continuous; typically, we find that functions such as the Sobel or Canny operators, produce a set of bright spots concentrated along a series of discontinuous lines and arcs. Detecting linear structures within noisy images like these is of considerable importance, particularly when the object under inspection is placed in front of the camera in an unknown position and/or orientation. This is the realm of the HT. Imagine an object being dropped haphazardly onto a table. One of the first tasks for almost any inspection algorithm is to determine the position and orientation of that object. If it has one, or more, straight edges, the HT can be used to estimate their orientation and position coordinates. The physical position and orientation of the object can then be estimated. Notice that this relies on the presence of straight-line features. In this role, the HT competes with the heuristic ad hoc methods outlined in Appendix F as well as the Radon transform (Chaps. 14 and 41). Each has its own application domain. Throughout this book we constantly remind the reader that Machine Vision is a specialised area of Systems Engineering. Here is a prime example of a situation in which remembering this general principle is vitally important. Ingenious design of the mechanical handling sub-system can, in some applications, circumvent the need for sophisticated computational techniques such the Hough Transform. Sometimes, it is possible to build a transport mechanism that limits the degree of variation of the object's orientation. If its position, but not orientation is fixed, it may be possible to determine the latter using a simple scan to plot the intensity profile around a circular path. In certain applications, the object may possess obvious physical features, such as holes, that allow the orientation to be determined quickly and easily. (Appendix F) However, none of these fortuitous features is universally present; in those situations where object placement is uncertain, the HT has a pivotal role. The Hough Transform is more general than being just one more method for finding straight lines. It can detect circles and ellipses too. Moreover, it can do so in situations where objects overlap, or for other reasons, there is some occlusion. The ability to detect ellipses may seem to be of no more than academic importance. However, when we realise that viewing a circle obliquely produces an ellipse in the image, we see that this is of considerable practical value. Again, we see that the HT competes with other techniques when there is sufficient application knowledge to be able to predict image appearance. In view of these comments, the Hough Transform should perhaps be regarded as a tool for location, orientation and recognition, rather than verification. The value of the HT is its ability to cope with situations where noise and/or dirt causes other methods to fail.

1.7.1.18 Chapter 19: Morphological Image Processing

The principles of image processing for Machine Vision are described in broad terms in Chap. 14. We turn our attention now to consider in more detail an important class of operations, namely morphology. These provide powerful tools for filtering both grey-scale and binary images, as well as identifying and locating significant features. Noise removal is an obvious first step in many vision tasks and several distinct morphological operators are available for this. We can, for example, eliminate bright spots, or dark spots separately, or together. Alternatively, we can also remove bright and/or dark streaks at specific angles, or at any angle. Noise reduction operators can be applied to the "raw" video (analogue) signal, or after segmentation, to a binary image. To understand the importance and role of morphology in feature detection, consider the task of reading printed text by machine. A naive approach to recognising a capital letter A uses template matching; we place a perfect A over the sample of text and wiggle it around, until we obtain a perfect, or simply a "good," fit. Despite its strong intuitive appeal, this simple procedure does not work very well. It is very sensitive to variations of font, size, orientation, skew (italics), as well as paper and print quality. A more robust approach is to identify smaller features: "components" of the alphanumeric characters. Thus, we might instead look for places where simple patterns, such as

| — / ● ∩ ⊂

fit on the image. Features detected by morphology may lie in the foreground, or background. For example, a printed letter "O" may be located by looking for a thin black circle (figure), or a solid white disc: the "hole" in the middle. Morphology is important for Machine Vision because we can design feature detectors easily and intuitively, to suit the application requirements. This chapter concentrates on the use of morphological operators, while ➋ Chap. 20 discusses how they can be reformulated to operate in hardware or software, to obtain high processing speed. The QT software described in ➋ Chaps. 21 and ➋ 41 facilitates experimentation with morphological operators.

1.7.1.19 Chapter 20: Image Processing Using Finite-State Machines

Morphology has been described in ➋ Chap. 19. It is a popular and versatile method of detecting and measuring features in both grey-scale and binary images. Common sense suggests that the bigger the mask to be fitted (structuring element) the slower the execution in software. This chapter shows why this is not necessarily so. By reformulating the erosion-dilation algorithm, it is possible to obtain an implementation, called SKIPSM, whose execution time is independent of the size of the structuring element. This algorithm lends itself readily to either software or hardware implementation. Therefore, it opens possibilities that would otherwise be impractical. Intelligence is needed to build and apply a SKIPSM machine, because it cannot provide a compact implementation for all structuring elements. However, important shapes, such as open and closed rectangles, diamonds and circles, crosses (and), bars (—, I, \ and /), loops (∩, ∪ and ⊃), and Ls (in all four configurations) are all amenable to this approach. Furthermore, they can be combined to form composite structuring elements, such as

= ^ ∠ ✕ ⊥ 〈 ✱ ⊙ ◉ ● ⊗ ◗ ⊠ ⊡ ▣ ◈ ◈ ◈ △ ▲ ▼

One of the interesting features of SKIPSM is its ability to perform several morphology operations in parallel. For example, it is possible to perform dilation and erosion at the same time. Surprisingly, this does not increase the computation time significantly. It is also able to improve the speed of certain important functions using a generalised SKIPSM approach. These include grass-fire transform, thinning, skeletonisation, condensation (shrinking each distinct blob to a single point), detecting skeleton limb ends and joints. Grey-scale morphology is also discussed and algorithms exist for edge detection, low-pass filtering and median filtering.

Experimenting with morphology is possible using the QT interactive software (Chaps. 21 and 41). A more generalised discussion of algorithm reformulation to ease implementation is to be found in Chap. 17.

1.7.1.20 Chapters 21 and 41: QT – Prototyping Image Processing System

Chapters 21 and 22 describe rapid prototyping packages (QT and NeatVision respectively) that were designed specifically for Machine Vision applications. The essential feature of QT is rapid and simple interaction [2]. The user views a picture on the display screen and types a three-letter mnemonic command and possibly a list of parameters. The image is processed and the result displayed immediately. For most commands, operating on images of reasonable size, the processing time is well under 1 second. In fact, the user sees two on-screen images: one ready to be processed and the input image for the previous operation. This *modus operandi* was developed in the 1970s and has not been superseded, as it is remarkably effective in the hands of an experienced user. On many occasions, the author has entertained industrial visitors for the day. They have brought sample widgets with them to illustrate the requirements for an inspection application. Perhaps for an hour or two, the author experimented with the lighting and camera. When a good image was produced, it was analysed using QT, or one of its several predecessors. Many times, it was possible to develop a prototype image processing algorithm within a few minutes, rarely over an hour and only occasionally not at all. (In the last eventuality, the result was still valuable as it indicated a non-viable project.) The visitors left with a clear picture of the possibilities and limitations of the options available as solutions for that particular application. It must be repeated that, in the author's experience, this has happened numerous times. The power of this approach is only really appreciated when seeing QT in operation. Many of the applications used to illustrate this book were evaluated using QT (In particular, see Chaps. 16, 19, 27, 31, 38). At the time of writing, QT has over 400 functions, written in MATLAB™. These are described and illustrated in Chap. 41. The QT software is available from the publisher's web site http://extras.springer.com/2012/978-1-84996-169-1.

1.7.1.21 Chapter 22: NeatVision: Development Environment for Machine Vision Engineers

NeatVision is another prototyping system for image processing and was designed to be easy to operate, particularly by inexperienced users. Like QT, it has a rich instruction repertoire but employs an entirely different user interface; an image processing program is represented by a network of interconnected modules. Images and parameters derived from them flow between modules, in an obvious and intuitive manner. Each primitive and high-level image processing function is represented visually, by a rectangle with colour-coded inputs and outputs, indicating the corresponding data type. A high-level compound function can be defined by a network incorporating primitive and other compound functions. Once a network has been proven to work satisfactorily, it can be "condensed" into a new module, again represented visually by a rectangular icon. The new module can then be incorporated into other programs, like any primitive function. NeatVision has virtually unlimited capabilities for expanding the command repertoire. While novice users normally prefer NeatVision's visual programming environment, more experienced people are often happier with a command-line interface, such as that provided by QT. In both cases, the user has to learn the function of each module before he becomes an effective user. The advantages of NeatVision are the same as those of any other visual programming environment, being well suited to writing programs that have a rich branching structure. On the other hand, QT and other command-line systems, allow rapid interactive program development, since the command-to-command cycle is shorter. NeatVision was written using JAVA, while QT is based on MATLAB. In both cases, the underlying software is very powerful but rather cumbersome and unsuitable for rapid prototyping. One reason for this is that both Java and MATLAB require the user to remember, and type, the names of image files. Neither NeatVision nor QT do. That may seem to be a small detail, but on such features as this a system is often adopted or forsaken. Although they are very different, the user interfaces provided by NeatVision and QT greatly improve the ease of use and the ability to operate effectively in an interactive manner. Experience has shown repeatedly that this is a key feature when designing algorithms for Machine Vision. Unlike conventional programming, a vision engineer is never quite sure what step to perform next – he must be able to experiment. This is totally alien in normal programming; imagine not being sure what number, or arithmetic operator, to write into a mathematical expression. Developing vision algorithms requires a fundamentally different approach. Choosing a command-line or graphical interface is best left to the vision engineer. Both are effective!

1.7.1.22 Chapter 23: Intelligent Image Processing Using Prolog

Prolog often scares programmers experienced in using standard computer languages, such as C, C++ and Java. Prolog is different! For example, it has no FOR loops and no IF. . .THEN. . .ELSE statements. It does not even allow instructions! It is not particularly good at performing arithmetic, so it would not be sensible to write image processing function, such as image filtering, using Prolog. Despite these points, Prolog has much to offer. Consider the following task: you have lost a book and have asked a friend to find it for you. You would very likely describe the book, indicating its size, colour and subject matter/title. Your friend then goes and looks for the book, without any explicit statement from you about where to look. In this analogy, the description of the book corresponds to the Prolog "program" (properly called an *application*), while your friend operates like a Prolog theorem prover. Another example: suppose you wanted to find a husband/wife for yourself, assuming that there is a vacancy for that role. This time, you would specify your "requirements" in the form of a Prolog application and leave the theorem prover to find a potential candidate. Try doing the same in one of the languages mentioned earlier. General spatial relationships, such as *above*, *left-of*, *inside* and *between* can be expressed very succinctly in Prolog but cannot be represented nearly so readily using a language such as Java, or in MATLAB or QT (➜ Chaps. 21 and ➜ 41). Standard Prolog has no special facilities for handling images. However, a simple extension to Prolog allows it to control a QT system as a slave. It is then possible to embed QT commands in Prolog. The result is a language, called PQT, that combines the image processing capabilities of QT with the expressional power of Prolog. Image processing is not performed in Prolog; it simply diverts image-based operations to QT, which passes the results back to Prolog for interpretation. Now, consider the task of verifying that table-ware has been laid out properly for dinner. A person might well specify what he/she regards as being a good layout, *in English*, using qualitative relationships such as *above*, *beside*, *between*, etc. PQT can readily accept a natural language description and then analyse a complex visual scene, containing many different objects, in order to verify that the table is set correctly. Such a task could be performed in Java or QT but the program would be long and awkward to write and understand. The advantage of PQT over QT is expressional convenience. Java programmers rely on the language, to reduce the effort of expressing complex numerical algorithms. In this respect it is far more convenient than assembly code, or Prolog. However, PQT is superior to Java, MATLAB, or QT for describing and analysing highly variable images. But there is more to PQT than that. The same language can be used to plan the seating at the dinner table, ensure that the meal is nutritionally well balanced, inspect the cutlery to make sure it is clean and examine the food for foreign bodies. Of course, it needs the appropriate sensors and illumination. PQT can control the lighting as well!

1.7.1.23 Chapter 24: Pattern Recognition

Sometimes, the definition of a "*good,*" or "*faulty,*" object cannot be expressed properly, either in natural language, or in a more formal way (e.g., mathematical notation or pseudo-code). For example, it is impossible to write a formula that defines colour classes (e.g., "*yellow*" or "*attractive wood grain*"). Any classification that attempts to assess the aesthetic appearance of highly variable objects necessarily relies on subjective judgement. Human beings are able to grade and inspect natural objects, such as growing plants, fruit, vegetables, etc. using unknown subjective criteria that they have learned but cannot say express adequately. The same its true of food products, such as cakes, loaves, pies and pizzas (see ➲ Chaps. 2, ➲ 36 and ➲ 37). Engineering artifacts that require precise measurements, or clearly defined inspection checks (e.g., "*scratch*"/"*no scratch*"), are not normally suitable for Pattern Recognition methods. However, it should not be supposed that manufacturing processes always attain a high level of order everywhere; even supposedly close-tolerance artifacts may have highly variable parts. A fuel injection nozzle (close tolerance) produces a much more variable spray jet. Sometimes, inspection is possible using heuristic rules, expressed in symbolic terms and which a human inspector has been able to formulate. In this case, declarative, Prolog-style programming is useful (➲ Chap. 23). However, there is another large group of potential applications where the ultimate judge of what is "*good,*" or "*not acceptable,*" is a single trusted person, or a committee of experts, who despite their long experience cannot express their reasoning in any sensible way. In such a situation, we can use one of the self-adaptive Pattern Recognition systems described here. A pattern classifier typically accepts a multi-element vector, measuring a fixed set of image features, and learns, by example, to emulate a human teacher. A set of image measurements might conveniently be generated using morphology (➲ Chap. 19). The teacher is expected to do no more than view each object in turn and return a "*good*" *or* "*faulty*" decision for each one. Introspection is not needed, nor is there any attempt to coerce the inspector to provide any more information than a simple binary decision. Pattern classification rules vary in complexity from simple linear discriminants, to much more powerful techniques, including Neural Networks. Another group of classifiers makes decisions based upon distances from a point in high-dimensional measurement space representing the test sample to multiple stored exemplars. Classifiers based on hyperspace distance measurement include the Nearest Neighbour classifier, Compound Classifier (Supplement to ➲ Chap. 16) and others using Radial Basis Functions. Learning rules for linear classifiers have been known since the 1960s. Those for Neural Networks were developed more recently. The Nearest Neighbour learning rules are proven only by experimentation but have been useful in areas such as colour recognition (➲ Chap. 16). Pattern classifiers should be judged statistically on their performance. They do not always produce the "correct decision," as judged by the teacher. It must be accepted from the outset that Pattern Recognition is not a precise way to make decisions. However, it can cope with difficult situations, particularly where there is little detailed application knowledge. It is also important to realise that self-adaptive learning procedures require a great deal of training data to obtain results that are generalisable to unseen samples. Nevertheless, Pattern Recognition provides a valuable way to proceed in many situations where other methods fail. Once learning is complete, decision making can be performed at high speed in software, or specialist hardware.

1.7.1.24 Chapter 25: Implementing Machine Vision Systems Using FPGAs

It has already been noted that Machine Vision requires two quite different types of image processing system: interactive (➲ Chaps. 21 and ➲ 22) for algorithm design and stand-alone for target systems installed on the factory floor. It is important that the latter be able to match the speed of modern manufacturing machinery. Certainly QT (➲ Chap. 21) and NeatVision (➲ Chap. 22) cannot, except in a very few special cases. General purpose computers are becoming ever more powerful and many tasks that were previously impossibly slow in software are now in commonly implemented in that way. It is clear from the examples cited throughout this book that practical inspection algorithms often require numerous passes through a high-resolution image, thereby presenting a heavy load on a general purpose processor. Sometimes, an inspection algorithm can be reformulated in such a way that it is easier or faster to implement (➲ Chaps. 17 and ➲ 20). Nevertheless, there are still occasions when a general purpose computer is just not fast enough. In such a case, dedicated high-speed electronic hardware is needed. It is difficult to believe that the time will ever arrive when this is not the case, as sophisticated new "processor hungry" algorithms are always waiting to be employed. Real-time robot control by vision is always very demanding and requires the very highest processing speed. However, before we resort to using dedicated electronic hardware, we should always ask the question "*Can we improve the lighting-viewing arrangement to make the processing simpler/faster?*" Let us assume that we have satisfied ourselves that we cannot. So, we are forced, on occasions, to build specialist image processing hardware. Field programmable gate arrays (FPGAs) offer a convenient and flexible platform on which real-time machine vision systems may be implemented. ➲ Chapter 25 discusses how FPGA based design differs from conventional software design for implementing image processing algorithms. Simply porting from software to hardware can give disappointing results. The process of transforming an algorithm for mapping it onto an FPGA is described in detail, showing how a range of low-level image processing operations (➲ Chap. 14) may be efficiently implemented. Interfacing external devices to the FPGA, image capture, image display, and interacting with the user are also discussed, together with debugging and algorithm tuning. Once again, we see that Machine Vision is concerned with numerous interacting systems issues, not just the theory of image processing algorithms.

1.7.2 Applications Case Studies and Practical Considerations

➲ Chapters 26–38 illustrate the general points made earlier, describing a variety of applications case studies. Of particular note is the way that systems considerations impinge on the design of a vision system. Cost, simplicity, user interface, lighting, object variability are the main factors that have influenced the designs discussed here. ➲ Chapter 39 concludes by discussing management, ethical and social issues.

1.7.2.1 Chapter 26: Very Low-Cost In-Process Gauging System

As with any capital investment in manufacturing, cost is always a consideration when designing a Machine Vision system. Recurrent costs are also important but, if they are properly designed, many vision systems require little maintenance, apart from occasional testing, recalibration and routine cleaning of the lamps and optics. Many systems are installed primarily to reduce operating costs. A pay-back period of 2 years is often taken to be the absolute limit, with 6 months much to be preferred. Even when product or process safety is the first consideration, cost is still important. This is a case study of one application where cost and product safety are both critical. Many applications require only simple processing of the digital images but they can still be very challenging, in terms of building a reliable system capable of withstanding hostile factory conditions. A vision system to gauge two types of automotive parts was developed. One of the part types is a power steering connector in which the depth, width and position of a groove and the end of the power steering line are measured. The second type of part was a crimped connector attached to a brake hose. Two diameters of the crimp and the length of the hose inserted into the connector were measured. The hardware to accomplish the gauging tasks consisted of a standard video camera, light source, image digitizer and industrial personal computer. In order to minimize hardware costs, a standard 50 mm C-mount lens and extension tube were used. By modern standards, the computer was rather slow. Nevertheless, every part can be gauged at full production rates. Indeed, the gauging rate so far exceeds the production rate, that with a video multiplexer the system can process data in real time from four measurement stations simultaneously. Any system that is fast, inexpensive, simple to operate and does the job it was designed to do is a complete success; vision engineers must never be seduced by their vanity to seek sophistication.

1.7.2.2 Chapter 27: Automated Handling of Coils with Flying Leads

There is a need to use robot-like devices to coils with loose wires. (The factory that inspired this work makes audio equipment.) Given that a high-contrast image can be obtained, probably using back-lighting (➋ Chaps. 8 and➋ 40), the image processing is easy. Simple thresholding suffices (➋ Chap. 14). However, the interpretation of the resulting binary images is far from straightforward and requires calculations that neither QT nor NeatVision can perform easily. In the example discussed here, it is necessary to employ a minimum of five cameras: one overhead and four others, viewing the sides of the coil. Using the overhead camera it is possible to decide which side view is best. The reasoning needed to make this decision is ideally suited to Prolog but not QT, MATLAB or NeatVision (➋ Chap. 23). The overhead view can be processed to provide coordinates in two axes, [X,Y], in the horizontal plane. A side view can be used to derive one horizontal (X or Y) and the one vertical coordinate (Z). By combining overhead and side views, it is therefore possible to locate a point, such as the end of a wire, in 3-dimensional space. Locating another point close to the end of the wire allows its orientation to be calculated. These two pairs of position parameters are sufficient to guide a robot to grasp the wires. The ambiguity caused by the fact that there are two loose wires can be resolved easily by reasoning in Prolog. The matrix calculations needed to perform the coordinate-axis transformations are well within the capability of MATLAB (QT's host language) or JAVA (NeatVision). Of greater concern is the problem of resolving real or apparently inter-twined wires. The interpretation of pairs of images is again suitable for Prolog. In this application, where there is a high degree of uncertainty due to the flexible nature of the wires, the image processing is trivial but the use of Artificial Intelligence techniques is essential.

1.7.2.3 Chapter 28: A Telecentric Vision System for Broach Verification

Missing machining steps during the manufacture of industrial parts often leads eventually to product failure. Hence, a common requirement in Machine Vision is to build a system that can verify that a product has been properly fabricated or assembled. This chapter describes such a vision system that is able verify that a certain type of internal gear has been processed (broached) properly. With a conventional lens, the large depth of the gear results in a complex image from which it is difficult to extract the desired edge contour. However, a telecentric lens (➲ Chap. 6) greatly simplifies the image, since only those light rays that are nearly parallel to the optical axis of the lens contribute to forming the image. Once a high-contrast image has been created, the gear silhouette can be isolated, by automated thresholding. This makes the system less sensitive to variations in light level than it would be with fixed-level thresholding. This topic is considered in more detail in ➲ Chap. 13 and occurs in many other places in this book. In ➲ Chap. 19, morphology was used in an algorithm for counting the teeth of a gear (➲ Chap. 21 presents a completely different algorithm, in the form of a QT command sequence, for this task.). The simple structuring element used could be implemented using the SKIPSM approach outlined in ➲ Chap. 20. The three most significant features which contribute to the reliability and good performance of the system are the telecentric optics, automated thresholding and morphology. Following a direct design path, towards unadorned simplicity of the target system, was the key to success.

1.7.2.4 Chapter 29: Challenges of Low Angle Metal Surface (Crosshead) Inspection

It was made clear earlier that Machine Vision systems often combine several different technologies: handling of the objects to be inspected, image acquisition hardware, lighting optics, optics, video cameras, image capture hardware, digital processing hardware, image analysis software and user interface. In this application study, a system was built for on-line inspection of a diesel engine component (a crosshead) that relies for its success on the harmonious integration of all of these. The system acquires and examines two images of each part. Both images are captured when a part-in-place signal is received from the handling system. The surface inspection camera and light source are positioned at opposed low angles relative to the surface (➲ Chaps. 8 and ➲ 40). One camera is set to detect surface irregularities, which manifest themselves as shadows on the image of the top surface. These shadows are detected, measured and compared to user specifications. The other camera is placed so that it can detect the presence of foreign bodies in cavities on the lower part of the crosshead. The image contrast is enhanced with circularly polarized lighting and imaging (➲ Chaps. 8 and ➲ 40). A graphical user interface provides easy setup and debugging of the image processing algorithms. A database module collects, archives, and presents inspection statistics to the user. The inspection rate is 60 parts per minute. An automotive parts manufacturer currently uses the system for on-line inspection of four types of crossheads. As a result of this study, the customer wishes to enhance the system by inspecting the surfaces of the cavities. The important features of this study are: careful system integration, lighting-viewing conditions, collection analysis and display of performance statistics, user interface, user acceptance and processing speed (see Chap. 25).

1.7.2.5 Chapter 30: A Machine Vision System for Quality Grading of Painted Slates

Automated surface inspection is a major application area for Machine Vision systems; sheet metal, paper, plastic film, carpet, fabric, DVDs, magnetic recording media, floor and wall tiles are some of the products that have benefited so far. There are numerous other, less obvious applications, where sheet material is cut up, and assembled into products such as electrical connectors, industrial face masks, medical dressings, cakes, biscuits (cookies), clothing, etc. Any product component made by rolling or extrusion might benefit from similar surface inspection techniques. The algorithms needed are usually different from those employed when examining discrete piece parts. Surface inspection is one of the types of application described in Chap. 2 and, as a result, relies on rule-based decision-making (Chap. 23), or Pattern Recognition methods (Chap. 24). Morphology (Chaps. 19 and 20) is also likely to be important for feature identification. Chapter 30 describes an industrial inspection system that is able to identify visual defects on the surfaces of painted slates. The design of a real-time automated slate inspection system is a challenging task. The surface is coated with dark glossy paint and has an undulating form. Material is transported through the inspection area via high-speed conveyors. A customised lighting-viewing arrangement was devised and flexible image-processing procedures written to accommodate the large variety of visual defects that can appear on the slate surface. Vibration generated by the slate-transport system had to be accommodated. The final target system was subjected to a robustness evaluation, aimed at building the customer confidence needed to supplant the manual inspection procedure previously used.

1.7.2.6 Chapters 31–35: Inspecting Glassware

Glassware inspection is, in many ways, an ideal application for demonstrating the philosophy and practice of Machine Vision but the problem has not been fully solved yet. The fact that glass is transparent and has a refractive index that is significantly different from that of air makes the inspection of glassware very difficult, unless we can devise an effective lighting-viewing method (Chaps. 8 and 40). It is virtually impossible to see all of the features of interest on a bottle, or jar properly, without adjusting the lighting at some point. Understanding the nature of light transmission through a glass bottle requires a good level of understanding of optics (Chaps. 5–7) It may even be necessary to simulate the behaviour of a bottle as an optical component by computer simulation. This approach was useful for understanding x-ray images (Chap. 12). Moulded glassware is important commercially, as it is used widely, for packaging food, drinks, pharmaceuticals, cosmetics and toiletries, as well as domestic, scientific and industrial products. It has many attractive properties but two major disadvantages: it is brittle and the edges of broken glass are very sharp and dangerous. Careful inspection is therefore of crucial importance. Human inspectors cannot cope for long with the high production speeds of a modern manufacturing plant. These factors all contribute to motivate the automation of glassware inspection. This was one of the earliest areas of application for Machine Vision systems to be studied. Yet, there is no general or analytical solution; every problem has to be considered individually and analysed experimentally. The factory environment places limitations on what is possible. When they emerge from the mould, bottles and jars are still red hot, emitting a considerable amount of infra-red radiation. There are two choices here: either use the IR self-emission for imaging (➲ Appendix I), or remove it by filtering (➲ Chap. 5) and apply light externally. The camera and its lens must be protected from the fumes and sprays that pervade the factory atmosphere near the moulding machines (➲ Chap. 40). The bottles are transported from the moulding machines, through the annealing tunnel, to final packing along a conveyor belt. This suggests the use of a line-scan camera. Certain critical faults can occur anywhere around the bottle/jar; they have to be rotated to make sure these defects are always detected, whatever the orientation. High quality (i.e., close tolerance) glass products are made for electronic display devices, where they often form the substrate for a complex pattern of thin film electrical conductors. Hence, glass inspection presents not just one "big problem" but a myriad of tasks, each of limited scope. Sometimes quirky solutions are appropriate but serve only a very limited function. In this type of situation, a well equipped design laboratory, with a wide range of lighting equipment, cameras and lenses is essential. It should also possess an interactive image processing facility, to assist in the design and testing of inspection algorithms (➲ Chaps. 21, ➲ 22 and ➲ 41). The algorithms needed for bottle inspection vary in complexity from simple thresholding, for shape analysis (➲ Chap. 14) to morphology (➲ Chaps. 19 and ➲ 20) and may involve Pattern Recognition (➲ Chap. 24) or Artificial Intelligence (➲ Chap. 23). ➲ Chapters 31–35 describe quite different inspection applications involving glass. ➲ Chapters 31 and ➲ 32 demonstrate some convergence of ideas for inspecting moulded glass products. (The authors developed their ideas independently.) In view of their ubiquitous use as containers for food, pharameuticals, toiletries, etc. the safety of bottles, jars, vials, etc. is of paramount importance. Without apology, we repeat that human beings cannot perform repetetive high-speed, inspection reliably, so any improvement through automation is very welcome (➲ *Fig. 1.1*). ➲ Chapter 33 is concerned with analysing how glass panels shatter. The shape and size of the broken glass fragments is an important safety issue, particularly for automobiles. ➲ Chapters 34 and ➲ 35 are concerned with close tolerance glass artifacts. Here, high quality (i.e., close tolerance) glassware is essential for its proper function.

1.7.2.7 Chapter 36: Inspecting Food Products

Food products are usually more variable than engineering components but they are not totally unconstrained (➲ Chap. 2). For example, the height of "good" loaves made on a given production line is unlikely to vary by more than a few percent around the nominal value. Indeed, manufacturers of such products as loaves, cakes, pies, confectionary bars and pizzas are obliged by law to state minimum limits on weight, which in turn defines minimum size parameters. The upper limit is set by the container/wrapping. Aesthetic criteria, economics and customer attitudes may define even tighter restrictions. The application of Machine Vision to many food products is therefore still practical and economically viable. There is, of course, an ever-present need to improve the hygiene, safety and quality of all materials and products that we consume. Stored bulk food materials are prone to attracting insects, birds and rodents, as do growing plants. Infestation is a major issue in food processing and detecting body parts, droppings, nesting materials, etc. in food products or materials is of critical importance. In addition, manufacturers always want their products to appeal visually to the customer. As the sensory medium for Machine Vision, VIS, UV, IR and x-rays, are all valuable and have different capabilities (Chaps. 7–12 and Appendix I). A variety of inspection methods is appropriate for food products, including morphology (Chaps. 19 and 20), Hough Transform (Chap. 18), Prolog-style declarative programming (Chap. 23), colour recognition (Chap. 16) and Pattern Recognition (Chap. 24). Often, naive image analysis techniques will suffice. For example, when inspecting biscuits (cookies), the area of an object silhouette, or a measure of its resemblance to an ellipse, circle, or square, provides a basic level of discrimination. Where greater discerning power is needed, standard image processing methods, such as those described in Chap. 14 and implemented in NeatVision (Chap. 22) and QT (Chap. 21), are useful. Although Chap. 36 does not dwell on practical issues, such as the hygiene requirements for the materials/object transport mechanism, these are essential for a reliable inspection system (Chaps. 12 and 28). Many food products, such as biscuits, are made at very high speed. Achieving the throughput rate needed in a practical inspection system can be a challenge. Fast dedicated hardware is expensive but sometimes unavoidable. The ability to reformulate algorithms to maximise the processing speed in a conventional computer requires ingenuity. Chapters 17 and 20 hint at the way this might be done.

1.7.2.8 Chapter 37: Automatic Produce Grading System

This chapter shows that the technology devised for inspecting industrial products can also be applied successfully to measuring natural materials, in this case asparagus. It is emphasised in Chap. 2 that organic products, such as fruit, and vegetables are usually highly variable, compared to industrial components. Nevertheless, there is an extensive range of potential applications for Machine Vision, in agriculture, horticulture and food processing, particularly where the level of variation is restrained. This can occur naturally, for example as a result of the (statistical) limits of normal plant growth, or artificially, by processes such as trimming before inspection. In any application involving highly variable and/or flexible materials, one of the more difficult issues to resolve is the design of the mechanical handling sub-system. Apart from the obvious requirement that the object under examination has to be held/moved so that it can be illuminated and viewed properly, other demands exist: (a) The inspection system must not be damaged by the product. (For example, the machine must not jam, corrode, or interact chemically with juices exuded by the product.) (b) The machine must not damage the product. (c) The machine must not allow the accumulation of waste that will nurture the growth of micro-organisms. (d) For reasons of hygiene, it must be possible to dismantle and clean a food-handling system in a few minutes. This chapter emphasises that these are not minor issues, simply to be "tidied up," after developing clever image processing algorithms; they are absolutely central to good design. (The same point is made in Chap. 11.) An important feature of Chap. 37 is the use of vision to estimate bulk product properties, in this case volume and hence weight. (X-ray inspection systems provide an alternative approach. See Chap. 12). In this application, it is important to measure and understand the statistics of system performance, since the vision system is being used to estimate bulk properties, rather than delivering parameters for each item individually. While precision measurement of individual objects, is always welcome, it is important that the machine must not bias the statistical distribution of results. At the very least, the mean and variance for each parameter, such as length, diameter and volume, must be known. Better still, the statistical distribution of each measurement should be known. This requires that the system be calibrated carefully and regularly. In many large-volume, and/or high unit-value applications, the cost of installing a vision system can often be justified on economic grounds alone, even if it only achieves a seemingly modest reduction of waste. Significant savings can often be made simply by obtaining more precise and accurate measurements. A small percentage saving on waste, either by better discrimination of defects, or controlling trimming more accurately, can sometimes lead to large financial savings. In some instances, the case for using vision can be quite compelling economically, even if the system does not perform in a way that excites the most sedulous vision engineer. We must never assume without question that precise measurements can be computed at the same time as the product is being examined for damage, rotting, infection, etc; it is very likely that different lighting-viewing methods will be needed. While the inspection of low-variability natural products and semi-controlled manufactured goods, such as loaves, cakes, pies, etc. seems to lie at the fringe of Machine Vision as we describe in the pages of this book, this is clearly an area of major social and economic importance. The most important lesson of ➲ Chap. 37 is that it leaves no doubt that success in Machine Vision requires our adopting a systems approach during the design process.

1.7.2.9 Chapter 38: Analysing the Creasing Properties of Fabric

A large clothing retail store in UK wishes to analyse the creasing properties of the fabrics used in products, such as shirts, blouses, dresses, etc. At the moment, creasing is judged subjectively and inaccurately, by comparing a sample with carefully designed models, photographs, etc. There is a need for a laboratory, rather than a factory, based vision system to automate this task. A sample of the plain unprinted fabric is first subjected to a standard creasing regime. It is then stretched gently and placed on a table. Lighting at a low angle is then applied from one side (➲ Chap. 40). The resulting image has a low contrast but it is sufficient to be able to see the creases. ➲ Chapter 38 presents the results of applying several different (grey-scale) image filters plus thresholding, to produce a binary image (➲ Chap. 14). Several measurements are then taken to describe the texture of this image (➲ Chap. 19). The result is a multi-element vector that can then be used as the input to a pattern classifier (➲ Chap. 24). If necessary, this process can be repeated by applying more than one grey-scale filter. Their pattern vectors are then concatenated. The "teacher" required to train the pattern classifier is the existing human inspector who defines the creasing coefficient. The classifier then learns to predict this coefficient. This is the normal mode for learning by example in a Pattern Recognition system. Similar techniques are suitable for analysing other textured surfaces, such as machined metal, paintwork, decorative marble, wood etc.

1.7.2.10 Chapter 39: Environmental, Social and Ethical Issues

Machine Vision technology has changed considerably over the last 35 years. There have been major improvements in lighting, optics, cameras, camera-computer interfaces, software and hardware for processing images. However, the most significant advances have been in mental attitudes to the way vision systems are used. In the early days of the development of the subject, there was a great deal of scepticism. Nowadays, most people are aware of the possibility of adjusting pictures with a computer, using programs such as Adobe Photoshop™, while visual effects, such as posterising, pseudo-colouring and morphing, are used extensively on television and film. The concept of image processing is familiar now, even though the techniques for doing this are not widely understood. This has had an impact on the willingness of industrial engineers to countenance the use of Machine Vision. However, there is a danger of developing unrealistic, over-optimistic expectations, created by science fiction drama. Vision engineers have also learned a lot since the inception of the subject, particularly about appropriate attitudes and priorities. The following points have been learned, often forgotten and then rediscovered:

- Systems thinking is of paramount importance (➲ Appendices B–D)
- Do not expect to rely on formulae or a list of prescribed steps. (The author has often contemplating trying to condense his knowledge into an expert system but still remains convinced that it is far too difficult. He began working on this book instead!)
- Consider the application requirements (➲ Appendix B)
- Decide exactly what the vision system is supposed to do and do not be deflected from that
- Visit the factory
- Gather plenty of fresh samples of objects to be inspected
- Do not expect the customer to understand or trust the technology, at first (➲ Appendix C)
- Do not expect a vision system to be able to work in uncontrolled natural lighting (➲ Chaps. 8 and ➲ 40).
- Look at the object to be inspected. (This simple advice is often ignored.)
- Everybody is an expert on vision – in his own estimation but not in reality.
- Ask questions (➲ Appendix B contains some typical questions but do not over-rely on this.)
- Select an appropriate camera or scanner (➲ Chaps. 8–10)
- Experiment with the lighting-viewing conditions (➲ Chaps. 8 and ➲ 40)
- Design the optics scientifically (➲ Chap. 5)
- Experiment with the image processing (NeatVision and QT typify the type of software needed. ➲ Chapters 21, ➲ 22, ➲ 41, ➲ Appendix G)
- Demonstrate feasibility at each stage in the design. (This is what QT and NeatVision are intended to do)
- Protect the lights, camera and optics (➲ Chap. 41).
- Remember and apply Occam's Razor: *It is vain to do with more what can be done with less.* (The KISS principle is a more direct way to express the same idea: *Keep it simple, Stupid!*)

➲ Chapter 39 also briefly considers some of the social and ethical issues relating to Machine Vision. This is an area where more serious work is needed. Like all modern technologies, Machine Vision can be used for good or ill. Its power to benefit society lies in its ability to free human beings from tedious boring jobs that people cannot do very well.

The views expressed in ➲ Chap. 39 are those of the author and do not necessarily reflect those of other contributors to this book. Of all the many lectures on Machine Vision that the author has given in his career, the points made in this chapter elicited by far the most enthusiastic response. Even if the reader does not agree, the author hopes that it will trigger discussion and more thought about the important non-technical issues.

References

1. Batchelor BG (1991) Intelligent image processing in Prolog. Springer, Berlin. ISBN 0-540-19647-1 & 978-3-540-19647-1
2. Batchelor BG, Waltz FM (1993) Interactive image processing for machine vision. Springer, New York. ISBN 3-540-19814-8 & 978-3-540-19814-7

3. Batchelor BG, Waltz FM (2001) Intelligent machine vision. Springer, New York. ISBN 3-540-76224-8 & 978-3-540-76224-9
4. Batchelor BG, Whelan PF (eds) (1994) Industrial vision systems, vol 97, SPIE Milestone Series. SPIE – The International Society for Optical Engineering, Bellingham. ISBN 0-8194-1580-4
5. Batchelor BG, Whelan PF (1997) Intelligent vision systems for industry. Springer, London and Berlin. ISBN 3 540 19969 1 & 978-3-540-19969-4
6. Batchelo BG, Hill DA, Hodgson DC (eds) (1985) Automated visual inspection. IFS Publications Ltd, Bedford (Also published by North Holland, Amsterdam, 1985, ISBN 0-44487577-98). ISBN 0-903608-68-5

2 Inspecting Natural and Other Variable Objects

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_2,

Abstract: This chapter is concerned with the design of artificial vision systems that attempt to emulate one particular human activity: inspecting and manipulating highly variable natural objects and human artifacts. During the last quarter of the twentieth century and the first decade of the twenty-first century, *Machine Vision*, evolved from an exotic technology, created by academics, into one that is of considerable practical and commercial value, and which now provides assistance over a wide area of manufacturing industry. Hitherto, Machine Vision has been applied extensively in industry to tasks such as inspecting close tolerance engineering artefacts, during or shortly, after, manufacture. However, Machine Vision technology can also be applied to those areas of manufacturing where natural materials and other highly variable objects are processed. Our discussion in these pages is aimed at those areas of manufacturing where wide tolerances are encountered, as well as activities, such as agriculture, horticulture, fishing, mining, etc., where similar conditions apply. However, we should not lose sight of the fact that certain so-called high-precision industries are plagued by significant degrees of uncertainty. For example, the electronics industry is renowned for working with high-precision components (integrated circuits, printed circuit boards, etc.). However, it is also concerned with highly variable entities such as solder joints, flexible cables and components with flying-lead connectors (resistors, capacitors, diodes, coils, etc.). This chapter is relevant to applications such as these, as it is to examining fruit, vegetables, animals, fish, etc.

2.1 Introduction

2.1.1 The Problem Domain

We shall concentrate primarily on the inspection of natural materials and products with an ill-defined shape, size, colour or texture. Food manufacturing forms an important area of application but there are applications of this kind in a wide range of industries, including agriculture, fishing, horticulture, mining, catering, pharmaceuticals, clothing, foot-wear, etc. In addition, many tasks in engineering manufacture require the handling and processing of highly variable objects.

Both Machine Vision and Robotics have hitherto achieved their greatest success with close-tolerance products. By broadening the scope of our subject, we are implicitly acknowledging that Machine Vision has reached a certain level of maturity and that it is able to reach into application areas that have hitherto been considered to be too difficult. There are significant commercial and practical benefits to be obtained. For example, there is a strong and continuing demand for new instruments that can improve the quality and safety of all products, especially food and pharmaceuticals items. If we are successful, Machine Vision will bring significant economic, health, safety and social benefits to several areas of industry that have hitherto been inaccessible to it.

When we apply Machine Vision techniques to the inspection and handling of manufactured goods, we know roughly what to expect. This is certainly true for high-precision objects, such as stampings, castings, mouldings, lathe-turned objects, etc. It is also true for many types of mass-produced food items. For example, one loaf of bread, confectionary bar, meat pie or cake, of a given kind, should be "similar to" all other items of the same nominal type. Even live-stock exhibits some similarities but these may be harder to define objectively. Most pigs of a certain breed will have "similar" colouring patterns. More generally, natural products of a given class usually possess some degree of "similarity" to each other. Indeed, this helps us to define the concept of a "class". More recently, Machine Vision, and the allied subjects of Pattern Recognition and Artificial Intelligence, have all developed to the point where we can broaden the concept of "similarity".

2.1.2 Applications Classified by Task

The potential for applying Machine Vision technology in food industry, agriculture and horticulture is huge, although the penetration of these markets to date has been somewhat less than in manufacturing. Present and projected applications of Machine Vision to natural products may be classified conveniently according to the function they perform. No particular commercial importance is implied by the order of the following list:

- *Analysing the content* of a mixture (e.g., count how many sweets of each kind are put in a box; estimate the lean-to-fat ratio of a batch of butchered meat; check the mixing of granulated materials)
- *Analysing the shape* of whole products as a prelude to processing them using robotic manipulators (e.g., fruit, vegetables, animal carcasses, etc., might be trimmed and packed by a visually guided robot)
- *Analysing texture* (e.g., bread, cork floor tiles, wood panels, etc.)
- *Assembling food products* (e.g., pizza, quiche, meat pies, etc.)
- *Checking aesthetic appearance* (e.g., loaves, cakes, quiches, trifles, wood veneer, marble tiles, etc.)
- *Cleaning* (e.g., selective washing of cauliflower, broccoli, leeks, lettuce, cabbage, etc.)
- *Coating* (e.g., chocolate coating of confectionary bars, icing (frosting) on cakes)
- *Counting* (e.g., counting cherries on the surface of a cake, counting bubbles on top of a trifle, estimating the number of healthy leaves on a growing plant)
- *Decorating* (e.g., cakes, chocolates, trifles, pies, etc.)
- *Detecting foreign bodies* (These may be “natural” and intrinsic to the product (e.g., seeds, nut shells, twigs, etc.), or truly “foreign” (e.g., stones, rodent remains, broken machine parts, contact lenses, false teeth, etc.))
- *Detecting surface contamination* (e.g., mildew, mud, bird excrement, etc.)
- *Estimating the size or volume* (e.g., fruit, fish, animal, poultry carcasses, meat fillets, etc.)
- *Grading* (e.g., identifying premier-quality fruit and vegetables)
- *Harvesting* (there is a huge variety of tasks of this type)
- *Identifying loss of integrity* (e.g., rotting of fruit or vegetables, withering due to disease or lack of moisture, bird pecks, insect holes,)
- *Measuring linear dimension* (e.g., fruit, vegetables, animal carcasses)
- *Packaging* (e.g., fragile or variable products, cream cakes, meringues, etc.)
- *Sorting* (e.g., fruit from leaves; fish by size / species on a trawler)
- *Spraying* (e.g., selective spraying of insecticides and fertiliser)
- *Stacking / packing* (e.g., carrots, oranges, frozen chickens, boxes, jars, bottles, cans or tubs, flexible bags)
- *Trimming and cutting* (Fish and vegetables all require robotic cutting. Butchery and automated shearing of sheep are also included in this category)

This does not include the multitude of tasks that can arise in manufacturing, where low-tolerance features occur, and giving rise to highly variable images to be processed. The following list gives a hint of the wide variety of inspection tasks of this kind that exists:

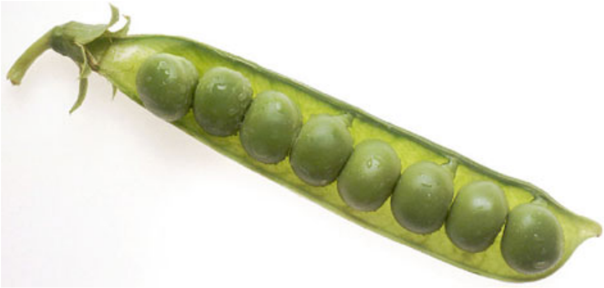
- Adhesive strings and drops
- Complex assemblies of components
- Electrical coils that have flying wires attached

- Feed stock for automatic machining
- Flexible pipes
- Gobs of semi-molten glass (prior to forming jars, bottles, etc.)
- Moulds loaded with powders, granulated or semi-fluid material
- Material-feed hoppers
- Powdered material prior to melting, sintering, etc.
- Rubber or plastic gaskets, water and oil seals, etc.
- Solder joints
- Spray cones, flames and smoke plumes
- Sprue, swarf, remnant material (for example, after punching)
- Unfired ceramics, including tiles, table-ware, etc.
- Welds
- Wiring looms

The fact is that even so-called high-precision manufacturing is not as tidy and well controlled as we would like. The main reason is that raw materials are unpredictable in shape, size, and colour. Another important factor is that it is sometimes expedient, or necessary, to apply a low-tolerance manufacturing technique (e.g., applying adhesive) to high-precision parts. The engineer's skill is manifest in doing this in such a way that the "untidy" parts are not apparent to the customer.

2.2 Product Variability

In British English it is common to say that two similar objects are "as alike as two peas in a pod". Yet, peas in a pod can be very different from one another (➲ *Fig. 2.1*). They can differ very considerably in size, shape, colour and surface texture. Some peas are likely to be bright green, while others have a brown or yellow tinge, or be nearly white. Their skins can be split and some peas have insect holes. They can also differ in shape; some peas are almost square, having been squashed by their neighbours in the pod. Most peas are nearly spherical, while others may be ovoid. Natural objects, like peas, frequently vary from one another in many different ways. In the light of these remarks, it may seem that by stating that "objects A and B are as alike as two



◘ Fig. 2.1

Peas are not alike in shape, colour or surface texture (Photograph Renee Comet, http://visualsonline.cancer.gov/details.cfm?imageid=2612, accessed 6th October 2008)

peas in a pod", we are not making a very strong statement about their similarity. In fact, we need to revise our concept of similarity, since all of the peas growing in a given pod are genetically identical; they all contain the same macromolecules and they have very nearly the same taste and nutritional value. Natural variations in physical size, colour, shape and texture all have to be *tolerated* by an inspection system. Peas are never more than 15 mm in diameter. They are never blue, striped or hairy. They do not have a sponge-like surface, nor do they move of their own volition. This is the nature of their similarity, which is fundamentally different from that relating two cars of the same model, or two bottles made in the same mould.

2.2.1 Linear Dimensions

We need to define a measure of variability. Let X be some conveniently measured linear dimension, such as the diameter of an apple, the length of a banana, or the circumference of an orange. Let X_0 be the average value of X, for a certain class of objects, C. Furthermore, let δX be the maximum observed deviation from X_0. (Hence, all estimates of X for objects in class C lie in the range $X_0 \pm \delta X$.) The degree of variation of objects in class C is defined as V_C where

$$V_C = \log_{10}(X_0/\delta X) \tag{2.1}$$

Statisticians would, no doubt, argue that we should instead relate V_C to the standard deviation of X, rather than the total range of this variable. However, the logarithm in the definition of V_C makes this change relatively unimportant. Furthermore, as far as both manufacturers and customers are concerned, the range of variation ($X_0 \pm \delta X$) is often more important than statistical measures. If an object, such as a loaf, is too large, it will not fit into its packaging. On the other hand, if it is too small, the customer will not buy it! The present measure of variability (V_C) is conceptually simpler than one based on statistical parameters and is perfectly satisfactory for our present purposes.

We shall see that V_C is large for high-tolerance (engineering) products and small for highly variable objects (e.g., food and natural products). In the following discussion, remember that almost all of the successful applications of Machine Vision to date have been based on products for which $V_C \geq 2$ (➲ *Fig. 2.2*).

Among the very highest precision objects made in industry are mirrors and other optical components. A good-quality plane mirror has a surface ground to about $\pm\lambda/10$, where λ is the working wavelength. Suppose a mirror has a diameter of 50 mm and is expected to reflect green light ($\lambda = 0.5$ μm). Then, $V_C = 5$. (It is more meaningful to compute V_C in this way than to compare to the thickness of the mirror substrate.)

Electronic circuit boards combine high precision components (semiconductor chips) and low precision features (e.g., resistor placement and solder joints). Integrated circuit chips with a conductor width of 2 m (i.e., $X \approx 0.2$ m) and a diameter of 8 mm have a variability score $V_C = 4.6$.

A "typical" engine part for an automobile engine might be 10 mm long and have a tolerance of 10 m. Then, $V_C = 3$.

A bottle or jar has a slightly greater degree of variability. A wine bottle is about 300 mm tall and its variation in height is roughly 1 mm. (Although it is made in a mould, the bottle is liable to sag under its own weight, before the glass cools to become "solid".) Hence, $V_C = 2.4$.

Now, consider more variable items. A jam doughnut is about 150 mm in diameter and its variation in diameter can be as much as 10 mm. (It is not truly round either.) Hence, $V_C = 1.18$. Baked goods, formed by cooking semi-fluid dollops of dough on a plain tray, can vary by even

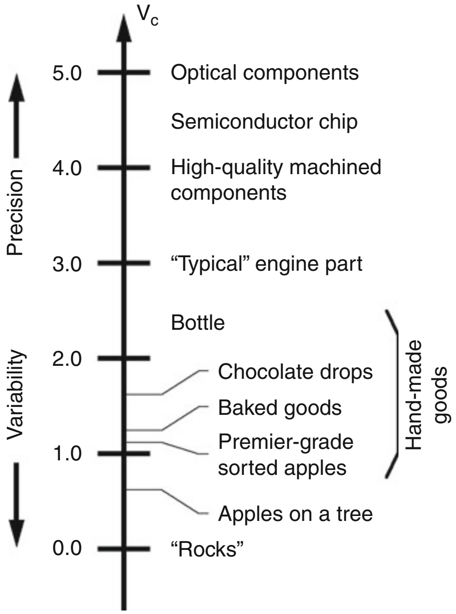


Fig. 2.2
The measurement of variability in human artefacts and natural products. All values of Vc are approximate. Hitherto, almost all applications have been based on products for which Vc > 2.0

more than this. The baking process, combined with the ill-defined size of a "dollop" can produce values of V_C less than 1.0. The author measured two packets of chocolate drops, of different sizes, yielding measurements of V_C equal to 1.54 and 1.56. Bread rolls, croissants and many other baked goods and items of confectionary are highly variable, giving values of V_C equal to 1.0 or even less.

Natural products, such as apples, of a given variety, typically vary by a large amount, giving values of V_C as small as 0.4–0.5. Modern supermarkets tend to avoid selling apples of highly variable size, by the simple expedient of grading them before they are put on sale. (Grading apples represents a potential application for Machine Vision.) In a series of measurements of premier-quality graded apples, a value of $V_C = 1.10$ was recorded.

Agriculture, as distinct from food processing, is concerned with very high degrees of variability, since growing animals and plants can differ greatly in size. A farmer cannot properly control the size of fruit on a tree; he or she must pick whatever happens to be there. Hence, V_C is likely to be very small, possibly under 0.5. Of course, farm animals also vary in size. For adult pigs of a given breed, $V_C = 0.5$.

There is another important class of objects that we have not mentioned yet but which deserves attention: hand-made goods. Consider a hand-thrown pot, such as a jug, or vase. No two vessels, even of the same nominal type are exactly alike in size and shape and we might expect V_C to be about 1.3–1.9. Hand-carved and painted artefacts produce a roughly similar range of variation, as do hand-cut and engraved glass-ware. Exceptionally, human beings can make objects for which V_C is as large as 2.3.

There is another even worse situation in mining and quarrying, where it is almost impossible to define the size of a "typical lump" of rock. Here, it is meaningless to quote typical values for our measure of variability, V_C. Let it suffice to say that gravel, lumps of rocks and coal, etc., are very variable ($V_C \leq 0.3$) and therefore require non-specific handling techniques.

2.2.2 Shape

The so-called *shape factor* is one such shape descriptor that is popular among vision engineers. The *shape factor* of a blob-like object is defined as 0.079577 (1/[4π]) times the ratio of its area to the square of its perimeter. This has a maximum value of 1.00 for a circle and has low values for spider-like objects. For example, a hair comb gave a value of 0.01532. Five ivy leaves (➋ Appendix G) produced values of shape factor varying from 0.425 to 0.546. Similarly, five chicken "butterflies" (➋ Chap. 12) gave values between 0.421 and 0.597. On the other hand, a croissant produced a very similar value (0.633) to that obtained from a slice of bread (non-lidded tin loaf, 0.631). The fact that intra-class variability is comparable to and may even exceed inter-class variability merely indicates that shape factor is not a particularly good measurement of shape. Other shape measurements may be more sensitive to the particular types of shape variation that are important in a given application.

Despite the differences that exist among objects like ivy leaves, human beings are readily able to classify them as belonging to the same general class. Of course this merely confirms that human beings are well suited to recognising natural objects, despite their highly variable shapes. This is just one example of the general principle that tasks that a person finds easy may be quite difficult for a machine (and vice versa).

When designing Machine Vision systems, we have to assume that the shapes of both natural products (e.g., fruit, vegetables, nuts, individual leaves, decorative domestic plants, etc.) and certain manufactured items (baked goods, confectionary, meat pies, joints of meat, etc.) will vary considerably. We have only to pick up a handful of fallen leaves, or buy a kilogramme of carrots or potatoes, to observe the high level of variation of shape that a vision system must tolerate. Of course, while shape variation is usually manifest in three dimensions, it is only observed in two dimensions, assuming that we employ a single video camera. A stereoscopic vision system or 3-D scanner may see the full effects of shape changes but we do not need either of these to understand the nature of the problem that shape variation causes. It must be understood that conventional (i.e., geometric) concepts of shape are totally meaningless in this context. For example, "round" fruit is not truly circular, in the mathematical sense. Furthermore, straight lines, right-angle corners, rectangles parallelograms, ellipses and other geometric figures do not exist in nature, except in the special case of crystalline structures.

The unpredictable shape of certain natural objects creates severe problems for an automated mechanical-handling system. Fixed tooling is out of the question for tasks such as trimming lettuce, shearing sheep, or automated butchery. Alternatively, multi-axis robots require complicated multi-sensor feed-back control techniques to cope with highly variable object shapes. Even seemingly simple vision-based tasks relating to natural objects can require the use of complex equipment. For example, removing spot-like surface defects on nominally round fruit or vegetables (e.g., "eyes" on potatoes) requires the use of a 4-axis (X,Y,Z,θ) visually-guided robot. Trimming rhubarb requires a 3-axis robot (X,Y,θ), while placing decorative features on top of a cake requires 4 degrees of freedom, (X,Y,Z,θ). Everyday tasks, such as removing the leaves from carrots, or trimming leaks, are often quite difficult when we try to automate them. We should bear in mind too that in many cases we must treat natural products with great care, to avoid damaging them. By clever user of guide rails, shaped conveyors, etc., it is sometimes possible to reduce the complexity of a robot handling highly variable products. For example, a single-axis robot could be used to trim the ends of carrots that have already been aligned, perhaps using a slotted conveyor. Nevertheless, the complexity of the handling equipment is usually higher than it is for high-precision objects. The general rule is that, if we know what an object is and where it is, simple handling techniques are possible, even desirable. Variations in shape make this prerequisite difficult to achieve.

Many shape-analysis techniques that work satisfactorily on one example of a given type of object may not work with other samples of the same product kind. In other applications, ill-defined shapes are present that are amenable to only the simplest and most robust methods of analysis. In some situations, we may even have to change the method of analysis as the shape changes. A child's toy illustrates this point (Appendix F). Organic materials and objects frequently change shape as they grow. Hence, we must be prepared for a similar situation to occur, when we are working with natural products.

2.2.3 Why Physical Tolerances Matter

There are several reasons why physical dimensions matter:

(a) Objects to be inspected must be fed through the inspection machine, whatever their size and shape. The mechanical handling system must accommodate all variations of these parameters without jamming. Clearly, this is much more difficult if these objects are of uncertain size and shape.

(b) In view of the preceding point, the system for mechanical handling of highly variable objects must either be very intelligent, or very simple, such as a plain-top conveyor belt. Of course, a conveyor belt is far simpler and cheaper than a multi-axis robot but has one serious drawback: it is unable to hold the object in any particular pose before the camera. The design of robust mechanical handling systems for highly variable objects, especially plants with flexible foliage, requires considerable skill and ingenuity.

(c) The camera may not be able to see certain features. Certain objects (e.g., bread rolls and some cakes) are actually made on a conveyor. It is therefore quite difficult to examine their bases. Certain objects will always fall into the same pose, if dropped haphazardly onto a flat surface (e.g., a plain conveyor belt). Other objects may have two or three stable states, while objects such as apples have one or more quasi-stable states. As a result, measurements of certain parameters, such as apple diameters, may show statistical bias, unless care is taken to orientate the objects first. The conveyor itself may obscure some features. It is common practice to use shaped (e.g., cupped) conveyors to stop delicate objects such as eggs, fruit, etc., from banging into one another. Inevitably, this causes some restriction on viewing angles.

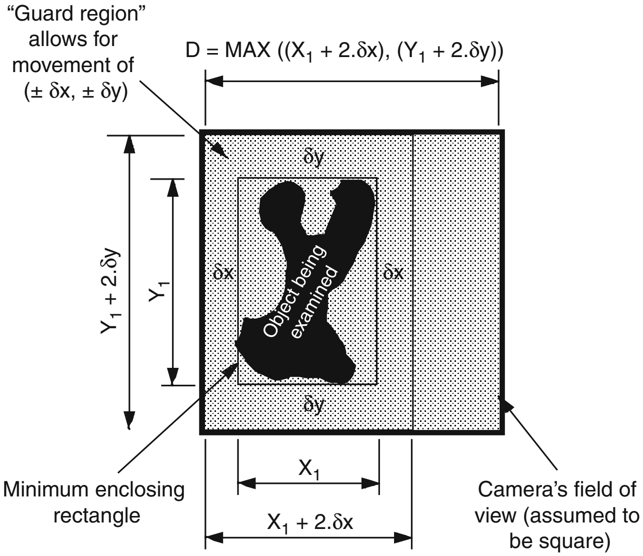
(d) Varying shape presents problems by obstructing illumination, or viewing. At its simplest, this means that certain features are obscured, either by being out of the camera's line of sight, or by being in a shadow. The latter can *always* be solved by coaxial illumination and viewing but this may not yield sufficient contrast to be of practical use. Obtaining images from the unpredictable form of untrained foliage presents a particular challenge and a high level of intelligence is needed to perform operations such as pruning, picking fruit, selective spraying, etc. In general, the lighting arrangement needed for highly variable products is problematical, since the optimal illumination configuration for one sample may be far from ideal for another. The problem is made worse by the fact that samples of the same type of object may lie in different poses. The range of options for maximising image contrast by good lighting is rather smaller than that available for well-defined, high tolerance products. The use of intelligent lighting is another possibility for coping with high levels of product variability but has, as yet, received scant attention.

(e) When object size and/or shape is highly variable, choosing the most appropriate lens can present problems. The field of view must be large enough to accommodate the biggest objects of a given class. Yet, this may mean that the smallest examples may not be seen easily. The problem is worse than we might imagine at first, because natural products tend to be delivered haphazardly for inspection. It is impossible to predict how they will fall to rest, since they roll, fall into a quasi-stable pose, or even change shape as thy come to rest (e.g., soft-heart lettuces, certain types of cut flowers). All of these produce an uncertainty of position that must be accommodated by increasing the size of the camera's field of view (➋ *Fig. 2.3*).

(f) Changes in object size alters the contrast in x-ray images. The thicker the sample, the denser an object is to x-rays and the lower the contrast on small foreign bodies of fixed size and material will be.

(g) Finally, the image processing operations needed to inspect natural and other highly variable objects (Vc $\leq$ 2) often differ fundamentally from those used for high-tolerance parts. (Vc $\geq$ 2). The former have to be more intelligent, since they have to cope with greater levels of uncertainty about all aspects of the object's appearance: size, shape, pose, position, texture and colour.

Perhaps the most significant problem of all is the lack of a clear objective criterion for what is an acceptable product. Bakers have devised heuristic rules for identifying malformed loaves


"Guard region" allows for movement of (± δx, ± δy)
D = MAX ((X1 + 2.δx), (Y1 + 2.δy))
δy
δx
δx
Object being examined
δy
Y1 + 2.δy
Y1
X1
X1 + 2.δx
Minimum enclosing rectangle
Camera's field of view (assumed to be square)

■ Fig. 2.3

Uncertainties in the position of the object being examined require that the camera's field of view be larger than the object itself. If the object orientation is not controlled, the field of view must be even larger. (Also see ➋ Chap. 40 [Methods 8 and 9] and ➋ Chap. 41 [QT function fov].)

or cakes. These are used principally to train inspection personnel and were derived by bakers, trying to predict what their customers want. The best practice is, of course, to ask a large number of customers what they will accept. Although this is expensive, it does not necessarily give a perfect picture of what constitutes a "good" product, since customers cannot always vocalise their preferences. Precision in such an exercise is notoriously difficult to achieve, because there is no such thing as a "typical" customer. Judgements are made sub-consciously, based upon a large variety of factors, such as packaging, cost, and even location on a supermarket shelf. In some companies, opinion about what is acceptable may even come from casual remarks made by the CEO's wife! However, the final decision about what constitutes an acceptable loaf or cake is based on purely subjective criteria by individual customers in a supermarket. Strict rules do exist for grading and classifying fruit and vegetables. (Indeed, these are fundamental to the international trade agreements.) These include but are not restricted to specification of shape, size and colour. Criteria for grading living plants, and animals are almost inevitably expressed by heuristic rules, derived by nursery-men, farmers, whole-salers, retailers, etc., who try to anticipate what the end-customer wants. Aesthetic judgement about decorative items, such as wood panelling and marble tiles is difficult to automate but criteria based on experience probably represent the best explicit statements that we can achieve. To a software engineer, this suggests the use rule-based decision-making techniques and expert systems. Self-adaptive Pattern Recognition techniques, including Artificial Neural Networks, are appropriate when a human inspector can only say that an object is either "good" or "faulty".

2.2.4 Flexible and Articulated Objects

Many natural and fabricated products are totally or partially flexible and present considerable problems for the designer of a vision system. The mechanical handling sub-system and image processing algorithms have to be designed with particular care.

Tasks involving the manipulation of simple articulated objects may be far more difficult than we might imagine. Even a simple pair of scissors alters its topology as it is opened and shut (Appendix F). A chain of linked levers is even more difficult to inspect and may require that we flex the assembly to examine it properly. This may well require sophisticated sensing (vision and force feed-back) and manipulation (robotics), as well as a high-level of intelligence and knowledge about the application. The posture of an articulated object may become unstable, causing it to fall over, as is it (un)folded. Certain features may become invisible, for example when an articulated arm is folded. Flexible objects and articulated mechanical systems are often fragile and therefore require special care in handling. Non-contact opto-electronic sensing is, of course, ideal, provided that the inherent difficulties of working with flexible or articulated systems can be overcome. Unfortunately, we are not yet able to build an effective "multi-purpose" handling system for flexible and articulated objects; each application has to be considered separately.

It is fairly easy to design a mechanism to transport carrots, once their leaves have been removed but they do not oblige us by growing according to our whims! Vegetables and fruit may well be delivered to an inspection system with their leaves and stalks still attached. Of course, it is likely to make the design task easier for the vision engineer, if they have been removed beforehand. Many types of fruit and vegetable can be handled successfully by cleverly-designed mechanical systems but these are expensive, prone to jamming and damaging the product. In addition, food-trimming machines require frequent cleaning and may be quite wasteful by removing too much "good" material. It would be better, therefore, if we could use vision to minimise the physical contact between the machine and the product that it is processing.

Gardening and horticulture present some interesting but very challenging applications, including pruning, grafting, taking cuttings, replanting seedlings, etc. These are difficult tasks for a machine and consequently require intelligent image analysis techniques and careful design of the robot and its end-effectors. Micropropagation is a well-constrained task of this type and offers immediate financial and social benefits. (Human beings have to wear face masks, hair nets and overalls to avoid infecting the tiny plants, as they are working.)

Broadly similar problems occur in engineering industry, for example, when handling objects that have trailing tubes, wires or cables (➲ Chap. 27). Industrial parts-handling applications create a wide spectrum of difficulties.

Some objects such as certain cuts of meat and fish, foliage, leather, cloth, etc., are, of course, flexible. Sometimes a vision system is required specifically to guide a robot to handle and process such materials. This type of task still presents a formidable challenge to both vision engineers and robot designers. Hence, on many occasions, we still have to rely on human beings to deliver flexible materials to an inspection station. (The chicken-breast "butterflies" illustrated in ➲ Chap. 12 are fed to the x-ray system in this way.)

Once again, it may be impossible to achieve optimal lighting everywhere. At first sight, structured lighting (➲ *Fig. 2.4*) seems to provides a possible way to acquire detailed information about the 3-dimensional structure for certain applications. However, there are considerable problems, due principally to glinting, occlusion, and the limited working speed. of light-stripe triangulation techniques.

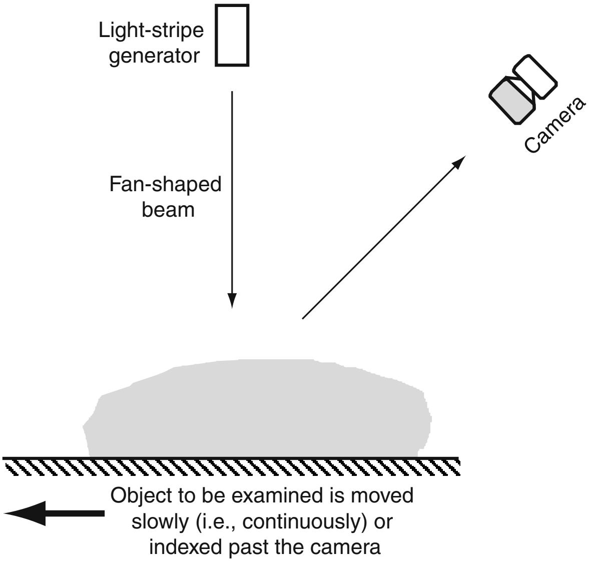

Light-stripe generator
Camera
Fan-shaped beam
Object to be examined is moved slowly (i.e., continuously) or indexed past the camera

Fig. 2.4
Structured lighting to measure 3-d shape. The light-stripe generator may be as simple as a diode laser fitted with a cylindrical lens. If the positions of the laser and camera are reversed, the data is more difficult to analyse

2.2.5 Soft and Semi-Fluid Objects

Machine Vision has an important role to play in inspecting soft and semi-fluid materials, such as uncooked dough, pastry and cake mixture, minced meat, whipped cream, jam, various gels and pastes used in the food and pharmaceutical industries, etc. Many industrial artefacts contain materials that are semi-fluid at some time during the manufacturing process. Examples are uncured foams, molten solder, globs of adhesive, gobs of molten glass, etc. These all require special handling techniques, as we must not disturb a semi-fluid dollop unnecessarily, simply to suit the whim of a vision engineer.

A characteristic that is peculiar to this class of materials is that their composition can be changed very easily and quickly. To avoid drying and curing semi-fluid material, infra-red and ultra-violet should be removed from the illuminating radiation, using standard optical filters. Sometimes in particularly dirty factory environments, an air purge is used to maintain clean optical surfaces. In this situation, we must ensure that it does not blow dust or moisture onto the damp surface. An ill-directed air purge can cool the surface too quickly and blow dirt onto it.

However, if care is taken to avoid the problems just mentioned, Machine Vision is ideal for inspecting semi-fluid materials, as no physical contact is needed. Thus, the shape of a lump of unbaked dough, or a dollop of whipped cream, is not altered by the inspection system. Furthermore, there is no possibility of introducing micro-organism contamination to food or pharmaceutical products. These facts provide powerful motivation for using Machine Vision in these industries. However, there is one fundamental problem: *What does a "good" lump of dough look like?* How do we define an acceptable range of shapes for dollops of cream on a cake or trifle? This the classical "ground truth" problem encountered when designing Pattern Recognition systems.

2.2.6 Colour Variations

2.2.6.1 Colour Variations in Natural Products

One of the greatest challenges encountered when applying Machine Vision to natural objects lies in the recognition of colour. Natural objects tend to display very subtle variations of colour, which usually changes continuously across an object's surface. While sharp colour gradients sometimes occur in nature, they are usually less common and less pronounced than those found in industrial products with highly variable coloration. Natural objects of the same nominal type may differ from one another, with significant changes in colour. On the other hand, damage or disease in plant or animal material may be manifest by very subtle colour changes. Sometimes infra-red and ultra-violet illumination help to improve image contrast; IR is particularly sensitive to detecting bruising in fruit.

2.2.6.2 Colour Variations in Manufactured Products

Subtle colour changes occur on the surfaces of many industrial artefacts as a result of processes such as drying, oxidation, rust formation, chemical corrosion, micro-organism attack, etc. Detecting these conditions can be just as difficult for a Machine Vision system as any task involving natural products. Highly sensitive sensors have been developed for industries such as textiles and automobiles, where high colour fidelity is essential. However, these usually measure the colour either at a single point, or averaged over a region. To date, the use of a video camera for monitoring subtle colour variations across a large area, such as a complete car body, has not been very successful, to date.

2.2.6.3 Pseudo-Natural Colour Recognition Task

Many engineering artefacts are either monochrome, or polychrome with colours that are distinct in both space and hue. The most notable exception to this is to be found in high-fidelity colour printing, which attempts to emulate natural colouration. For this reason, we should regard colour printing as a "natural" scene, although two important difference exist:

- We know very precisely what colours to expect at each point in a printed colour image. In principle, this allows direct, point-by-point comparison to be made between two samples of colour printing. However, this approach is not possible as a basis for inspecting true natural products.
- Colour printing uses very few pigments, perhaps only three or four, to approximate continuous colour variations. It is necessary therefore to view the printing at an appropriate resolution; if we wish to see individual dots of coloured ink, we need a very high resolution. To view printing in approximately the same way that we do, we must use a much lower resolution to obtain a degree of spatial integration.

2.2.6.4 Traditional Colour Theory

A common practice in books and papers on Colour Theory is to annotate the standard CIE Chromaticity Diagram by placing a spot to indicate the colour of, say, a ripe tomato, lemon or lettuce. Unfortunately, this excessively naïve view is unhelpful for our purposes and can be misleading. In reality, a ripe tomato should be represented by a large diffuse cluster, not a single point, as there is likely to be a considerable variation in colour across its surface. In theory, the ripening process of a tomato could be mapped onto the Chromaticity Diagram by drawing a thick arc, starting at green, travelling through yellow, to orange and then red (➲ *Fig. 2.5*). However, the standard treatment of colour is inadequate to represent the subtlety of natural phenomena, such as tomatoes ripening. The difficulty lies not, so much in the representation of colour, as the analysis of the data. No general theoretical model yet exists. Instead colour recognition can be approached using Pattern Recognition learning methods (➲ Chap. 24), or the Programmable Colour Filter described in ➲ Chap. 16.

2.2.6.5 Problems in the Naming of Colours

The core of the colour recognition problem lies in the fact that we do not know what a label such as "yellow" really means. Of course, you "know" what "yellow" is, because you were taught from an early age to associate that word with a certain class of visual stimuli. Since then, you have had your mental concept of "yellow" fine-tuned by years of exposure to other people, even though each has a slightly different idea. The people who taught you what "yellow" means did not teach me, so my concept of "yellow" is slightly different from yours. Proving this is easy: simply ask

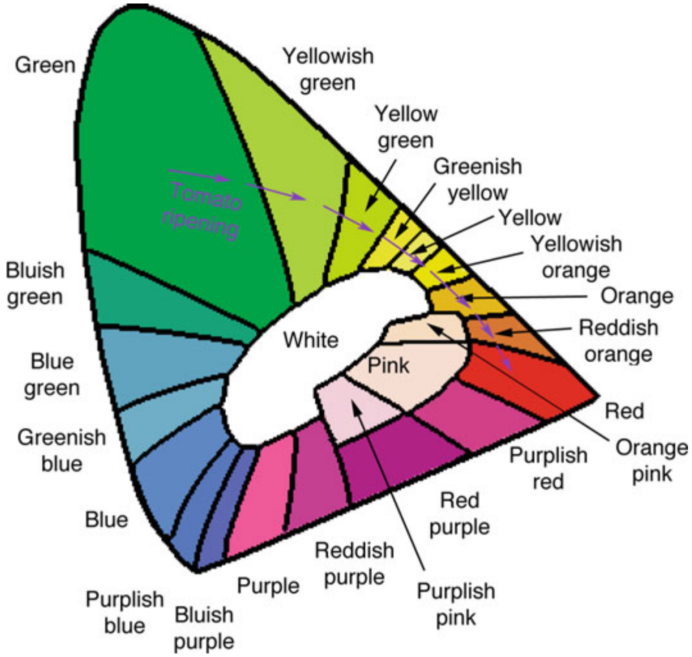


Fig. 2.5
To reduce the cost of printing, the CIE Chromaticity Diagram is often published in monochrome form, with labelled "cells" added as an aid to understanding. Sometimes each "cell" is assigned a uniform colour, as shown here, but this merely illustrates what labels such as "reddish orange", "red", "purplish red", etc., typically represent. These "cells" create the false impression that it is possible to discriminate accurately between colours; they do not provide the basis for universally agreed colour identification. Furthermore, the diagram makes no contribution towards the recognition of commercially important colour categories, such as "butter yellow", "leaf green", or "well cooked cake". Notice that colour labels apply to sets not points on this diagram. Hence, the process of ripening in a tomato should be represented diagrammatically by a broad band, not a single point locus

a few people to draw limits of what they perceive to be "yellow" on the spectrum, or any other rainbow-like pattern. (➋ Chap. 16 presents the results of one simple study of this kind.) Why are there such wide differences in the definitions of colours? We all agree on certain "core" colours being given names such as "yellow", "red", "turquoise", etc. However, we differ in our naming of those colours at the boundaries between these classes. There are certain reasons for this:

(a) Different languages name colours in different ways. ➋ *Figure 2.6* explains the differences in the naming of certain colours in Welsh and English. (These two languages have been juxtaposed for over a thousand years.) There are even significant differences in the use of colour names between Welsh speakers from North and South Wales. The English word "yellow" cannot be translated exactly into French. It is less surprising that major differences exist between English and Japanese, as well as between English and Chinese.

(b) Certain diseases, brain injury, ageing and drugs can all cause changes in colour perception.

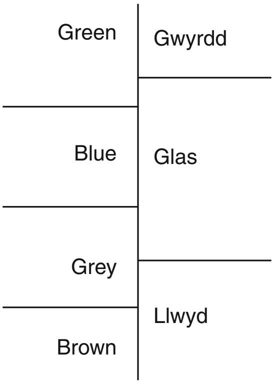


Fig. 2.6
Differences in the naming of certain colours in Welsh and English. A Welsh speaker will refer to the colour of grass and blue sky using the same word. Despite this, many Welsh speakers insist that the translation of "blue" is "glas".

(c) Specialist occupations, such as printing, weaving, etc., lead to refinement of previously learned colour concepts.
(d) Ambient lighting. The grey skies of northern Europe and the subsequent heavy reliance upon electric lighting provides a very different visual environment from that encountered by people living in the tropics. It is well known that lighting alters our perception and naming of colours.

Since the naming of colours is so subjective, we must build Machine Vision systems so that they are able to learn what colours to expect in a given application. This where Pattern Recognition comes into its own.

2.2.6.6 Colour Sensors

By themselves, most video cameras are incapable of discriminating reliably between healthy and diseased tissue, or providing a reliable basis for deciding when fruit on the tree is ready for picking. However, more accurate instruments for measuring the spectral composition of light (from a point source) are available and can be used to calibrate a video camera. Assuming that we have given proper consideration to calibrating the camera, guaranteeing that the light source is stable and sufficiently bright over the visible spectrum and cleaning all optical surfaces, there is no reason why a low-noise solid-state camera cannot be used for colour analysis. The major problem for colour recognition lies elsewhere.

2.2.6.7 RGB Representation

A colour video camera measures the so-called RGB components of the light falling at each point in its retina. It does this by placing monochrome photo-sensors behind three differently coloured filters. These selectively attenuate different parts of the spectrum. R, G and B have traditionally been associated with detecting "red", "green" and "blue" although this idea must not be carried too far. A camera that outputs an HSI video signal (measuring Hue, Saturation and Intensity) uses an RGB sensor and performs the RGB-HSI transformation within its internal electronics.

Let [r(i,j), g(i,j), b(i,j)] denote the measured RGB values at the point (i,j) in a digital image. The set of colours in the image can be represented by S_3 where

$$S_3 = \{[r(i,j),\ g(i,j),\ b(i,j)]/1 \le i,j \le N\} \tag{2.2}$$

By mapping the image into this particular 3-dimensional measurement space (called RGB-space), we forsake all knowledge of the position within the image for each colour vector. However, by extending this definition slightly, we can preserve this information. The set of points

$$S_5 = \{[r(i,j),\ g(i,j),\ b(i,j),\ i,\ j]/1 \le i,\ j \le N\} \tag{2.3}$$

lies in a 5-dimensional space and preserves both position and colour information. For most purposes, S_5 is usually too cumbersome to be useful in practice. The representation embodied in S_3 is normally preferred, when we want to recognise colours independently of where they lie within an image.

We can combine the output of a colour recogniser with the spatial co-ordinates, (i,j), to obtain essentially the same information as is implicit in S_5. The following is a hybrid representation which combines colour labelling (not "raw" RGB values) with geometric position (i,j) within an image.

$$S_{\text{hybrid}} = \{[\text{colour\_name},\ i,\ j]/1 \le i,\ j \le N\} \tag{2.4}$$

This representation is more economical than S_5 yet contains more information than S_3. Moreover, S_{hybrid} can be stored and processed as a monochrome image, in which "intensity" (i.e., an integer) represents symbolic colour names, i.e., a palette. (The result is an image like that used when "painting by numbers".) It is often convenient to display such an image in pseudo-colour, since similar values of the integers defining the values of *colour_name* may represent quite different physical colours. The design and use of a colour recogniser will be explained in ➋ Chap. 16, which also emphasises the innate differences between colour variation in natural products and human artefacts.

2.2.7 Transient Phenomena

The following story was told by a colleague, who is a highly experienced and well-respected vision systems designer. He was commissioned to design a machine to detect patches of mould on the surface of (edible) nuts. The customer supplied him with a large bag of nuts, enabling him to study the problem in detail. He conducted a detailed feasibility study, then designed and built an inspection system. Of course, this took him several weeks. Then, he contacted the client, to arrange a demonstration, prior, to delivering and installing the system in a food-processing factory. A person from the client company arrived for the demonstration, carrying another large bag of nuts. Following good engineering practice, both parties wanted to test the inspection system on previously unseen samples. However, it did not respond well in these tests and failed to satisfy the client's expectations. There was a very simple reason for this disappointment: the new sample of nuts had fresh mould which was *green*, whereas the original samples had older *brown* mould. The client had omitted to tell the vision engineer that the green mould quickly changes colour to brown. This fact was so obvious to the client that he had not thought it important enough to tell the vision engineer. Since brown and green mould patches can be identified by eye, it was "obvious" to the client that both could be detected easily by a Machine Vision system. It is a universal truth that, whenever a client tries to tell the vision engineer how to do his/her job, trouble will follow.

The lesson for us is that materials of biological origin change with time. The surfaces of industrial artifacts can also change their appearance as a result of oxidation, drying, as well as other chemical, physical and biological processes. The visible changes in biological materials can at times be very significant, yet in certain situations they can be very subtle. Dehydration will cause foliage to change shape quickly as it wilts, but early stages of decay are manifest as very slight changes of colour. It is essential that Machine Vision systems be designed on exactly the same type of product/material samples as will be seen by the final, ("target") system. This seemingly obvious principle is not always obeyed!

Transient phenomena occur in a wide variety of applications, particularly in the food and pharmaceutical industries, agriculture and horticulture. On-line data capture is essential whenever the appearance is susceptible to significant changes. In this event, we must build an image-acquisition rig over the production line, even for a feasibility study; off-line analysis will not suffice.

2.2.8 Very Complex Objects

It is nonsensical to search for optimal lighting methods for very complicated objects, because features that are enhanced by one lighting method may be obscured by another, and vice versa. Few objects can match the complexity and unpredictability of form of a living plant. Imagine that we want to view its leaves, perhaps searching for spots indicating fungal disease. Lighting from the sides inevitably casts shadows, leaving "internal" features poorly illuminated. Even omni-directional lighting will fail to produce good images within a dense thicket (➋ *Fig. 2.7*). Coaxial illumination and viewing produces no shadows whatsoever but is prone to creating highlights, due to glinting (➋ *Fig. 2.8*). This can be problematical, if the object surface is shiny, perhaps through it being wet or greasy. Some relief can be provided using a circular polariser (➋ *Fig. 2.9*), or crossed linear polarisers (➋ *Fig. 2.10*). In many applications, however, one standard lighting method can be used, even though it is not optimal for any particular task. Such an arrangement is used in a wide variety of applications where the shape of an object is too complicated to allow optimal lighting methods to be devised.

Machine Vision has the potential to do something that natural vision finds impossible: combine several views of the same object or scene, taken under different lighting conditions. So far, this exciting idea has not been exploited to its full potential. Intelligent lighting, perhaps generating different colours from the same set of "point" sources (multi-colour LEDs) is now a real possibility. Imagine a lighting rig, consisting of an array of multi-colour LEDs, each one individually controlled. By switching the LEDs and combining the images that result, a variety of interesting effects can be achieved that have no counterpart in human vision. Although this has very real potential for inspecting objects that have a highly complicated structure, much more fundamental research needs to be done.

2.2.9 Uncooperative Objects

By definition, inanimate objects are neutral in neither co-operating nor obstructing visual examination. On the other hand, animals such as pigs, chickens and fish will avoid strange and

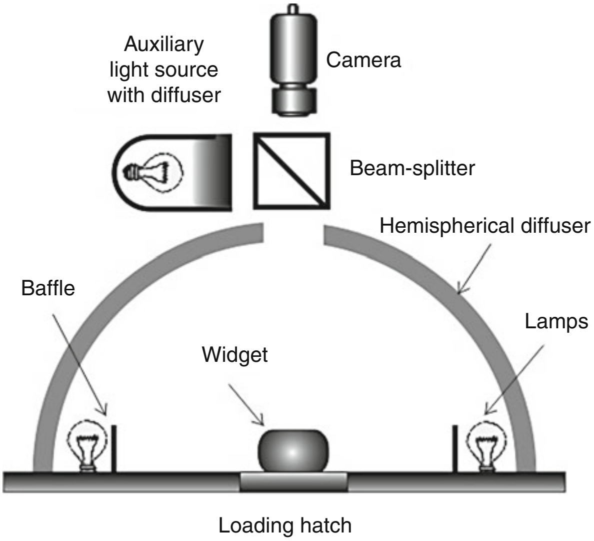


Fig. 2.7
Omni-directional lighting. The auxiliary light source and beam-splitter are used to compensate for the hole cut in the hemi-spherical diffuser

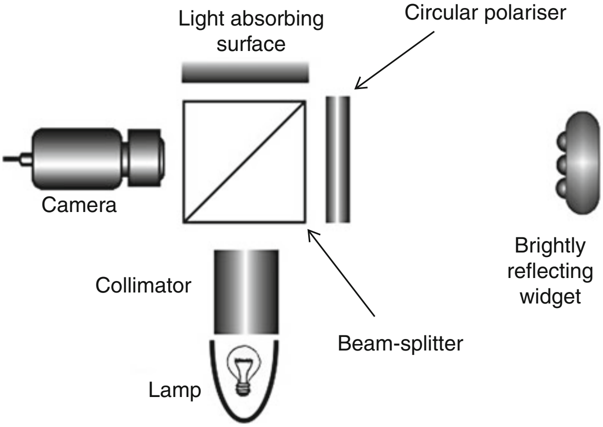


Fig. 2.8
Coaxial illumination and viewing

seemingly threatening equipment, particularly if it makes an unusual noise, has a strange smell, brightly coloured or flashing lights or even superficially resembles the animal's natural predator. These can be major obstacles to gaining an animal's acceptance of a vision system. Moreover, it must be borne in mind that an animal does not see the world as we do. It may be sensitive to a different part of the electro-magnetic spectrum. It may perceive two LEDs as being the eyes of a potential predator. Animals can sometimes be encouraged to move past a camera in a semi-controlled manner by the judicious use of guide rails (pigs, sheep, etc.), or glass panels or tubes

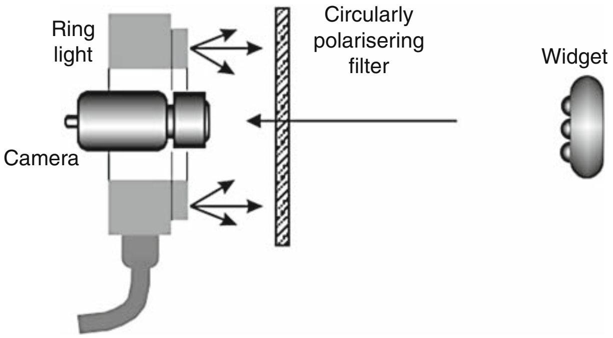


Fig. 2.9
Circular polariser used to reduce the effects of specular reflection

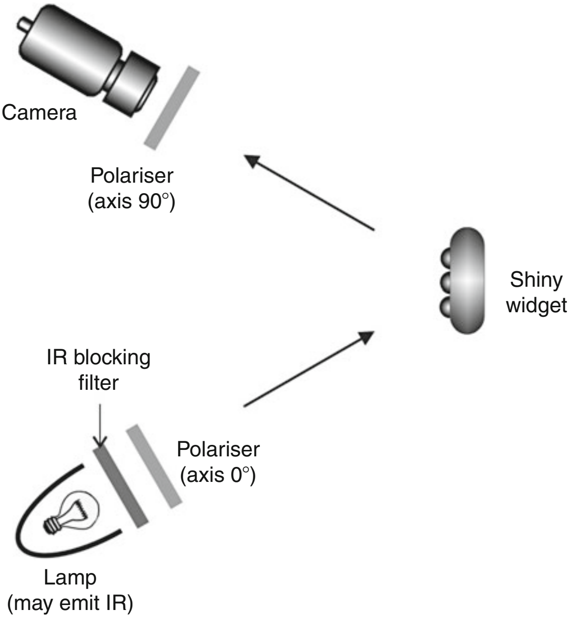


Fig. 2.10
Crossed linear polarisers used to reduce the effects of specular reflection

(insects, fish, etc.). Some animals may also need encouragement, in the form of some prospect of a reward (food), to move at all.

Needless to say, the inspection equipment must be very robust to work safely in close proximity to animals and it must be able to continue working unattended for long periods in extremely dirty working conditions. Designing a suitable image acquisition system for this type of application is far removed from a vision engineer's normal experience, developing algorithms, writing software and designing hardware.. However, it is just as important. Failure to pay attention to these matters will significantly alter the vision system's prospects of success.

2.2.10 Texture

Texture is one of the most difficult areas for Machine Vision engineers, because there is no clear objective definition of what it actually is. Moreover, it is often extremely difficult to collect samples of "good" and "bad" textures, particularly in those situations where texture is merely an indicator of another more significant parameter. For example, the texture of the cut surface of a loaf is known to be related to the "in mouth" feel of the bread. The nature of the texture varies across the slice. (Indeed, bakers often deliberately cause this to happen, by rolling the dough to maximise the surface brightness of white bread when it is cut in the normal manner.) The sensation of biting an apple can be predicted, to some extent, by the cellular-level texture, although the exact nature of the relationship is unclear. The adhesion of paint to a steel surface is known to be affected by the surface texture of the metal. The texture of a coated surface is an indication of its weathering and wear properties. The surface texture of many types of engineering products, most notably cars, white goods, woven fabrics and knitwear, is critical to customer acceptance.

It is not difficult to derive a multitude of measures for representing texture. However, analysing the data causes a severe problem, because we do not know how to interpret the numbers that we obtain. Apart from a few exceptions, texture analysis remains an elusive and enigmatic problem that continues to challenge vision engineers.

2.3 Systems Issues

We conclude this brief introduction to our subject by making some pertinent remarks about the practical issues that must be considered to ensure that an inspection system works properly and reliably over a long period of time. The most important points to remember are:

- Natural products are not all alike; they vary very considerably.
- It is important that the inspection system does not damage, pollute or infect the object being examined.
- It is important the object being examined does not damage the inspection system.
- A series of general principles relating to the design of industrial vision systems summarised in Appendix C and guided the design of the questionnaire in Appendix B. Nearly all of these apply with equal validity to systems for examining natural products.

The criteria by which an inspection system is judged are the same, whether it is applied to natural products or engineering artefacts:

- It must be easy to use; a PhD in Engineering or Computer Science should not be necessary to operate it.
- The system must be acceptable to the work-force. This requires careful attention to a wide variety of factors, including the user interface and other points in this list.
- It must perform at the speed required. (For example, it must be able to keep up with production in a food-processing plant.)

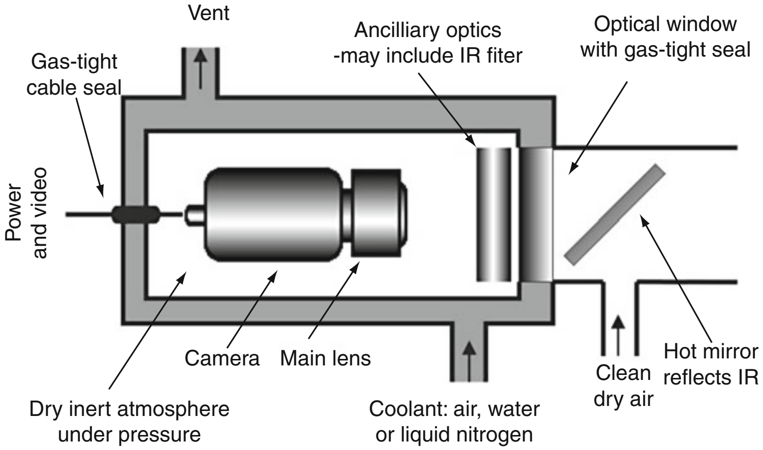

Vent
Gas-tight cable seal
Ancilliary optics -may include IR fiter
Optical window with gas-tight seal
Power and video
Camera
Main lens
Hot mirror reflects IR
Clean dry air
Dry inert atmosphere under pressure
Coolant: air, water or liquid nitrogen

Fig. 2.11
Protecting the camera

- It must perform the necessary function, with a minimum of human intervention. Stringent tests are required to ensure that the system functions properly. Judgement will be made on a statistical basis, against previously agreed criteria, as set out in the specification.
- As far as possible, it should be possible to reuse the system to inspect new product lines.
- The system must be robust and reliable.
- Since cost is always of great importance, the system should be able to pay for itself in a few months. The food industry is notable for operating on very low profit margins, so this is often difficult to achieve.
- The system must be safe and hygienic. Since hygiene is critical in the food and pharmaceutical industries, we deal with this point separately.

Animals are likely to cause particular problems for a vision system, because they create a lot of pollution. They are liable to create a considerable amount of air-borne pollution in the form of droplets, when they urinate, defæcate, sneeze, cough or simply bellow. They also increase the amount of dust in the atmosphere by kicking up straw or feed-stuff, as well as by shedding hair. Animals may even lick a lamp or camera if it is within reach. Making sure that the vision system is protected from pollution of this kind is vitally important, if it is to have a long and useful working life. There are many techniques for doing this, some of which are summarised in ➲ *Fig. 2.11*. Of course, for safety's sake, the animal must also be protected from the vision system. Animal pens are hostile places for vision systems, so protection for both livestock and machine is essential.

Of course, inanimate objects also cause pollution, which can render a vision system useless. Biological materials are even worse than most standard engineering materials in causing air-borne pollution. Flour and other fine powders abound in many food factories, as do mists, fumes, smoke, occasional splashes, etc. All of these can settle on lenses, mirrors, lamp surfaces, and other optical components, causing malfunction of an inspection system.

One of the biggest hazards for a vision system in a food, toiletry, or pharmaceutical factory is *hygiene*. The need to keep all factory equipment scrupulously clean, including all exposed components of the vision system, means that equipment will be washed down on a regular basis. Not only must the optical-lighting-camera sub-system survive being hosed down, it must be easy for factory personnel to dismantle it, so that any traps where debris could accumulate are cleaned.

Further Reading

Books

Batchelor BG, Charlier J-R (1998) Machine vision is not computer vision (Keynote paper). In: Proceedings of SPIE conference on machine vision systems for inspection and metrology VII, Boston, Nov 1998, vol 3521, pp 2–13, ISBN 0-8194-2982-1

Batchelor BG, Waltz FM (2001) Intelligent machine vision: techniques, implementation and applications. Springer, London, ISBN 3-540-76224-8

Batchelor BG, Whelan PF (1997) Intelligent vision systems for industry. Springer, London, ISBN 3-540-19969 1

Crystal D (1997) The Cambridge encyclopedia of language, 2nd edn. Cambridge University press, Cambrdge. ISBN 0-521-55967-7

Davies ER (2000) Image processing for the food industry. World Scientific, Sinagapore. ISBN 981-02-4022-8

Edwards M (2004) Detecting foreign bodies on food. Wooodhead, Cambridge. ISBN 1-85573-729-9

Graves M, Batchelor BG (2003) Machine vision for the inspection of natural products. Springer, London, ISBN 1-85233-525-4

Graves M, Smith A, Batchelor BG (1998) Approaches to foreign body detection in foods. Trends Food Sci Technol 9(1):21–27

Pinder C, Godfrey G (eds) (1993) Food process monitoring systems. Blackie, London, ISBN 0-7514-009-8

Sun DW (2007) Computer vision technology for food quality evaluation. Academic, Burlington. ISBN 10: 0123736420

Technical Articles

Aleixos N, Blasco J, Moltó E, Navarrón F (2000) Assessment of citrus fruit quality using a real-time machine vision system, ICPR00, vol I, pp 482–485

Arnason H, Asmundsson M (1997) Computer vision in food handling and sorting, HPRCV97 (Chapter IV:1). Marel HF, Iceland

Ávila M, Durán ML, Caro A, Antequera T, Gallardo R (2005) Thresholding methods on MRI to evaluate intramuscular fat level on Iberian Ham, IbPRIA05, vol II, p 697

Ávila MM, Durán ML, Antequera T, Palacios R, Luquero M (2007) 3D reconstruction on MRI to analyse marbling and fat level in Iberian Loin, IbPRIA07, vol I, pp 145–152

Backes AR, de M Sá JJ Jr, Kolb RM, Bruno OM (2009) Plant species identification using multi-scale fractal dimension applied to images of adaxial surface epidermis, CAIP09, pp 680–688

Barnes M, Duckett T, Cielniak G (2009) Boosting minimalist classifiers for blemish detection in potatoes, IVCNZ09, pp 397–402

Barni M, Cappellini V, Mecocci A (1997) Color-based detection of defects on chicken meat. Image Vis Comput 15(70):549–556

Barni M, Mussa AW, Mecocci A, Cappellini V, Durrani TS (1995) An intelligent perception system for food quality inspection using color analysis, ICIP95, vol I, pp 450–453

Beare R (2001) An ultrasound imaging system for control of an automated carcass splitting machine; Ballerini L (2001) A simple method to measure homogeneity of fat distribution in meat, SCIA01(O-Th4B)

Belaid A (1990) Metrology in quality control of nuts, ICPR90, vol I, pp 636–638

Belhumeur PN, Chen D, Feiner S, Jacobs DW, Kress WJ, Ling HB, Lopez I, Ramamoorthi R, Sheorey S, White S, Zhang L (2008) Searching the world's herbaria: a system for visual identification of plant species, ECCV08, vol IV, pp 116–129

Benn A, Barrett-Lennard D, Hay PJ (2000) Image analysis for meat. US_Patent 6,104,827, 15 Aug 2000

Berke J, Gyorffy K, Fischl G, Kárpáti L, Bakonyi J (1993) The application of digital image processing in the evaluation of agricultural experiments, CAIP93, pp 780–787

Blasco J, Cubero S, Arias R, Gómez J, Juste F, Moltó E (2007) Development of a computer vision system for the automatic quality grading of mandarin segments, IbPRIA07, vol II, pp 460–466

Bolle RM, Connell JH, Haas N, Mohan R, Taubin G (1996) VeggieVision: a produce recognition system, WACV96, pp 244–251

Bossu J, Gee C, Truchetet F (2008) Development of a machine vision system for a real time precision sprayer. Electron Lett Comput Vision Image Anal 7(3):xx–yy

Burgos-Artizzu XP, Ribeiro A, Tellaeche A, Pajares G, Fernandez-Quintanilla C (2010) Analysis of natural images processing for the extraction of agricultural elements. Image Vis Comput 28(1):138–149

Carrión P, Cernadas E, Gálvez JF, Rodroguez-Damián M, de Sá-Otero P (2004) Classification of honeybee pollen using a multiscale texture filtering scheme. Mach Vision Appl 15(4):186–193

Casanova D, De Mesquita Sá JJ Jr, Bruno OM (2009) Plant leaf identification using Gabor wavelets. Int J Imaging Syst Technol 19(3):236–243

Casasent D, Chen XW (2003) New training strategies for RBF neural networks for X-ray agricultural product inspection. Pattern Recognit 36(2):535–547

Cataltepe Z, Cetin E, Pearson T (2004) Identification of insect damaged wheat kernels using transmittance images, ICIP04, vol V, pp 2917–2920

Cernadas E, Carrión P, Rodriguez PG, Muriel E, Antequera T (2005) Analyzing magnetic resonance images of Iberian pork loin to predict its sensorial characteristics. Comput Vis Image Underst 98(2):344–360

Cernadas E, Durán ML, Antequera T (2002) Recognizing marbling in dry-cured Iberian ham by multiscale analysis. Phys Rev Lett 23(11):1311–1321

Chacon M, Manickavasagan A, Flores-Tapia D, Thomas G, Jayas DS (2007) Segmentation of wheat grains in thermal images based on pulse coupled neural networks, ICIP07, vol II, pp 273–276

Chalidabhongse TH, Yimyam P, Sirisomboon P (2006) 2D/3D vision-based Mango's feature extraction and sorting, ICARCV06, pp 1–6

Chapron M, Boissard P, Assemat L (2000) A multiresolution based method for recognizing weeds in corn fields, ICPR00, vol II, pp 303–306

Chapron M, Khalfi K, Boissard P, Assemat L (1997) Weed recognition by color image processing, SCIA97 (xx–yy) 9705; Hahn F, Mota R, Nobel Chile Jalapeno sorting using structured laser and neural network classifiers, CIAP97, vol II, pp 517–523

Chapron M, Martin-Chefson L, Assemat L, Boissard P (199) A multiresolution weed recognition method based on multispectral image processing, SCIA99 (image analysis)

Chatterjee S (2008) Anisotropic diffusion and segmentation of colored flowers, ICCVGIP08, pp 599–605

Chen M, Dhingra K, Wu W, Yang L, Sukthankar R, Yang J (2009) PFID: Pittsburgh fast-food image dataset, ICIP09, pp 289–292

Chen Z, Tao Y (2001) Food safety inspection using "from presence to classification" object-detection model. Pattern Recognit 34(12):2331–2338

Cho SY, Lim PT (2006) A novel virus infection clustering for flower images identification, ICPR06, vol II, pp 1038–1041

Clarke J, Barman S, Remagnino P, Bailey K, Kirkup D, Mayo S, Wilkin P (2006) Venation pattern analysis of leaf images, ISVC06, vol II, pp 427–436

Codrea CM, Aittokallio T, Keränen M, Tyystjärvi E, Nevalainen OS (2003) Feature learning with a genetic algorithm for fluorescence fingerprinting of plant species. Phys Rev Lett 24(15):2663–2673

Davies ER, Bateman M, Mason DR, Chambers J, Ridgway C (2003) Design of efficient line segment detectors for cereal grain inspection. Phys Rev Lett 24(1–3):413–428

Davies R, Heleno P, Correia BAB, Dinis J (2001) VIP3D: an application of image processing technology for quality control in the food industry, ICIP01, vol I, pp 293–296

Derganc J, Likar B, Bernard R, Tomazevic D, Pernus F (2003) Real-time automated visual inspection of color tablets in pharmaceutical blisters. Real-Time Imaging 9(2):113–124

Díaz G, Romero E, Boyero JR, Malpica N (2009) Recognition and quantification of area damaged by Oligonychus perseae in avocado leaves, CIARP09, pp 677–684

Ding K, Gunasekaran S (1998) 3-dimensional image reconstruction procedure for food microstructure evaluation. Artif Intell Rev 12(1–3):245–262

Dissing BS, Clemmesen LH, Loje H, Ersboll BK, Adler-Nissen J (2009) Temporal reflectance changes in vegetables, CRICV09, pp 1917–1922

Dobrusin Y, Edan Y, Grinshpun J, Peiper UM, Wolf I, Hetzroni A (1993) Computer image analysis to locate targets for an agricultural robot, CAIP93, pp 775–779

Edan Y (1995) Design of an autonomous agricultural robot. Appl Intell 5(1):41–50

Edmatsu K, Nitta Y (1981) Automated capsule inspection method. Pattern Recognit 14(1–6):365–374

Enomoto K, Toda M, Kuwahara Y (2009) Scallop detection from sand-seabed images for fishery investigation, CISP09, pp 1–5

Erz G, Posch S (2003) A region based seed detection for root detection in minirhizotron images, DAGM03, pp 482–489

Feyaerts F, Van Gool LJ (2001) Multi-spectral vision system for weed detection. Phys Rev Lett 22 (6–7):667–674

Foschi PG, Liu H (2004) Active learning for detecting a spectrally variable subject in color infrared imagery. Phys Rev Lett 25(13):1509–1517

Fox JS, Weldon E Jr, Ang M (1985) Machine vision techniques for finding sugarcane seedeyes, CVPR85. (Univ. of Hawaii) Hough. Feature Computation. Interesting use of pyramids and hough transform, pp 653–655

France I, Duller AWG, Lamb HF, Duller GAT (1997) A comparative study of approaches to automatic pollen identification, BMVC97, pp xx–yy

Fu H, Chi Z (2006) Combined thresholding and neural network approach for vein pattern extraction from leaf images. IEE P-Vis Image Signal Process 153(6):881–892

Galleguillos C, Babenko B, Rabinovich A, Belongie SJ (2008) Weakly supervised object localization with stable segmentations, ECCV08, vol I, pp 193–207

Galleguillos C, Faymonville P, Belongie SJ (2009) BUBL: an effective region labeling tool using a hexagonal lattice, Emergent09, pp 2072–2079

Galleguillos C, Rabinovich A, Belongie SJ (2008) Object categorization using co-occurrence, location and appearance, CVPR08, pp 1–8

Garcia-Consuegra J, Cisneros G, Martinez A (199) A methodology for woody crop location and discrimination in remote sensing, CIAP99, pp 810–815

Ghorbel F, de Bougrenet de la Tocnaye JL (1990) Automatic control of Lamellibranch Larva growth using contour invariant feature extraction. Pattern Recognition 23(3–4):319–323

Gómez-Sanchis J, Camps-Valls GMoltó E, Gómez-Chova L, Aleixos N, Blasco J (2008) Segmentation of hyperspectral images for the detection of rotten mandarins, ICIAR08, pp xx–yy

Gouiffes M, Fernandez-Maloigne C, Tremeau A, Collewet C (2004) Color segmentation of ink-characters: application to meat tracabeality control, ICIP04, vol I, pp 195–198

Grasso GM, Recce M (1997) Scene analysis for an orange harvesting robot. Artif Intell Appl 11(3):9–15

Gregori M, Lombardi L, Savini M, Scianna A (1997) Autonomous plant inspection and anomaly detection, CIAP97, vol II, pp 509–516

Guo M, Ou ZY, Wei HL (2006) Inspecting ingredients of starches in starch-noodle based on image processing and pattern recognition, ICPR06, vol II, pp 877–880

Han MH, Jang D, Foster J (1989) Inspection of 2-D objects using pattern matching method. Pattern Recognit 22(5):567–575

Harrell RC, Slaughter DC, Adsit PD (1989) A fruit-tracking system for robotic harvesting. Mach Vision Appl 2(2):69–80

Harron W, Dony R (2009) Predicting quality measures in beef cattle using ultrasound imaging, CIIP09, pp 96–103

Heinemann PH, Pathare NP, Morrow CT (1996) An automated inspection station for machine-vision grading of potatoes. Mach Vision Appl 9(1):14–19

Huang Q, Jain AK, Stockman GC, Smucker AJM (1992) Automatic image analysis of plant root structures, ICPR92, vol II, pp 569–572

Im C, Nishida H, Kunii TL (1998) A hierarchical method of recognizing plant species by leaf shapes, MVA98, pp xx–yy

Impoco G, Licitra G (2009) An interactive level set approach to semi-automatic detection of features in food micrographs, CAIP09, pp 914–921

Ishii A, Mizuta T, Todo S (1998) Detection of foreign substances mixed in a plastic bottle of medicinal solution using real-time video image processing, ICPR98, vol II, pp 1646–1650

Jackman P, Sun DW, Du CJ, Allen P (2009) Prediction of beef eating qualities from colour, marbling and wavelet surface texture features using homogenous carcass treatment. Pattern Recognit 42(5):751–763

Jiménez AR, Ceres R, Pons JL (1999) A machine vision system using a laser radar applied to robotic fruit harvesting, CVBVS99, p 110

Jiménez AR, Ceres R, Pons JL (2000) A vision system based on a laser range-finder applied to robotic fruit harvesting. Mach Vision Appl 11(6):321–329

Jiménez AR, Jain AK, Ceres R, Pons JL (1999b) Automatic fruit recognition: a survey and new results using range/attenuation images. Pattern Recognit 32(10):1719–1736

Jin WB, Gu WZ, Zhang ZF (2009) An improved method for modeling of leaf venation patterns, CISP09, pp 1–5

Jones R, Frydendal I (1998) Segmentation of sugar beets using image and graph processing, ICPR98, vol II, pp 1697–1699

Joutou T, Yanai K (2009) A food image recognition system with multiple kernel learning, ICIP09, pp 285–288

Kaewapichai W, Kaewtrakulpong P, Prateepasen A, Khongkraphan K (2007) Fitting a pineapple model for automatic maturity grading, ICIP07, vol I, pp 257–260

Kirchgessner N, Spies H, Scharr H, Schurr U (2001) Root growth analysis in physiological coordinates, CIAP01, pp 589–594

Kita N, Kita Y, Yang HQ (2002) Archiving technology for plant inspection images captured by mobile active cameras "4D visible memory", 3DPVT02, pp 208–213

Kunii TL, Im C, Nishida H (1998) Recognizing plant species by leaf shapes: a case study of the acer family, ICPR98, vol II, pp 1171–1173

Laemmer E, Deruyver A, Sowinska M (2002) Watershed and adaptive pyramid for determining the apple's maturity state, ICIP02, vol I, pp 789–792

Larsen R, Arngren M, Hansen PW, Nielsen AA (2009) Kernel based subspace projection of near infrared hyperspectral images of maize kernels, SCIA09, pp 560–569

Lee CL, Chen SY (2006) Classification of leaf images. Int J Imaging Syst Technol 16(1):15–23

Lefebvre M, Gil S, Glassey MA, Baur C, Pun T (1992) 3D computer vision for agrotics: the potato operation, an overview, ICPR92, vol I, pp 207–210

Lepistö L, Kunttu I, Autio J, Visa A (2005) Classification of natural images using supervised and unsupervised classifier combinations, CIAP05, pp 770–777

Lepistö L, Kunttu I, Lähdeniemi M, Tähti T, Nurmi J (2007) Grain size measurement of crystalline products using maximum difference method, SCIA07, pp 403–410

Lepistö L, Kunttu I, Visa A (2005) Color-based classification of natural rock images using classifier combinations, SCIA05, pp 901–909

Li MW, Zhang W (2009) Research and implement of head milled rice detection high-speed algorithm based on FPGA, CISP09, pp 1–4

Li YF, Lee MH (1996) Applying vision guidance in robotic food handling. IEEE Rob Autom Mag 3(1):4–12

Liu QS, Liu GH, Using tasseled cap transformation of CBERS-02 images to detect dieback or dead Robinia pseudoacacia plantation, CISP09, pp 1–5

Lumme J, Karjalainen M, Kaartinen H, Kukko A, Hyyppä J, Hyyppä H, Jaakkola A, Kleemola J (2008) Terrestrial laser scanning of agricultural crops, ISPRS08 (B5: 563 ff)

Ma W, Zha HB (2008) Convenient reconstruction of natural plants by images, ICPR08, pp 1–4

Ma W, Zha HB, Liu J, Zhang XP, Xiang B (2008) Image-based plant modeling by knowing leaves from their apexes, ICPR08, pp 1–4

Mao WH, Ji BP, Zhan JC, Zhang XC, Hu XA (2009) Apple location method for the apple harvesting robot, CISP09, pp 1–5

Martínez-Usó A, Pla F, García-Sevilla P (2005) Multi-spectral image segmentation by energy minimization for fruit quality estimation, IbPRIA05, vol II, pp 689

Mathiassen JR, Misimi E, Skavhaug A (2007) A simple computer vision method for automatic detection of melanin spots in atlantic salmon fillets, IMVIP07, pp 192–200

McQueen AM, Cherry CD, Rando JF, Schler MD, Latimer DL, McMahon SA, Turkal RJ, Reddersen BR (2000) Object recognition system and method. US_Patent 6,069,696, 30 May 2000

Meade R (2006) Oven conveyor alignment system apparatus and method. US_Patent 7,131,529, 7 Nov 2006; US_Patent 7,222,726, 29 May 2007

Merler M, Galleguillos C, Belongie SJ (2007) Recognizing groceries in situ using in vitro training data, SLAM07, pp 1–8

Miao ZJ, Gandelin MH, Yuan BZ (2006) A new image shape analysis approach and its application to flower shape analysis. Image Vis Comput 24(10):1115–1122

Miao ZJ (2000) Zernike moment-based image shape analysis and its application. Phys Rev Lett 21(2):169–177

Mizuno S, Noda K, Ezaki N, Takizawa H, Yamamoto S (2007) Detection of Wilt by analyzing color and stereo vision data of plant, MIRAGE07, pp 400–411

Moeslund TB, Aagaard M, Lerche D (2005) 3D pose estimation of cactus leaves using an active shape model, WACV05, vol I, pp 468–473

Mühlich M, Truhn D, Nagel K, Walter A, Scharr H, Aach T (2008) Measuring plant root growth, DAGM08, pp xx–yy

Nam YY, Hwang EJ, Kim DY (2008) A similarity-based leaf image retrieval scheme: joining shape and venation features. Comput Vis Image Underst 110(2):245–259

Ni H, Gunasekaran S (1998) A computer vision method for determining length of cheese shreds. Artif Intell Rev 12(1–3):27–37

Nilsback ME, Zisserman A (2008) Automated flower classification over a large number of classes, ICCVGIP08, pp 722–729, IEEE DOI Link

Nilsback ME, Zisserman A (2010) Delving deeper into the whorl of flower segmentation. Image Vis Comput 28(6):1049–1062, Elsevier DOI Link

Pajares G, Tellaeche A, Burgosartizzu XP, Ribeiro A (2007) Design of a computer vision system for a differential spraying operation in precision agriculture using hebbian learning. IET Comput Vision 1(3–4):93–99

Pandit RB, Tang J, Liu F, Mikhaylenko G (2007) A computer vision method to locate cold spots in foods in microwave sterilization processes. Pattern Recognit 40(12):3667–3676

Park JK, Hwang EJ, Nam YY (2008) Utilizing venation features for efficient leaf image retrieval. J Syst Softw 81(1):71–82

Patel VC, McClendon RW, Goodrum JW (1998) Color computer vision and artificial neural networks for the detection of defects in poultry eggs. Artif Intell Rev 12(1–3):163–176

Patel VC, McClendon RW, Goodrum JW (1994) Crack detection in eggs using computer vision and neural networks. Artif Intell Appl 8(2):21–31

Patel VC, McClendon RW, Goodrum JW (1996) Detection of cracks in eggs using color computer vision and artificial neural networks. Artif Intell Appl 10(3):19–28

Peleg K, Cohen O, Ziv M, Kimmel E (1993) Machine identification of buds in images of plant shoots. Mach Vision Appl 6(4):224–232

Plebe A, Grasso G (2001) Localization of spherical fruits for robotic harvesting. Mach Vision Appl 13(2):70–79

Polder G, van der Heijden GWAM, Young IT (2003) Tomato sorting using independent component analysis on spectral images. Real-Time Imaging 9(4):253–259

Portman N, Grenander U, Vrscay ER (2009) Direct estimation of biological growth properties from image data using the “GRID” model, ICIAR09, pp 832–843

Qi LY, Yang QH, Bao GJ, Xun Y, Zhang L (2009) A dynamic threshold segmentation algorithm for cucumber identification in greenhouse, CISP09, pp 1–4

Qingyu W, Nan J, Heng L, Bin H (2009) A system for dynamically monitoring and warning algae blooms in Taihu Lake based on remote sensing, CISP09, pp 1–5

Rabinovich A, Vedaldi A, Galleguillos C, Wiewiora E, Belongie SJ, Objects in context, ICCV07, pp 1–8

Recce M, Plebe A, Taylor J, Tropiano G (1998) Video grading of oranges in real time. Artif Intell Rev 12(1–3):117–136

Rodenacker K, Gais P, Juetting U, Hense BA (2001) (Semi-) automatic recognition of microorganisms in water, ICIP01, vol III, pp 30–33

Rodriguez-Damian M, Cernadas E, Formella A, Fernandez-Delgado M, DeSa-Otero P (2006) Automatic detection and classification of grains of pollen based on shape and texture. IEEE Trans Syst Man Cybern, C: Appl Rev 36(4):531–542

Rodroguez-Damian M, Cernadas E, de Sa-Otero P, Formella A (2004) Pollen classification using brightness-based and shape-based descriptors, ICPR04, vol II, pp 212–215

Ronneberger O, Burkhardt H, Schultz E (2002) General-purpose object recognition in 3D volume data sets using gray-scale invariants: classification of airborne pollen-grains recorded with a confocal laser scanning microscope, ICPR02, vol II, pp 290–295

Saitoh T, Aoki K, Kaneko T (2004) Automatic recognition of blooming flowers, ICPR04, vol I, pp 27–30

Saitoh T, Kaneko T (2000) Automatic recognition of wild flowers, ICPR00, vol II, pp 507–510

Samal A, Peterson B, Holliday DJ (1994) Recognizing plants using stochastic L-systems, ICIP94, vol I, pp 183–187

Sánchez AJ, Marchant JA (2000) Fusing 3D information for crop/weeds classification, ICPR00, vol IV, pp 295–298

Sanchiz JM, Pla F, Marchant JA (1998) An approach to the vision tasks involved in an autonomous crop protection vehicle. Eng Appl Artif Intell 11(2):175–187

Sanchiz JM, Pla F, Marchant JA, Brivot R (1995) Plant tracking-based motion analysis in a crop field, CAIP95, pp 302–309

Schatzki TF, Witt SC, Wilkins DE, Lenker DH (1981a) Characterization of growing lettuce from density contours: I. Head selection. Pattern Recognit 13(5):333–340

Schatzki TF, Witt SC, Wilkins DE, Lenker DH (1981b) Characterization of growing lettuce from density contours: II. Statistics. Pattern Recognit 13(5):341–346

Šeatovic D (2008) A segmentation approach in novel real time 3D plant recognition system, CVS08, pp xx–yy

Shi CJ, Ji GR (2009) Recognition method of weed seeds based on computer vision, CISP09, pp 1–4

Shiranita K, Hayashi K, Otsubo A, Miyajima T, Takiyama R (2000) Grading meat quality by image processing. Pattern Recognit 33(1):97–104

Shiranita K, Miyajima T, Takiyama R (1998) Determination of meat quality by texture analysis. Phys Rev Lett 19(14):1319–1324

Somers B, Delalieux S, Verstraeten WW, Verbesselt J, Lhermitte S, Coppin P (2009) Magnitude- and shape-related feature integration in hyperspectral mixture analysis to monitor weeds in citrus orchards. IEEE Trans Geosci Remote Sens 47(11):3630–3642

Song Y, Wilson RG, Edmondson R, Parsons N (2007) Surface modelling of plants from stereo images, 3DIM07, pp 312–319

Sun CM, Berman M, Coward D, Osborne B (2007) Thickness measurement and crease detection of wheat grains using stereo vision. Phys Rev Lett 28(12):1501–1508

Sun DW (ed) (2008) Computer vision technology for food quality evaluation. Elsevier, San Diego, ISBN: 978-0-12-373642-0

Sun X, Gong HJ, Zhang F, Chen KJ (2009) A digital image method for measuring and analyzing color characteristics of various color scores of beef, CISP09, pp 1–6

Tadeo F, Matia D, Laya D, Santos F, Alvarez T, Gonzalez S (2001) Detection of phases in sugar crystallization using wavelets, ICIP01, vol III, pp 178–181

Tang XD, Liu MH, Zhao H, Tao W (2009) Leaf extraction from complicated background, CISP09, pp 1–5

Tao Y (1996) Spherical transform of fruit images for online defect extraction of mass objects. Opt Eng 35(2):344–350

Taouil K, Chtourou Z, Kamoun L (2008) Machine vision based quality monitoring in olive oil conditioning, IPTA08, pp 1–4

Taylor-Hell JF, Baranoski GVG, Rokne JG (2005) State of the art in the realistic modeling of plant venation systems. Int J Image Graph 5(3):663–678

Tellaeche A, Burgos-Artizzu XP, Pajares G, Ribeiro A (2008) A vision-based method for weeds identification through the Bayesian decision theory. Pattern Recognit 41(2):521–530

Teng CH, Kuo YT, Chen YS (2009) Leaf segmentation, its 3D position estimation and leaf classification from a few images with very close viewpoints, ICIAR09, pp 937–946

Toraichi K, Kwan PWH, Katagishi K, Sugiyama T, Wada K, Mitsumoto M, Nakai H, Yoshikawa F (2002) On a fluency image coding system for beef marbling evaluation. Phys Rev Lett 23(11):1277–1291

Vazquez-Fernandez E, Dacal-Nieto A, Martin F, Formella A, Torres-Guijarro S, Gonzalez-Jorge H (2009) A computer vision system for visual grape grading in wine cellars, CVS09, pp 335–344

Vioix JB, Douzals JP, Truchetet F, Assémat L, Guillemin JP (2002) Spatial and spectral methods for weed detection and localization. Eurasip J Appl Signal Process 7:679–685

Wang HM (2009) Impact of magnetic field on Mung Bean Ultraweak Luminescence, CISP09, pp 1–3

Wang Q, Ronneberger O, Burkhardt H (2007) 3D invariants with high robustness to local deformations for automated pollen recognition, DAGM07, pp 425–435

Watchareeruetai U, Takeuchi Y, Matsumoto T, Kudo H, Ohnishi N (2006) Computer vision based methods for detecting weeds in lawns. Mach Vision Appl 17(5):287–296

Wijethunga P, Samarasinghe S, Kulasiri D, Woodhead I (2009) Towards a generalized colour image segmentation for kiwifruit detection, IVCNZ09, pp 62–66

Wiwart M, Koczowska I, Borusiewicz A (2001) Estimation of Fusarium head blight of triticale using digital image analysis of grain, CAIP01(563 ff)

Xie NH, Li X, Zhang XQ, Hu WM, Wang JZ (2008) Boosted cannabis image recognition, ICPR08, pp 1–4

Xun Y, Yang QH, Bao G, Gao F, Li W (2009) Recognition of broken corn seeds based on contour curvature, CISP09, pp 1–5

Ye L, Cao LZ, Ogunbona P, Li WQ (2005) Description of evolutional changes in image time sequences using MPEG-7 visual descriptors, VLBV05, pp 189–197

Yoshikawa F, Toraichi K, Wada K, Ostu N, Nakai H, Mitsumoto M, Katagishi K (2000) On a grading system for beef marbling. Phys Rev Lett 21(12):1037–1050

Zeng G, Birchfield ST, Wells CE (2006) Detecting and measuring fine roots in minirhizotron images using matched filtering and local entropy thresholding. Mach Vision Appl 17(4):265–278

Zeng G, Birchfield ST, Wells CE (2010) Rapid automated detection of roots in minirhizotron images. Mach Vision Appl 21(3):xx–yy

Zhang YH, Zhang YS, He ZF, Tang XY (2007) Automatic inspection of tobacco leaves based on MRF image model, ISVC07, vol II, pp 662–670

Zhang ZY, Wang FX, de B Harrington P (2009) Two-dimensional mid- and near-infrared correlation spectroscopy for rhubarb Identification, CISP09, pp 1–6

Zhou LY, Chalana V, Kim Y (1998) PC-based machine vision system for real-time computer-aided potato inspection. Int J Imaging Syst Technol 9(6):423–433

Zhu WX, Lu CF, Li XC, Kong LW (2009) Dead birds detection in modern chicken farm based on SVM, CISP09, pp 1–5

Zou J, Nagy G (2004) Evaluation of model-based interactive flower recognition, ICPR04, vol II, pp 311–314

Zou XB, Zhao J, Li YX (2007) Apple color grading based on organization feature parameters. Phys Rev Lett 28(15):2046–2053

3 Human and Animal Vision

Jonathan T. Erichsen · J. Margaret Woodhouse
Cardiff University, Cardiff, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_3,

Abstract: Since this is a book about artificial vision, it may seem strange that we include an essay on vision in the natural world. The reason is simple: we need to convince the reader that there is no single model for vision that is ideal for all circumstances. Our own visual system, sophisticated though it is, would not suit a housefly, eagle, owl, chameleon, or even a horse. Each of these animals has managed to thrive within a specific type of environment, for far longer than human beings have walked on two legs. Even though it is very limited in its visual capability compared to human beings, a frog has a vision system that has served it well for many millions of years. In many instances, an industrial Machine Vision system is like a frog: both can do certain well-defined tasks, without attempting to achieve the subtlety of discrimination and the high level of intelligent scene analysis that a human being can accomplish. In nature, there are four scenes that nearly every organism has to recognise: food, threat, mate, and shelter. This is not so very different from an inspection system that is merely intended to distinguish between 'good' and 'faulty' products. Animals do, however, have to cope with highly variable environments and cannot change the viewing conditions as a vision engineer can do to make his/her creation cheaper, faster, or more reliable. So, our task is usually simpler than designing a machine as complex as a frog. We must never make the mistake of thinking that only we see the world as it really is. We are only sensitive to a narrow range of optical wavelengths. We cannot sense polarisation, whereas bees and appropriately designed machines can. We cannot sense very low light levels that some animals have to use. Why should we want to replicate a system like ours that is confused by numerous visual illusions? The natural world has produced many different models for vision. For example, there are over 40 different types of eye. The compound eye of the insects and our own eye are obviously very different. Even the eye of a relatively close relation, the cat, is significantly different from our own, having a vertical pupil in bright light. Some animal eyes move the lens to focus on objects that are moving along the line of sight, rather than alter the shape of the lens, as our own eyes do. There is nothing particularly special about our colour vision either. Mantis shrimps, for example, have 12 different colour pigments in their eyes. Spiders are not content with just two eyes as we are; they have eight! This variety does more than present us with a collection of interesting curiosities; it provides inspiration! Vision engineers can benefit from learning about non-human vision systems. Perhaps the most important point of all is that vision engineers should have the confidence to be different; we should not be constrained always to try and emulate human vision. Nature isn't! Although it is immensely powerful, human vision is sometimes ineffective, and in other cases grossly over-sophisticated, for the humble tasks required of industrial vision systems. It is always vain to do with more what can be done with less. So, if frog vision works, use it!

Vision is pervasive in the animal kingdom and is a sensory capability that is variously relied upon to find reproductive mates, suitable food, and/or shelter while avoiding predators or other dangers. The receptor organs (e.g., eyes) and associated visual system not only provide the basis for visual perception of the surrounding environment but also drive and guide the visuomotor responses (e.g., locomotion, eye movements, reaching to grasp, etc.) in reaction to changes or objects of interest in the world. Although animal vision has intrinsic limits that vary from species to species as well as different properties (e.g., colour vision) that may arise for specific purposes in the course of evolution, its main characteristic is that it represents a generalised adaptation that can deal with a diverse range of problems and challenges. In contrast, machine vision is narrowly designed for a particular purpose or function and its related systems are normally highly constrained. This chapter offers a broad summary of human and animal vision, with an emphasis on vertebrates, paying particular attention not only to its basic design properties and limitations but also the sheer diversity of biological solutions to the problem of perceiving the world and responding appropriately. The hope is that such an examination of vision from a variety of perspectives will inform and inspire the future design and engineering of machine vision.

3.1 Introduction

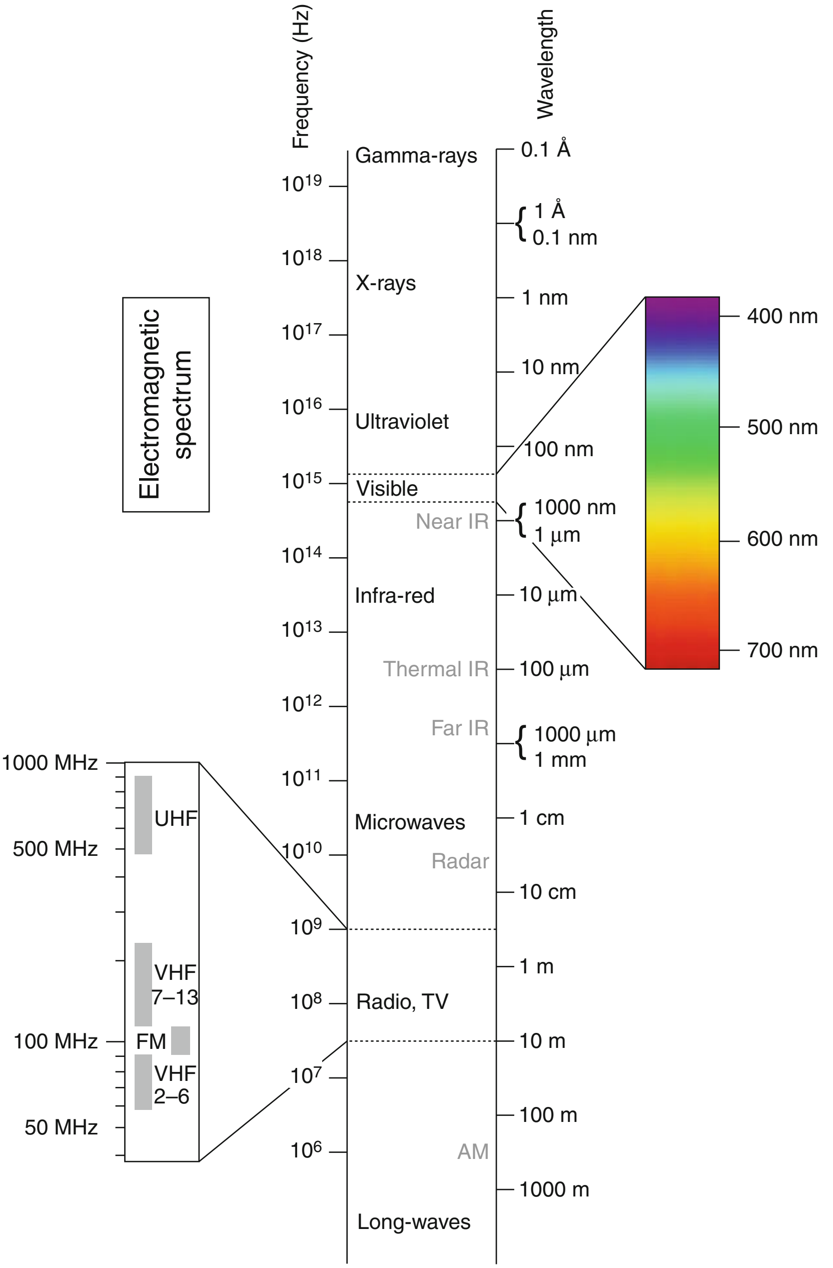
The capacity for vision, ranging from a crude sensitivity to light through to an acute appreciation of form, movement, and colour is found in nearly all living creatures. It is the dominant sense in many animals, and arguably reaches it highest level in primates and birds, whereas in those relatively few animals without sight, there is evidence that vision has only been lost in the course of evolution rather than having always been absent. The pervasiveness of vision may not always have been the case. Andrew Parker in his eloquent book, *In the Blink of an Eye*, argues that, prior to the Cambrian era, all animal life was blind. Indeed, he argues that it was the appearance of a sensitivity to light (➋ *Fig. 3.1*) that heralded a veritable explosion in the diversity of life forms as well as in the wide range of methods for achieving visual perception.

The evolutionary history of vision since that time is characterised by two major themes. In a given line of animals, vision is guided and at times perhaps constrained by the structural and physiological approach taken (e.g., compound eyes of insects, the 'inverted' retina of vertebrates, etc.). On the other hand, the existence of numerous convergent structures (e.g., vertebrate and squid eyes, the same types of visually guided movement found in insects and mammals, etc.) strongly suggests that they are solutions necessitated by general problems in vision that must be solved by all animals that need to achieve visually guided motion. The structural and optical constraints, as well as the sheer diversity of these biological solutions, affords the opportunity to examine vision from a variety of perspectives, and such analyses have the potential to inspire the design and engineering of machine vision.

In the present chapter, we will begin with an overview of the design characteristics of various vertebrate eyes and properties of visual systems. Then, after a brief consideration of the basic aspects of machine vision, we will turn our attention to human vision, such as the reader will be familiar with from his/her own experience. Finally we will examine the similarities and differences that arise amongst diverse animal species. Such an approach will allow us to highlight the instances of convergence in structure to achieve a particular function. Moreover, we can identify illustrative examples of the way in which constraints (e.g., purpose or environment) have moulded the design of visual systems.

3.2 Overview

Vision begins with the organ that is sensitive to light. In vertebrates, the eye has a common fundamental plan (➋ *Fig. 3.2*). At the front of the eye is a transparent refracting surface (the cornea) and at the back lies a layer of neural tissue (the retina) with light-sensitive elements (the photoreceptors). In between, there is a variable aperture (the pupil) and behind that a second refracting element (the lens). The lens affords the possibility of changing the focal length or power (accommodation) to adjust for the distance of an object of interest. The ability to accommodate varies considerably between animals and even in the same animal with age.



Louis E. Keiner - Coastal Carolina University

Fig. 3.1

The light wavelengths that are visible to humans. Visible light is only a small portion of the electromagnetic radiation spectrum

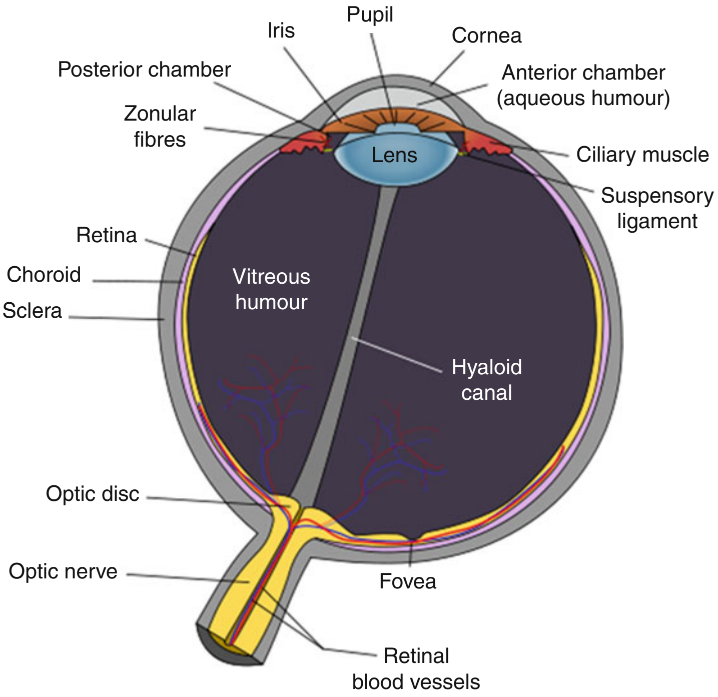


Fig. 3.2
The human eye

3.2.1 Eye Orientation

Every vertebrate has two eyes, but the orientation of the eyes in the head has a dramatic impact on the overall field of view, quite apart from any differences in optical design within the eye. Humans and owls, for example, have frontally placed eyes and, for that reason, must move their eyes or head in order to sweep a wide field. In contrast, a horse's or pigeon's eyes are angled away from each other and therefore provide an almost panoramic view of the world (*Fig. 3.3*).

3.2.2 Visual Performance

The quality of vision (e.g., resolution) is not uniform across the retina (or the field of view). Most animals have a central area in each eye that provides higher resolution and is normally used for fixating and inspecting an object of interest. This area is located centrally in the retina in order to take advantage of the better quality image along the optical axis of the refracting surfaces. In some species (e.g., humans and birds), there is an additional specialisation in this central area that takes the form of a depression in the retina itself (the fovea) (*Fig. 3.4*). The fovea varies in shape and depth across animals, but its function is to maximise the quality of the image reaching the photoreceptors by the displacement of some of the overlying retinal tissue. This retinal adaptation is another example of maximising the quality of the optical image. Whether or not there is a fovea, the density of photoreceptors is usually higher in the central

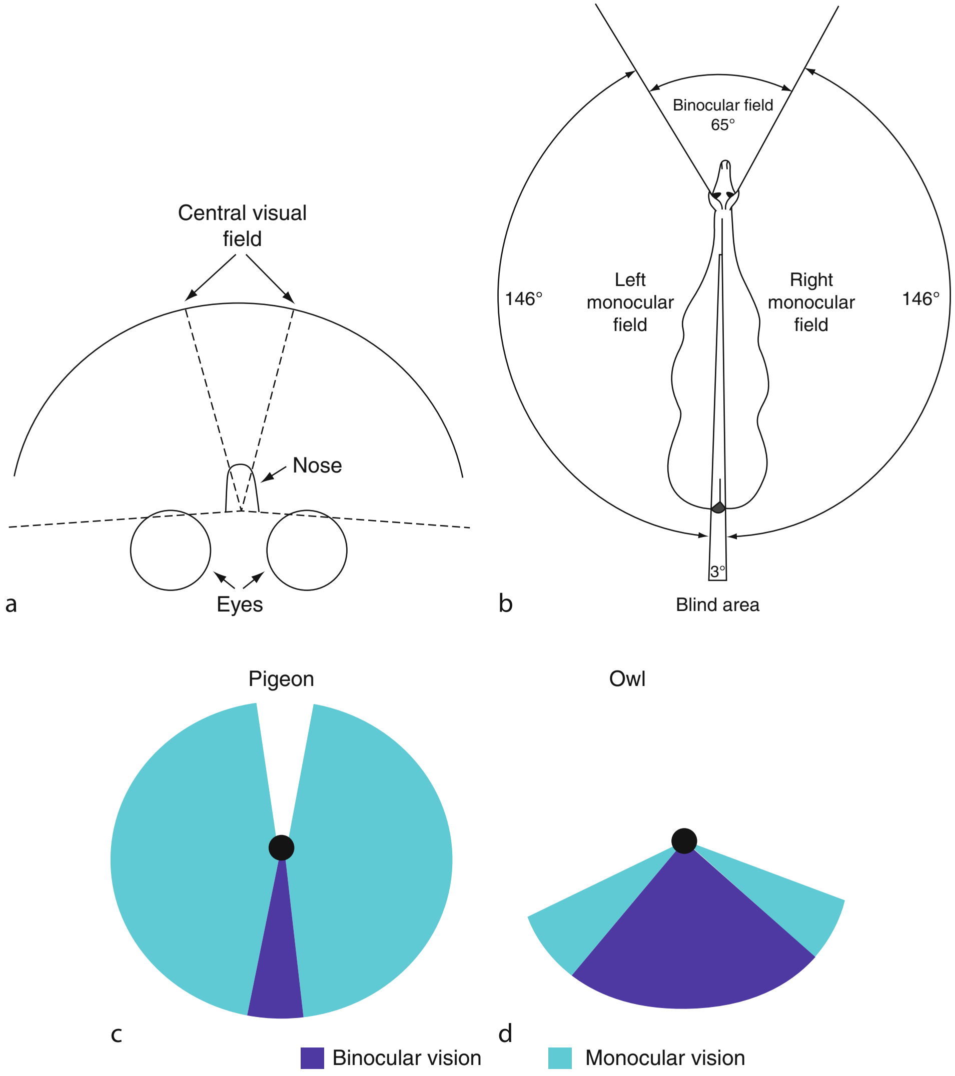


Fig. 3.3

The very different fields of view of a: (a) human (b) horse (c) pigeon (d) owl

area than in any other part of the retina. Taken together with the quality of the image reaching the photoreceptors, this provides the basis for the high acuity vision possible at the central retinal location and explains why it is preferentially used for visual fixation.

In order to obtain the most information from the visual world, there are many aspects of vision that must be optimised. These include resolution, contrast sensitivity, motion perception and, for many species, colour discrimination. Resolution is the ability to discriminate detail, for example, the ability in humans to distinguish correctly between different letters on an optometrist's eye chart. Contrast sensitivity is the ability to perceive an object against a background, in terms of brightness difference. In its simplest form, motion sensitivity is

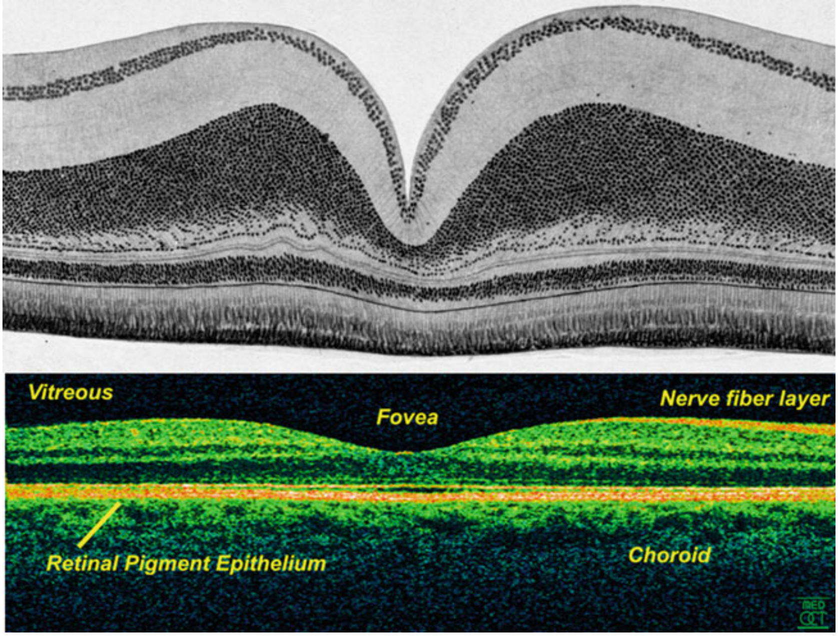


Fig. 3.4
Cross-sectional view of the central retina, including the fovea of a bird (*Sterna minuta*) (*upper figure*) and a human eye (*lower figure*)

the ability to perceive changes in any visual stimulus over time. Finally, colour discrimination, which is not found in all animals, is dependent on the discrimination of wavelength difference made possible by the absorbance properties of different visual pigments in the photoreceptors.

3.2.3 Visual Processing and Eye Design

When light reaches the photoreceptor pigments, the energy is transformed into cellular electrical signals in a process known as *transduction*. These signals are then propagated to other retinal cells, which perform a degree of image processing. However, ultimately the retina must send these resultant signals to the visual processing centres in the brain. This is accomplished by the exit of nerve fibres from the eye at the optic nerve head or disc. At this stage, various properties of the visual image (e.g., fine detail, colour, motion, etc.) have already been segregated into separate parallel channels that progress through several stages of processing before finally coming together in the highest visual areas of the brain to produce a coherent visual percept of the outside world (see ❯ Sect. 3.5.1).

The structures of the eye are often compared with some justification to the various components in a simple camera. Using this analogy, one can understand, for example, why a larger eye, which has a longer focal length and a correspondingly magnified image size, can produce a better resolution (i.e., higher quality) image. Similarly, a change in aperture size alters the intensity of light that falls on the light-sensitive surface (i.e., the retina), which is analogous to the light-sensitive array of a digital camera, in order to compensate for changes in

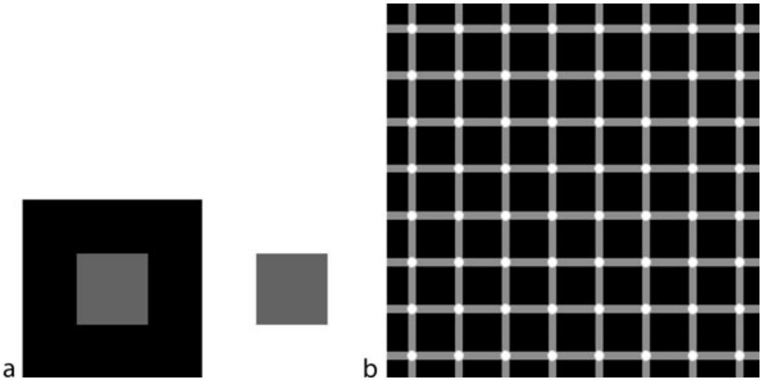


Fig. 3.5

Examples of visual illusions: (a) Simultaneous contrast. The central grey square has the same brightness, but appears different depending on the background. The reader can test this by masking the surrounds and comparing only the central grey areas. (b) The Hermann Grid. The observer sees illusory black dots at each of the intersections, other than the one directly fixated

light levels. However, this way of considering vision fails to take account of the need to *interpret* the image of the outside world. The simplest illustration of this is that the optical elements of the eye produce an inverted image, and yet we perceive the world as upright. This straightforward transposition is accomplished within the brain. At a higher level of complexity (but a lot more fun!), visual illusions clearly demonstrate the extent to which processing by the eye and/or the visual brain influences what we see (➜ *Fig. 3.5*).

No individual animal (including humans) has a limitless capacity for all aspects of vision. As with other parts of the body, there are always evolutionary and energetic constraints that influence the extant design and properties of the visual system. For example, a frontal eyed animal (like humans) has overlapping fields of view for the two eyes, thus providing the basis for binocular summation (i.e., improved light sensitivity) and depth perception (i.e., stereopsis). However, the disadvantage of frontal eyes is a more limited total field of view, thus making an animal more vulnerable to the approach of predators than are animals with more laterally directed eyes. Another example concerns wavelength sensitivity or colour vision, which is characteristic of diurnal animals that live under bright light conditions. One of the possible advantages of colour vision is to enhance the visibility of certain objects against their background (e.g., ripe red fruit hanging in a dark green tree). In contrast, nocturnal animals generally place a priority on light sensitivity rather than the ability to discriminate wavelength.

One of the interesting features of parallel processing in the visual system is that each property of the image is analysed by its own computing module. Presumably, such an arrangement is an efficient and rapid way of dealing with complicated stimuli; the greater the number and sophistication of these modules, the more comprehensive the visual perception. In some animals, aspects of the visual system can be tightly constrained to deal with a highly specific visual task or perception. One extreme example can be found in the toad's differential response to a short line stimulus, depending on its orientation with respect to its direction of motion. When such a stimulus is horizontal and moves, say, from left to right, the toad behaves as if the line were a worm, i.e., the frog approaches and tries to bite the object. However, when the same stimulus is vertical and moves left to right, the toad now reacts as if the line were a threat, i.e., the toad crouches down and puffs up its body. This response is completely hardwired and does not depend on experience. Yet, it clearly involves perception of different aspects of the stimulus, i.e., its orientation.

The toad's stereotypical response to certain visual stimuli is the exception rather than the rule. Most animals (e.g., many reptiles, birds and mammals) have a much more generalised visual perception which leads to visuomotor responses that are modifiable by experience. That is, the animal's vision is adaptable to a variety of different situations.

3.3 Machine Vision

Unlike animal vision, the major characteristic of machine vision is that it is not generalised, but rather is narrowly designed for a particular purpose or function. That purpose can be highly specific and may include, for example, the detection of flaws in a manufactured item, verification of various specifications (e.g., shape, colour, etc.), or controlling dynamic processes in production (e.g., filling a receptacle). Therefore, the design of the machine vision system will typically be highly constrained such that the parameters are narrowly tailored to the particular task in hand. The various functions, properties, and constraints of machine vision are covered in detail in the other chapters of this book.

3.4 Human Vision: Low Level Processing

3.4.1 Visual Acuity

In specifying the quality of vision, one of the most common descriptors is visual acuity: defined as the resolution of the visual system to discriminate fine detail. It is traditionally described in angular terms, i.e., the angular subtense of the detail just perceptible in the object. The precise value varies with the task. For example, for a normally sighted young adult discriminating the bars of a grating, the acuity limit is the gap between adjacent bars and is around 0.5 min of arc, whereas when discriminating and identifying a letter, the acuity limit is around 0.66 min of arc. At the level of the retina, the acuity will be dependent on the separation of photoreceptors, and as already described, this is at its minimum in the fovea (➋ *Fig. 3.6*).

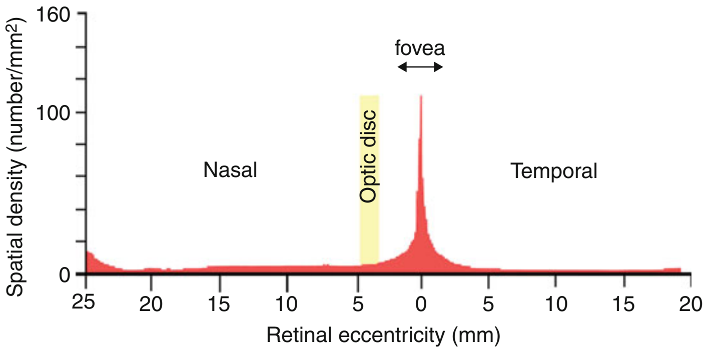


Fig. 3.6
The concentration of cone photoreceptors (in thousands) across the human retina

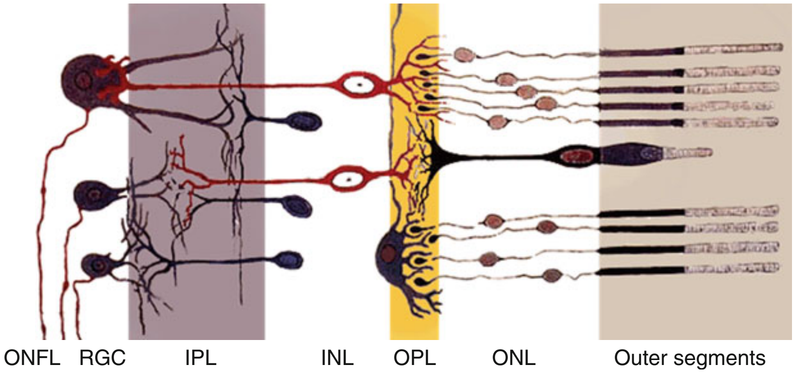


Fig. 3.7
Diagrammatic representation of cells within the retina and their linkages, including the outer segments of the rods and cones, the outer nuclear layer (ONL), the outer plexiform layer (OPL), the horizontal cells, amacrine cells and bipolar cells of the inner nuclear layer (INL), the inner plexiform layer (IPL) and the retinal ganglion cells (RGC) and their axons in the optic nerve fibre layer (ONFL)

In the fovea of the human eye, the separation of the photoreceptors (called cones from their cellular shape) is, on average, 0.5 µm and their width 1.5 µm. A calculation of resolution for receptors with this spacing yields 0.85 min of arc. This is similar to the average visual acuity measured in a young healthy adult, demonstrating that cone density is the primary factor determining visual acuity.

However, visual perception does not arise in the retina, and considerable processing takes place both within the retina and, more significantly, between the retina and higher brain centres. In particular, over most of the retina, there is not a one-to-one transmission of information, but pooling of signals from many photoreceptors to each output cell (a retinal ganglion cell) by virtue of connections amongst other cells within the retinal layers (➲ *Fig. 3.7*).

When the signals from several photoreceptors are pooled in this way, the critical factor at the retinal surface is not the receptor spacing *per se* but the extent of aggregation of receptors, which is referred to as the receptive field of the retinal ganglion cell. The larger the receptive field, the poorer the system's ability to distinguish adjacent images, i.e., the poorer the acuity. However, pooling information gives rise to larger overall signals and therefore to greater sensitivity to low light levels. This may also smooth out any 'noise' intrinsic to low light levels.

The trade-off between sensitivity and acuity is amply illustrated by the distribution of the two types of photoreceptors across the retina (cones and rods). Cones are used at bright light levels and their distribution mirrors the variation in acuity across the retina. The fact that resolution calculated from cone spacing matches measured acuity suggests that there is indeed one-to-one transmission from each cone to a single ganglion cell in the fovea, and this proves to be the case. On the other hand, rods are used at low light levels and have a distinctly different distribution across the retina.

There are no rods at the fovea (➲ *Fig. 3.8*). The reader can verify this, for his/her own eyes, by noting what vision is like at very low light levels (such as indoors at night with no lighting). If one looks directly at a small object (a number on a watch face, for example), the image will fall on the rod-free fovea and will disappear because cones do not function at this light level.

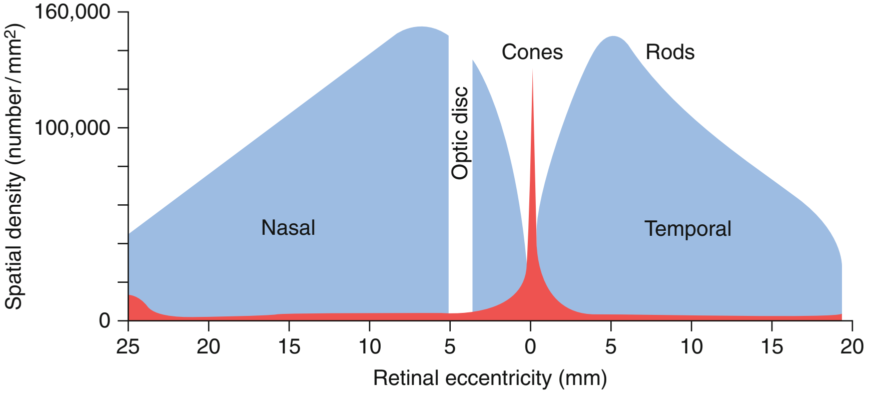


Fig. 3.8
The density of both rods and cones across the human retina

However, adjacent objects are still visible 'out of the corner of one's eye' because the retina adjacent to the fovea has a high density of rods.

As *Fig. 3.8* shows, the density of rods a few degrees away from the fovea is even higher than the density of cones at the fovea. If the only factor determining acuity were the receptor density, then our vision at or near the fovea would be much better at night than our foveal vision during the day. Yet, manifestly, it is not. Rods are connected together in very large numbers to a single output cell in order to increase the overall signal and therefore maximise sensitivity. In dim lighting, when rods dominate, visual acuity is sacrificed for sensitivity. In bright light, sensitivity is not an issue since there is ample light for the now dominant cones to operate, and the system can concentrate on maximising acuity.

3.4.2 Sensitivity and Adaptation

The human (and animal) visual system is more adaptable than the above description suggests. Rather than a simple two-state system, utilising cones and rods, the visual system is capable of operating over a very wide (10^{10} fold) range of light levels. (This is above the dynamic range of current imaging technologies.) The receptive fields can vary in size for differing light intensities, by virtue of the interconnections within the retinal layers. The brighter the light, the smaller the receptive fields (acuity winning over sensitivity) and the lower the light, the larger the receptive fields (sensitivity dominating over acuity). Thus, the vertebrate visual system can operate continuously over a very wide range of light levels.

The process of vision is initiated by the biochemical changes in pigment within the photoreceptor that generate the electrical response. All of the pigments found in the different photoreceptors, when extracted and viewed under dim lighting, are coloured. The rod pigment, for example, is sometimes called 'visual purple'. When activated by light, pigments become colourless. The process of pigment activation is therefore referred to as *bleaching*. In order for transduction to electrical energy to happen again, the pigment must return to its former state, or *regenerate*. After exposure to light, regeneration takes a finite time, and explains why the visual system, although capable of operating under widely different lighting levels, needs time to adapt when light intensity changes. When leaving a brightly lit foyer and entering a dark cinema, at first it is a struggle to find one's seat. Immediately after the sudden drop in light intensity, the retina is very insensitive, and vision is correspondingly poor. The pupil immediately dilates to allow more light in and, more slowly, the pigments regenerate. The interconnections within the retina become more active and sensitivity increases. These processes all contribute to dark adaptation. In time, the cinema auditorium appears more easily negotiable. The reverse process, light adaptation, is much quicker, involving pigment bleaching rather than regeneration. On leaving the cinema and entering daylight, the very sensitive rod-based system briefly gives discomfort, but after a few moments, sensitivity lowers. Pupillary constriction helps as well by reducing the light reaching the retina.

3.4.3 Spatial Contrast

Visual life is not all about detail. Indeed, so few tasks in the human daily round are conducted at or even near the acuity limit that a measurement of visual acuity can arguably be said to be irrelevant to establishing quality of vision. Of much greater significance is the ability to detect contrasts in the visual scene. This is the difference in brightness between objects or between figure and ground, leaving aside colour contrasts for the time being. This ability is called *contrast sensitivity* and is formally recorded as the contrast between adjacent bars of a sinusoidal grating that can just be discriminated, across a wide range of bar widths or spatial frequencies (➋ *Fig. 3.9*).

The illustration below (➋ *Fig. 3.10*) shows that sensitivity is maximal for gratings of medium frequency. The plot of contrast sensitivity against spatial frequency (*the contrast sensitivity function*) reveals that, in humans, maximum contrast sensitivity occurs at a frequency of 2–5 cycles per degree. This corresponds to a single bar width of 15–6 min of arc. As the frequency increases (i.e., bars become narrower), contrast sensitivity falls, and thus higher contrast is needed. The narrowest bar (highest frequency) detected at 100% contrast corresponds to visual acuity. The contrast sensitivity function (the shaded area in ➋ *Fig. 3.10*) thus represents the 'window on the world' as it encompasses the size and contrast of all objects that are visible. Objects outside this window are too small or too low in contrast to be seen.

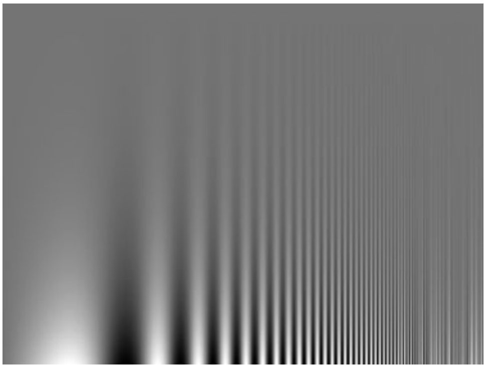


Fig. 3.9
Variation in contrast and grating width demonstrates the reader's own contrast sensitivity function

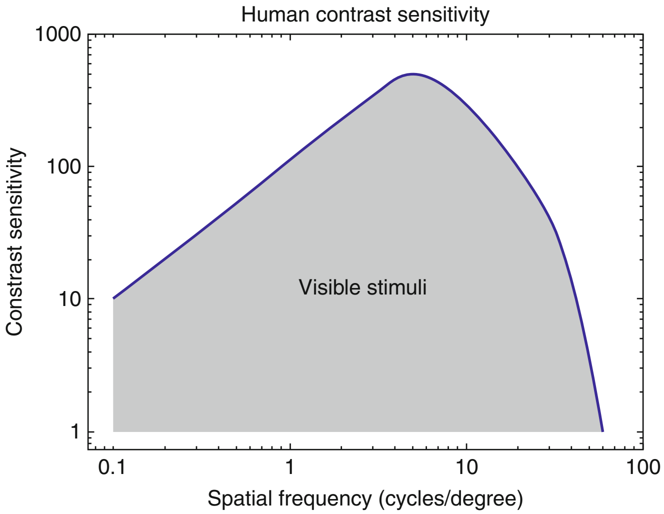


Fig. 3.10
Average human contrast sensitivity function

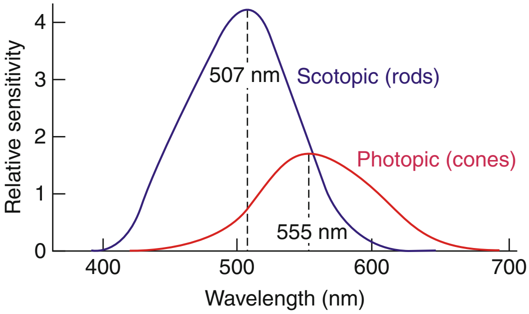


Fig. 3.11
Sensitivity to different wavelengths for both rods and cones in the human eye

3.4.4 Spectral Response

Just as there are constraints on the human visual system in terms of size and contrast of objects, so there are limits in terms of the wavelength of light to which the visual system can respond. The visible spectrum is so called because it encompasses wavelengths that are visible to the human eye (➋ *Fig. 3.1*). Many insects and some birds are sensitive to wavelengths, particularly in the ultraviolet, that are completely invisible to the human eye; thus, the visible spectrum of a bee or a bird can be quite different to that of a human.

Light that lies beyond the visible spectrum of a given animal does not evoke a response in either its rods or cones and therefore, does not initiate vision. Further, the visual system is not equally sensitive to all wavelengths within the spectrum. As ➋ *Fig. 3.11* illustrates in the human eye, the overall cone response (under bright lighting or photopic conditions) is

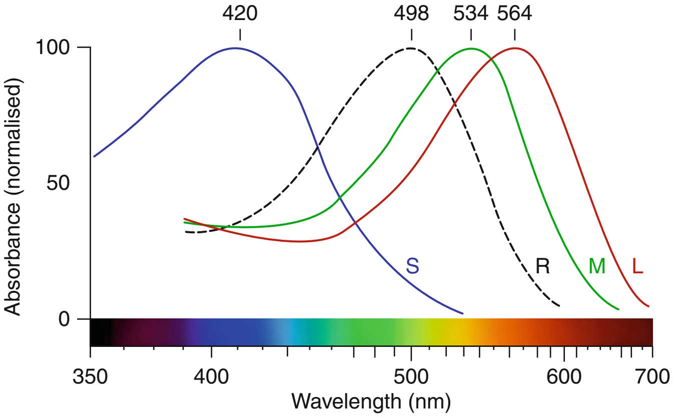


Fig. 3.12

Sensitivity to different wavelengths for rods and the three types of cones in the human eye (S = short [*blue*], M = medium [*green*], L = long [*red*] wavelength, R = rods)

maximal to light around 550 nm. Rods, which operate under dim illumination or scotopic conditions, have their peak sensitivity at 500 nm. Thus, the apparent brightness of a light source or an object is not simply a function of the intensity of light reaching the eye from the object, but also of its wavelength. For example, when cones are operating, yellow light (or light reflected from a yellow object) will appear much brighter to a person than red light of equal intensity.

The difference in peak wavelength sensitivity between cones and rods can be appreciated by an apparent colour shift when light is dimming. The reader can verify this by sitting in a flower garden as the light dims at twilight and noting the apparent blueness of the flowers becoming exaggerated. This occurs because rods are more sensitive to the short (blue) wavelengths than are cones.

Colour discrimination, or the ability to detect differences in wavelengths of light, is a property only of cone vision. It is possible in humans by virtue of the retina containing three distinct types of cones: cones maximally sensitive to long, medium, and short wavelengths, often referred to as ‘red’, ‘green’ and ‘blue’ cones, respectively (➋ *Fig. 3.12*).

The responses of the three cone types are kept separate as they are transmitted in parallel through the visual system. Differences in wavelength are ultimately processed in the brain by comparing the responses of the three (relatively) independent colour systems. Thus, for example, a wavelength of 625 nm initiates a particular ratio of response in long and medium wavelength cones that ultimately gives rise to a sensation that we normally call ‘yellow’. The sensation of ‘white’ arises when responses from all three cone types are approximately equal. Humans, however, possess only one type of rod, and therefore, wavelength discrimination or colour vision is not possible under scotopic conditions.

3.4.5 Accommodation

The values given thus far in specifying human vision (e.g., visual acuity limit of 0.8 min of arc) are dependent on a good quality image in focus on the retina. The main refractive component of the eye is the transparent cornea, but its focussing power is fixed. An active focusing

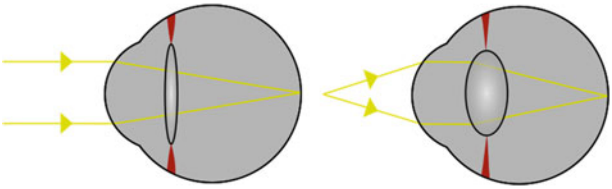


Fig. 3.13

Changes in the shape, and therefore power, of the lens allow the human eye to change focus for objects at different distances

mechanism within the eye is essential for maintaining a good quality image when the object of regard changes distance from the observer. In humans and many other vertebrates, this is attained by virtue of the flexible crystalline lens, which under the action of intra-ocular muscles changes shape and therefore its refractive power to adjust focus (➋ *Fig. 3.13*). This is the process of *accommodation*. For a distant object to be focused on the retina, the lens must adopt minimum power (i.e., its thinnest or least convex shape). Conversely, for a close object to be focused, the lens must adopt maximum power (i.e., its thickest or most convex shape).

The range of accommodation in a young adult human is about 8 cm to infinity. In fact, the ability to focus on close objects is highly dependent on age and begins to deteriorate almost from birth. A young child can focus on objects at 5 cm from the eyes, whereas a 40-year old will struggle to focus at 20 cm. This deterioration, known as *presbyopia*, is due to increasing rigidity of the lens, which becomes completely inflexible in old age. This is the reason why middle age is typically accompanied by the need to wear reading glasses or bifocals.

In order to obtain maximum visual information about any object (in daylight conditions), the fovea is normally used for fixating and inspecting. In addition to acuity and optical quality, good vision depends on the ability to maintain a stable gaze. To achieve this, head and/or eye movements must often be employed to direct the fovea as required. There are several types of these visuomotor movements and the kind used depends on the situation. For example, looking around at different stationary objects, e.g., reading a line of text, employs jerky eye movements (*saccades*) that are extremely rapid and precise, whereas visually tracking a slowly moving object requires fine continuous control of position as the eye smoothly changes its orientation to match the position of the object. The purpose of such ‘pursuit’ is to keep the object’s image stable and on the fovea at all times. In addition, as the head and body move through an environment (or the environment moves by), optomotor reflexes of the eyes are used to compensate for this movement, in order to maintain a stable view of the visual scene. All of these movements involve the two eyes moving together in the same direction. However, as the distance to an object changes, the eyes will alter their orientation, often in different directions, to minimise the binocular disparity of the image. This is called *convergence*. The brain exploits the remaining, inescapable disparity (due to the separation of the two eyes) to calculate the distance to a reasonably near object, while at the same time providing binocular depth awareness.

3.5 Human Vision: Higher Level Processing

The eye’s output fibres that form the optic nerve (➋ *Fig. 3.14*) eventually project to a relay nucleus in the brain where the different types of visual information are passed along for their

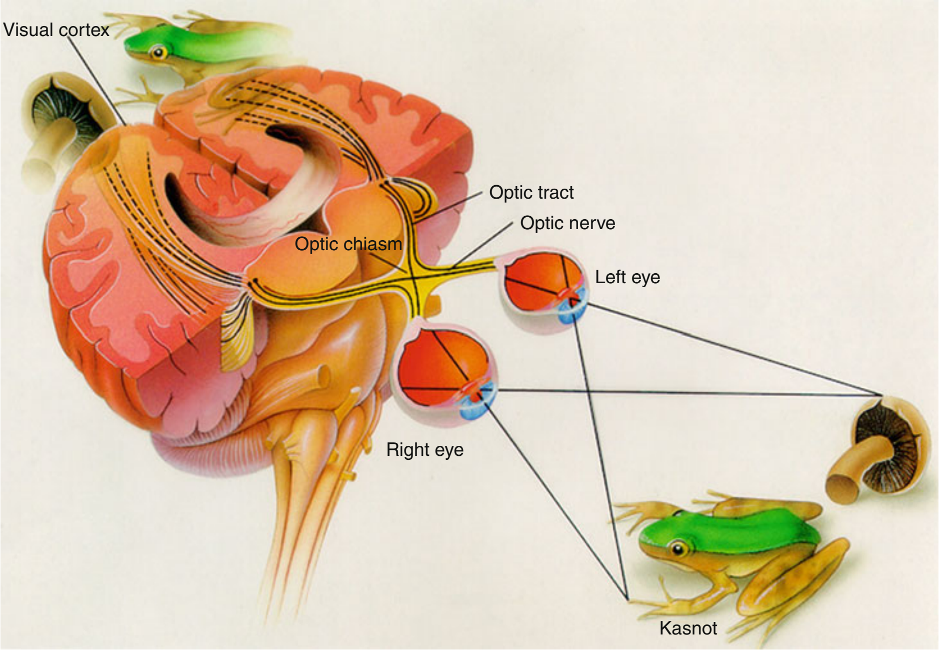


Fig. 3.14
The visual pathways in the human brain

initial processing in the primary visual cortex. At this point, the visual information from the right and left visual fields (hemifields) is proceeding to the left and right sides of the brain, respectively.

3.5.1 Parallel Processing

As already described for colour vision, numerous properties of the visual image are transmitted separately, and in parallel, to the primary visual cortex where certain basic features (e.g., orientation, size and contrast) are now extracted. As the information leaves the primary visual cortex, this parallel organisation of visual information is maintained, but now each type of information is sent to a different, largely independent higher cortical centre. Thus, the visual system at this level has not only a parallel but also a modular organisation. For example, there are separate higher cortical areas concerned with motion, colour, or object recognition rather than the visual image as a whole. One of the intriguing consequences of this modular arrangement is that, if a person suffers localised damage to the brain, involving one of the higher visual centres, the deficit can be very specific in nature. Neurological case studies of individual patients describe, for example, the inability to perceive either motion or colour. Other patients may be unable to recognise familiar objects, or even more specifically, faces. In many of these patients, all other aspects of their visual perception appear to be normal.

Conscious perception of a visual scene with all of the various properties reintegrated only occurs later, at higher levels of the visual system (➜ *Fig. 3.15*). At that stage, the various

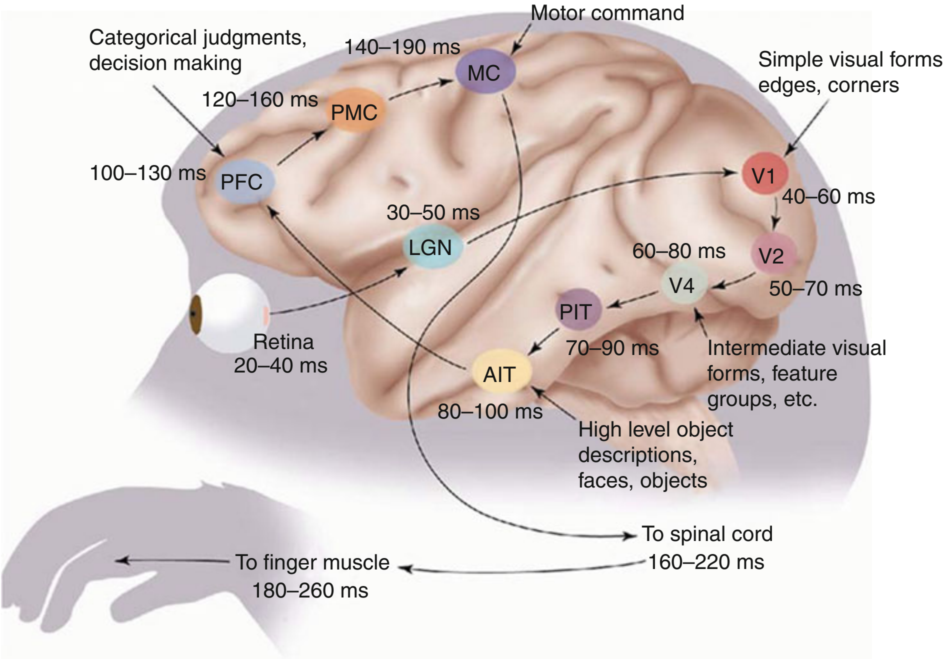
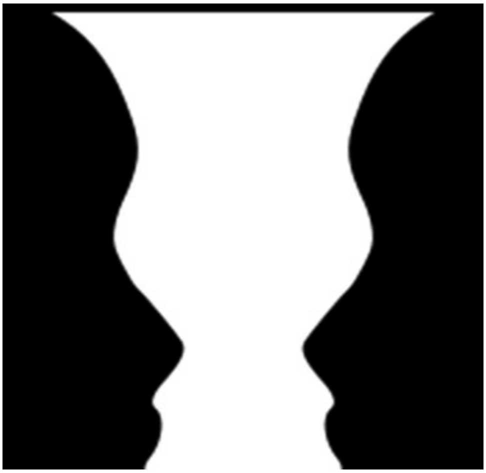


Fig. 3.15
The visual and motor pathways in the human brain that allow object recognition and control motor responses. Light-induced signals travel from the retina in the eye to the lateral geniculate nucleus (LGN) and then via the primary visual cortex (V1) on to higher visual centres (V2, V4, etc.). Recognition of the object of interest occurs in areas of the inferotemporal cortex (PIT and AIT). A motor response originates in prefrontal cortex (PFC) and is then transmitted via premotor cortex (PMC) to the primary motor cortex (MC), which produces the response in skeletal muscles via the spinal cord

components of the scene, each with their attendant characteristics of form, colour etc., are recognised and understood. A complete interpretation of the visual world involves not only identifying different objects and their spatial relationship to each other but also their significance. This can relate to the need to navigate through the world, whether there is a threat, the availability of food and so on. The context in which an object is viewed can affect the meaning assigned to it. Thus, an apple can be aesthetically attractive, evoke feelings of hunger or cause one to consider Newtonian equations of gravity. This demonstrates that visual perception is, by and large, not hard-wired and illustrates the inherent flexibility of the visual system.

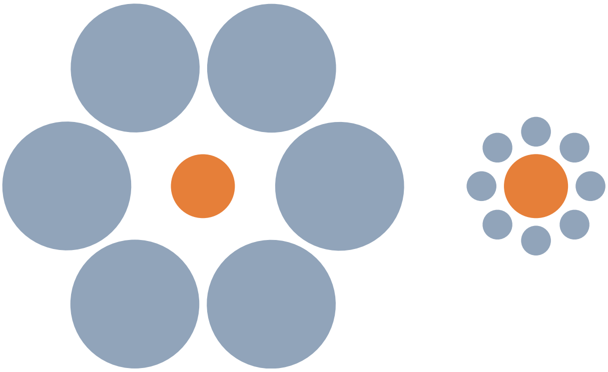
3.5.2 Visual Illusions

However, there are clear manifestations of the visual system making, in essence, assumptions about the world and this can lead to characteristic misinterpretations of visual scenes, i.e., visual illusions. Some of these illusions depend on ambiguity (see *Fig. 3.16*) whereas others appear to suggest that the brain has built-in models, spatial or otherwise, of how the world works (*Figs. 3.17* and *3.18*).



■ Fig. 3.16

An example of an ambiguous figure. Is it a *white* vase against a *black* background or two *black* faces against a *white* background?



■ Fig. 3.17

An illusion of size. The central *orange circles* are equal in size, but their different surrounds create apparent differences in size

In either case, the inability to overcome or ignore a visual illusion strongly suggests that at least some of the visual system has hard-wired circuitry that is analogous to the frog's differential response to horizontal and vertical line stimuli. Interestingly, many animals fall victim to the same visual illusions indicating that the built-in models within the visual system operate similarly across species.

3.6 Animal Vision

Presumably all animals, including humans, must face and solve the same challenges in viewing the world. Rather than limiting one's analysis to human vision alone, it is instructive to

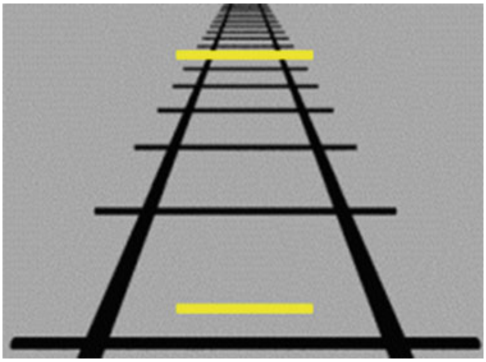


Fig. 3.18

An example of an illusion based on perspective. Both *yellow lines* are of the same length but the apparent distances in the drawing, created by perspective, distort their apparent size

consider the alternative solutions that evolution has produced in other species as well as those approaches that are held in common. Potentially, this knowledge tells us a great deal about the structure and functions of the visual system and may offer strategies for designing machine vision systems.

3.6.1 Visual Acuity

When one considers the range of animals living today, the importance of vision in their everyday lives varies considerably. Indeed, in many species, vision is not the dominant sense for perceiving the world; for example, many animals depend instead on an acute sense of smell or hearing. As a consequence, the ability to resolve detail (visual acuity) can range from a crude distinction between light and dark (e.g., the mole) all the way to acuities that exceed that found in the human eye (e.g., the eagle). Presumably, this relates to the requirements of an animal's lifestyle, and depends ultimately on the evolutionary history that led to the current species. The level of visual acuity is the product of both the quality of the eye's optics and the 'grain' of the photoreceptor system. In the case of a low acuity retina, the photoreceptors may either be more widely separated or may pool their signals into a single output cell rather than having their own dedicated channel to the brain. As mentioned when discussing the density of rods in the human retina, this has the consequence of increasing sensitivity at the expense of resolution.

In the human eye, the fovea represents an adaptation to enhance the quality of the image that reaches the photoreceptors in a region of the retina where the cone density is at its maximum. This region is used preferentially for the fixation and detailed examination of objects of interest. The concentration of photoreceptors does not remain constant across the retina; there is a precipitous decline in the density of cones as one moves away from the fovea (➜ *Fig. 3.6*). One obvious question that arises is why don't the eyes of animals, including humans, maintain a constant high photoreceptor density across the entire retina and thus allow good acuity vision in all parts of the visual field? Presumably, this reflects genuine energetic and structural constraints. These can include a limit on the diameter of the optic nerve that penetrates the retina and emerges from the back of the eye. A larger optic nerve would produce a correspondingly larger blind spot (i.e., absence of photoreceptors). In addition, higher cell

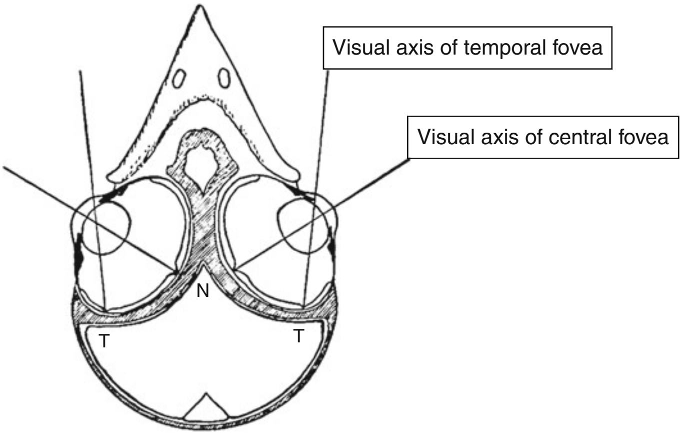


Fig. 3.19

Cross-section of a bird's eyes showing the line of sight of both foveas in each eye

density would increase the demand for attendant neural circuitry within both the eye and the brain.

Some animals have neither an apparent fovea, nor even a differential distribution of photoreceptors, and possess a relatively low acuity (e.g., rabbit), whereas others exhibit a relatively high density area in the central retina, with no foveal depression (e.g., cat and dog). Other than primates, foveas are not found in mammals but they are quite widespread in reptiles and birds. Indeed the central fovea in these species is extremely well defined (➔ *Fig. 3.4*), and in a number of birds the eyes have two foveas (➔ *Fig. 3.19*). Moreover, in most species of bird, the retina is dominated by cones, and interestingly, their density does not decline away from the fovea to the same degree as in human. This suggests that their peripheral acuity is correspondingly higher. Presumably this reflects the overwhelmingly diurnal (daytime) nature of most birds and therefore results in a relative scarcity of rods. Nocturnal birds, such as owls, have a higher proportion of rods to increase sensitivity to low light levels.

3.6.2 Light and Wavelength Sensitivity

The relative preponderance of cones and rods generally relates to an animal's lifestyle. Thus, the retina of a diurnal species that lives in a photopic environment has a relatively high proportion of cones (e.g., reptiles and birds), or even exclusively cones (e.g., ground squirrel). By contrast, the retina of a nocturnal species (e.g., rat), which is largely confined to a scotopic environment, will be dominated by rods. Those animals that live in both light regimes (e.g., humans) or those that are active primarily during dawn and dusk (e.g., fox) have mixed cone-rod retinae. One interesting consequence of having a mixed retina is that there are unavoidable time delays when moving from photopic to scotopic light levels to achieve dark adaptation. Another involves changes in the size and shape of the eye's aperture or pupil. In fact, whereas the pupil in humans and many other animals is circular, or nearly so, many species have quite different shapes. These appear in many cases to be adaptations to the different light conditions in which they live. The most familiar example of this is the cat's pupil, which is clearly a vertical slit in bright light conditions but much more rounded in the dark. Although outside the scope of this chapter, the reader is encouraged to look at the sheer diversity of shapes, sizes and orientations of animal pupils as this has intriguing implications for the design of optical systems.

As mentioned earlier, the visible spectrum for different animals can vary significantly. This, too, reflects the ecological environment as well as lifestyle requirements. On the one hand, two species living in the same place may have very different wavelength sensitivity. For example, the honeybee is sensitive to the ultraviolet patterning of flowers (but, interestingly, is relatively insensitive to the colour red) whereas the visible spectrum of most diurnal mammals does not extend into the UV. Their lenses filter out UV light, presumably to protect the retina against damage. Some nocturnal mammals (e.g., bats, rats) can perceive UV light, and their lenses are transparent to UV, as this protection is not required.

Moreover, marked differences in the environment are correlated with equally dramatic alterations in the wavelength sensitivity in closely related animals. Fish that live deeper in the ocean have a visible spectrum that is shifted toward short wavelengths (blue) when compared with more shallow living fish. Within the lifespan of a single animal, changes in the environment can result in an adaptive change to their visual system in response. This is illustrated by changes to the photopigment found in the eye of the European eel, which undergoes several metamorphoses involving a change in lifestyle from river dwelling to swimming in the depths of the sea. The significant difference in the characteristics of the light experienced at each stage brings about a shift in the properties of the pigments and therefore in the visible spectrum for this animal.

For all animals, wavelength sensitivity ultimately depends on the absorbance properties of the visual pigments in the photoreceptors. Colour vision, on the other hand, requires the interaction between two or more cone pigments in the retina. Many mammals have only two cone pigments (dichromats); primates often have a trichromatic system, and many birds have an additional UV or near-UV sensitive pigment (➋ *Fig. 3.20*). In the extreme case, the mantis

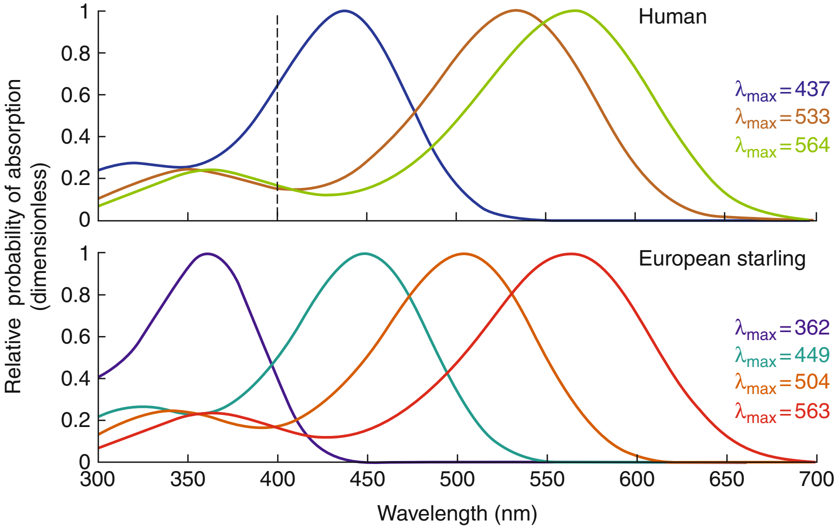


Fig. 3.20

Colour vision in humans is mediated by three pigments (trichromatic vision) and in many birds by four (tetachromatic vision)

shrimp has 12 distinct cone pigments. Colour vision based on a discrete number of pigments can involve the perception of not only primary colours, but also intermediate hues. For example, colour vision of bees appears in that sense to be similar to that in humans. On the other hand, in some animals, highly specific behaviour can be triggered by particular wavelengths only. For example, certain species of moth have feeding and egg-laying behaviours in response to quite precise wavelengths of light.

Presumably, the larger the number of pigments, the better the ability to distinguish subtle changes in hue. Measuring colour vision in an animal like the shrimp presents its own special challenges! The precise location of the absorbance peak of each pigment and the separation between these peaks will have significant consequences for the quality of colour vision. This has interesting implications for visual perception. For example, the 'blue' pigment of one animal may be sensitive to a different wavelength from the corresponding pigment of another animal. It is perhaps impossible to know whether two such animals will see any given visual scene in the same way. Predicting exactly how the separation between the absorbance peaks of various pigments will affect colour vision is equally problematic. It is interesting to note that the difference between the 'green' and the 'red' peaks in birds is much larger than that found in humans.

Finally, it is worth considering the functional significance, or adaptive advantage, of colour vision. Colour vision can be defined as the capacity to distinguish different wavelengths of light irrespective of their intensity. Current opinion suggests that a major purpose of colour vision is to enhance the contrast of certain objects against the predominant colour of their surroundings. Thus, primates, which originated in arboreal environments, have a green-centred colour system, in order to help distinguish objects of interest against the prevailing background. A second important function is to provide information to help in identification of objects (e.g., recognising specific fruits and/or their degree of ripeness).

3.6.3 Vision at Different Distances

One obvious property of the visual world is that objects lie at different distances from the eye. Therefore, for consistently good vision, the eye must have the ability to alter its refractive state so as to accurately focus the image on the retina, whatever the distance to the object. As pointed out earlier, the lens is the component of the eye used to achieve this change in focal length. In humans and many other animals, the mechanism of accommodation involves a change in shape and thus, radius of curvature of the lens. However, some animals take a very different approach. For example, instead of changing its shape, the lens may move toward or away from the cornea, in a manner analogous to a zoom camera lens. This is a common mechanism in teleost fish (e.g., mackerel or bass), but is also found in amphibians. One advantage of this focusing mechanism is that, because the shape of the lens does not change, there is no increase in spherical aberration during accommodation (e.g., a camera telephoto lens). In diving birds, the normal corneal refractive power is all but lost when the eyes are underwater, and therefore, the lens must be used to compensate while diving. In the cormorant, for example, the accommodating lens thickens and is actually pressed against the back of the iris. The middle portion has even been reported to protrude through the pupil, producing a much smaller radius of curvature in the central lens and a correspondingly greater focussing power. Although one normally thinks of the cornea as a static refracting surface, there are some reports that a number of bird species (e.g., chicken, pigeon) actually change the curvature of their corneas when focussing on near objects. The relative contribution of such a mechanism, compared to that of the lens, remains to be determined. As opposed to using dynamic changes in refracting surfaces like the cornea and lens, other solutions to the problem of viewing objects at different distances are also possible. For example, analogous to humans wearing varifocal spectacle lenses, some ground foraging birds possess an eye that simultaneously focuses distant objects on to the central retina and nearby objects (on the ground) on the upper retina. This so-called lower visual field myopia appears to be a fixed (possibly corneal) adaptation. In the same way that the visual spectrum can vary across species, the range of accommodation amplitude can also differ. This presumably relates to the visual requirements of the animal's lifestyle.

3.6.4 Visuomotor Behaviour

As mentioned previously, the main functions of eye and head movements in humans are to direct the central retina to objects of interest and/or to stabilise gaze to compensate for movements of the body or objects in the visual world. Interestingly, all of the same types of eye movements (i.e.,, saccades, pursuit, vergence, optomotor reflexes, etc.) are found in virtually every animal, not just vertebrates. In this context, 'eye movements' are not restricted to movements of the eyes in the head, but also encompass redirections of gaze accomplished by head and/or body movements. At one extreme, the eyes of the housefly cannot move, so the head or entire body is used to accomplish these optomotor movements. In a given animal, the extent to which the eyes can move in the head will presumably depend on a variety of evolutionary and/or ecological factors. For example, in the case of birds, the eyes are disproportionately large (outweighing the brain in many species), and eye movements are relatively restricted. This results in the characteristic rapid head movements that can be readily observed in birds.

3.6.5 Eye Orientation

When considering the issue of the number of eyes possessed by animals, it is intriguing to consider why, in almost all species, there is a minimum of two. One obvious advantage of two eyes is the possibility of stereopsis, or 3D vision. However, this capability is actually quite rare in the animal kingdom, suggesting that stereopsis is a possible adaptation arising from, but not a cause of, having two eyes. Perhaps, more simply, the occurrence of two eyes is the result of an imperative for symmetry in development. Whatever the reason, the orientation of the two eyes in the head varies considerably in vertebrates. There are frontal eyed animals (e.g., owls), lateral eyed animals (e.g., rabbits), and animals with the two eyes on the top of the head (e.g., alligators) or even with both eyes on the same side of the head (e.g., plaice or flounder). The resting orientation of the two eyes defines the overall visual field of the animal. So, for example, binocular animals have a relatively small total visual field with a large 'blind area' behind. Animals like the rabbit or pigeon, however, have extensive fields with little overlap between the two eyes that enable them to keep watching almost all the way around them (see ➋ *Fig. 3.3*). On the one hand, eye and/or head movements can be used to extend the world visible to the animal. In humans and many other vertebrates, the eyes move together, but in the case of some animals, the eyes are also capable being moved independently of one another. A particularly striking example of this can be found in the chameleon. On the other hand, eye movements like convergence can change the extent of the overlap between the visual fields of the two eyes. Although one normally thinks of convergence in connection with fine-tuning binocular disparity to provide for depth perception, many animals do not appear to have stereopsis. Another possible advantage of a greater overlap between the monocular visual fields is that combining the input from both eyes improves visual acuity and contrast sensitivity as well as increases light sensitivity. This improvement in vision is known as *binocular summation.*

A number of the issues we have already considered can interact in ways that provide further insight into the organisation of the visual system as well the specific behaviours of animals. An interesting example can be found in the rather comical head bobbing made by pigeons and other birds as they walk along the ground. This appears to result from the interaction of eye orientation in the head, the requirement for a stable gaze when viewing the world, and a curious higher order feature of the visual system known as *saccadic suppression*, which turns out to occur in virtually all vertebrates, including humans (see below). Pigeons have eyes that are orientated very laterally in the head. This necessarily means that, when a pigeon walks forward, its largely monocular view of the world lurches backward. Without some sort of compensation for this movement, this would result in 'smearing' of the image across the retina. This is similar to what happens when a video camera pans too quickly across a scene. There is no straightforward eye movement that can cancel this unwanted effect (➋ *Fig. 3.21*). Instead, the bird temporarily adopts a stationary head position while walking so that it can have a stable view of the world, until the head must 'bob' forward to a new steady position to catch up with its body. Interestingly, during the head bob, the pigeon actually sees little or nothing because of saccadic suppression. When the eye makes a saccadic movement (either by itself or as a result of

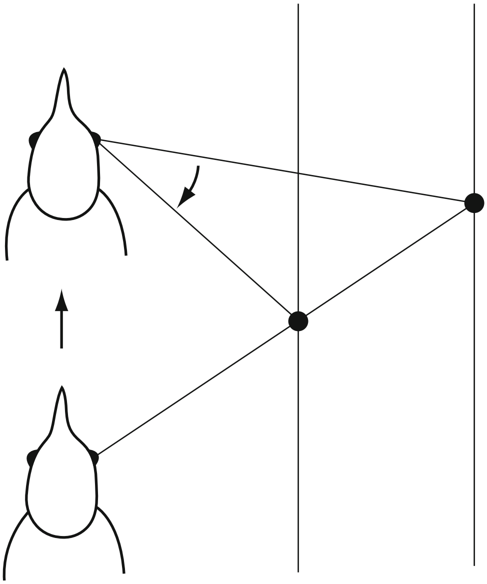


Fig. 3.21

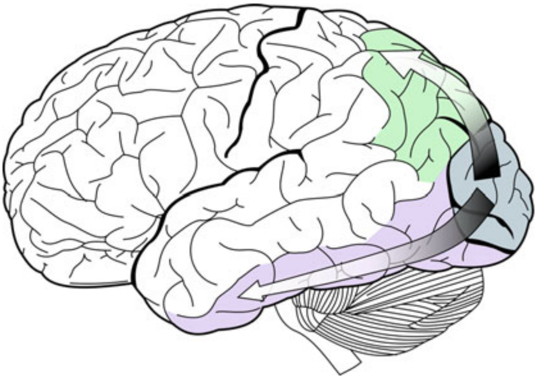
Forward translation of an animal with lateral eyes leads to different shifts in the retinal position of objects at different distances

a saccadic movement of the head), the visual system actually 'turns off' or suppresses the input from the retina to the higher visual centres in the brain. Demonstrate this to yourself by simply looking into a mirror and trying to see your eyes move without moving your head. Thus, the pigeon, while walking (i.e., a saccadic motion), must periodically stabilise its head in order to see anything at all. This sort of visual sampling is not necessary for humans while walking because our frontal eyes allow us to keep our gaze steadily on the world in front of us, thus avoiding any smearing of the central retinal image. However, when making saccadic eye movements, e.g., during reading, humans must also in effect 'sample' the visual world.

3.6.6 Visual Pathways

In all vertebrates, the basic design of the retina is reasonably similar. The transduced signals are then transmitted to higher centres in the brain. However, the pathways used for processing this visual information vary considerably across species, with different characteristic plans being found in each class (i.e., mammals, birds, reptiles, etc.). Although the exact details of brain anatomy are outside the scope of this chapter, a brief consideration of certain principles underlying the organisation of these pathways is relevant. The principal characteristic of visual processing in humans and other primates is the separation of different aspects of the visual stimulus into discrete parallel streams. For example, visual processing in higher brain centres divides into a dorsal stream largely concerned with motion and location and a ventral stream that deals with object recognition and colour (➋ *Fig. 3.22*). As described in ➋ Sect. 3.5.1, isolated brain lesions in humans can sometimes lead to highly specific visual deficits. The extent to which this degree of parallel and/or modular processing also exists in other animals is far from clear and remains to be determined.

All vertebrates that are capable of visual processes, such as recognising objects or assigning significance to what is being seen, must ultimately depend on more complex brain circuitry. Looking beyond reflexive stimulus–response actions, such as that described earlier for the frog, a good example of higher visual processing is the perception (or is it misperception?) of visual



◘ Fig. 3.22
Schematic illustration of the dorsal (*green*: motion and location) and ventral (*purple*: detail, colour and object recognition) streams of visual processing in the cerebral cortex

illusions. The fact that, like humans, some animals experience visual illusions is certainly evidence for a degree of pre-programming or hard-wired assumptions made by the brain. Moreover, the evidence that some illusions are perceived in the same way suggests further that this circuitry has a similar organisation. This would seem to be a fruitful approach for further investigation of higher visual processing in different animals.

3.7 Comparing Animal and Machine Vision

Inevitably, this brief stroll through the properties of human and animal vision can only highlight some of the basic principles of eye morphology, brain organisation, visuomotor behaviour, and perception. At first glance, one is struck by the enormous diversity encountered in all aspects of biological vision, including – but not limited to – eye shape and size, retinal properties, eye orientation in the head, etc. Almost certainly, this diversity reflects an underlying variation in the environments within which each of these animals is trying to survive. In this limited sense, animal and machine vision can share a common goal: the development of the 'equipment' necessary to solve a particular visual problem or task. As different animals often face very different challenges to which they must adapt, the solutions that arise in the course of evolution vary considerably. A careful examination of such adaptations may well provide further valuable insights for future work by computer vision scientists, which eventually benefits machine vision. The corollary of this view is that at least some aspects of animal vision either get their start or are subsequently shaped by the need to achieve quite specific, often relatively simple goals, just as occurs in some machine vision applications. In addition, constraints and limitations, including cost-effectiveness or efficiency, are just as important for animal vision as for machine vision. For example, in the course of evolution, evolution will presumably only 'invest' in the processing power that animals require for the purpose.

However, this is clearly not the whole picture, as survival for animals rarely depends on the ability to deal with a single problem: survival is a multi-function task! This fact alone will account for the higher complexity of animal vision, which in turn produces its arguably most important feature: *emergent properties*, which can be thought of as incidental benefits. Vision, perhaps originally designed for one, or just a few limited tasks, can lead to new abilities and opportunities simply because of the additional perception that is made possible. A good example of this is the advent of colour vision in some animals. Many have argued that wavelength sensitivity arose in primates originally to allow them to detect riper fruits against a different colour background. However, chromatic sensitivity leads to the solution of many other perceptual problems. In turn, species with colour vision can then exploit colour in their social communication, as evidenced in the colourful plumage of many birds.

One last thing to consider is the nature of perception itself. In the context of machine vision, the equipment is carefully designed to register certain objects or scenes and then make one or more discriminations. In contrast, animals generally must have a much more flexible visual system, possessing the ability to make many kinds of discriminations in a range of environmental and lighting conditions. Nonetheless, just as a machine can only take in the visual stimuli it is designed to receive, humans and animals do not actually 'see' the real world; we perceive what each of our visual systems allows us to experience. Animals may have just as valid a percept as we do, but it can be very different. Similarly, machines may 'see' the world differently from the way we do, but we must not suppose that our view is necessarily 'right' or indeed 'better'.

Further Reading

Bach M, 88 visual phenomena & optical illusions http://www.michaelbach.de/ot/. Accessed 18 Aug 2010

Douglas RH (2010) Vision: vertebrates. In: Breed MD, Moore J (eds) Encyclopedia of animal behavior, vol 3, Academic, Oxford, pp 525–543

Foley HJ, Matlin MW, Sensation and perception: the visual system. http://www.skidmore.edu/~hfoley/Perc3.htm. Accessed 18 Aug 2010

Hubel D, Eye, brain, and vision. http://hubel.med.harvard.edu/. Accessed 18 Aug 2010

Kolb H et al, WebVision: the organization of the retina and visual system. http://webvision.med.utah.edu/. Accessed 18 Aug 2010

Land MF, Nilsson D-E (2002) Animal eyes. Oxford University Press, Oxford

Parker A (2003) In the blink of an eye. Free Press, London

Polyak S (1957) The vertebrate visual system. University of Chicago Press, Chicago

Rochon-Duvigneaud A (1943) Les yeux et la vision des vertébrés. Masson, Paris

Tovée M (2008) An introduction to the visual system. Cambridge University Press, Cambridge

Vilis T, The physiology of the senses: transformations for perception and action. http://www.physpharm.fmd.uwo.ca/undergrad/sensesweb/. Accessed 26 Oct 2009

Viperlib. http://viperlib.york.ac.uk/. Accessed 18 Aug 2010

Wade N, Swanston M (2001) Visual perception: an introduction. Psychology Press, Hove

Walls GL (1967) The vertebrate eye and its adaptive radiations. Hafner, New York

4 Colour Vision

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_4,

4.1 Introduction

The sensation of colour arises as a result of the brain's response to variations in the wavelength of light received by the eye. It provides the brain with signals representing four different views of a scene. (Rod cells and three different types of cone cells provide these signals in the eye.) Hence, we can regard the eye-brain complex as a system that generates four precisely aligned images. These are then compared/combined with each other within the brain but the way in which this is done remains enigmatic. There are many subtle features of human colour vision but we shall argue that many of them need not concern us, as our objectives in Machine Vision are quite specific.

The scientific study of colour (*Colour Science*) is a complex subject; it is concerned with the perception of colour by the human/animal eye, the origin of colour in materials, colour in art and fashion, photography, printing and video, as well as the physics of light. Colour Science encompasses many technical disciplines, including physiology, anatomy, psychology, philosophy, chemistry, optics, materials science and many aspects of engineering. The subject is complicated by the fact that human perception of colour is influenced by many factors: physical and mental illness, ageing, physical injury, surroundings, motion, alcohol drugs, past experience (e.g., working in the printing industry) and even language and culture. Much of Colour Science was developed by the need to reproduce colours accurately, for printing, photography and video displays. This does not have much relevance to our immediate need, as we shall show later. In this essay, we shall concentrate on those aspects of the subject that relate directly to Machine Vision, taking a pragmatic approach and ignoring those areas of Colour Science that do not contribute directly to the design of practical inspection, measurement or guidance systems.

As so often happens in Machine Vision, we are interested in verifying that a manufacturing process is running properly. Hence, we know what to expect from a "good" sample. Any significant deviation from the norm is an error that must be detected by the inspection system. Moreover, we are limited by the sensors that we can use. Colour video cameras provide us with just three channels, usually referred to as *R*, *G* and *B*. While Colour Science suggests that other representations (such as hue, saturation and intensity, *HSI*) may be better for certain tasks, we have to base all of our processing on the RGB signals from a camera. If a colour recognition task is impossible using the RGB signals, it is also impossible with HSI as well. (*H*, *S* and *I* can be calculated from *R*, *G* and *B* and vice versa.) On some tasks, it may be more convenient in practice to use HSI rather than RGB but we gain no theoretical advantage in doing so. It must be understood that no representation (called a *Colour Model*), however attractive theoretically, provides any direct advantage to us, since we can only buy cameras that supply RGB signals. The RGB system for video cameras was devised on the assumption that human beings have trichromatic vision. There is good evidence for this; cinema and video image reproduction are based on this idea. If it were not so, we would have to tolerate poor colour reproduction when watching television and films.

4.1.1 Why Is a Banana Yellow?

First, let us consider the physics involved in this question. The visible part of the electromagnetic (EM) spectrum comprises wavelengths between about 0.38 and 0.74 μm (➋ *Fig. 4.1*). A light signal containing components with a non-uniform mixture of wavelengths will appear to be coloured to a human observer. If the energy spectrum is nearly constant for all

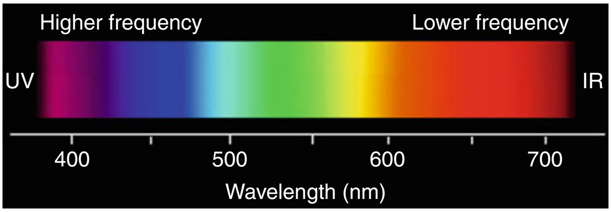


Fig. 4.1
The electromagnetic spectrum and colours within the visible (VIS waveband)

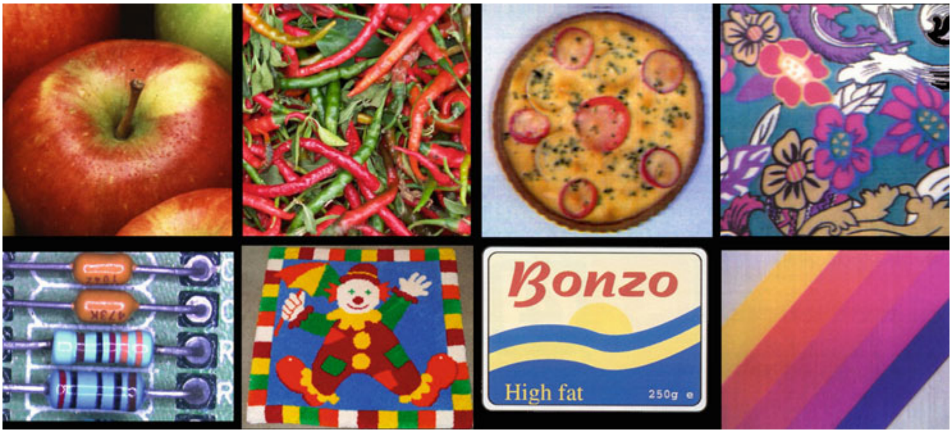
wavelengths in this range, the observer will perceive white light. The (non-uniform) spectrum of light reaching the eye of a human observer from a banana leads the brain to call it "yellow." A similar but slightly different spectrum of light might emanate from a canary, or lemon. The seemingly trivial question at the head of this section has a deeper meaning than we might at first think. When faced with this question, many people begin by thinking about Newton's demonstration of the multiple-wave composition of white light and the fact that bananas reflect light within a certain range of wavelengths, while absorbing others. Curiously, most people, even non-scientists, attempt to answer this question with reference to the *physics* of light absorption and reflection. However, light rays are not actually coloured. So-called "yellow light," reflected from a banana, simply appears to us to be coloured yellow, because the human brain is stimulated in a certain way that is different from the way that it is when the eye received light from an orange or tomato. This stimulus results in the brain responding with the colour label "yellow."

In fact, a banana is yellow because a lot of people collectively decided to call it "yellow." As young children, we are taught by our parents to associate the sensation of looking at lemons, butter, bananas and canaries (and many other yellow things) with the word *"yellow."* At that age, we do not know anything about the physics of differential light absorption. Of course, we do not need to do so, even as adults. A ripe banana *"is yellow"* because a lot of other people persistently call it *"yellow"* and we have simply learned to do the same. There is no other reason than that! In particular, the physics of light has got nothing whatsoever to do with the *naming* of a colour. Physics can only explain how the particular mixture of wavelengths that we have learned to associate with *"yellow"* is generated.

4.1.2 Defining Our Objectives

We can best define our area of interest by example, with specific reference to some of the industrial inspection and control tasks that we might want to perform (➜ *Fig. 4.2*). Broadly speaking, our objective is to provide *coarse* colour recognition facilities. We make no apologies for using the word "coarse," since it is a fair description of what we need to do! Here are a few more examples of tasks that fall within this category:

- Grading and sorting fruit and vegetables. This is important for quality control and for checking bulk delivery consignments against the details of the product actually ordered.
- Inspecting food products. A certain cake that is popular in Britain is called a Bakewell tart and consists of a small pastry cup, filled with frangipan, is covered in white icing (frosting) and has a cherry placed at its centre (➜ Chap. 23)



■ **Fig. 4.2**

Typical products for which coarse colour recognition is appropriate. (a) Grading/inspecting fruit. (b) Sorting vegetables. (c) Quality control of manufactured food products. (d) Inspecting dress fabric. (e) Reading colour codes of electronic components. (f) Inspecting carpet (hand-knotted rug). (g) Inspecting packaging (simulated image of butter-substitute container). (i) Identifying packaging (Feature. The colour bands identify the product type)

- Checking that fabric has been dyed, woven, knitted, or printed correctly.
- Checking that each carton of sweets (candies) contains a certain proportion of orange ones. What could be worse than not having any orange M&Ms in a packet? (➲ Chap. 16)
- Checking labels on boxes, bottles, jars, cans and other containers. In the pharmaceutical industry, there are serious concerns about the safety of mis-printed boxes. In other industries, carton inspection is important for two reasons. The customer sees the box first and the most popular "product" of all is containers!
- Reading component colour codes for the electronics industry.

More examples of coarse colour recognition will be encountered as we proceed through this chapter. Another example of coarse colour recognition is shown in ➲ *Fig. 4.3*.

4.2 Physics of Colour

Visible light is electro-magnetic (EM) radiation within a narrow wave-band and is usually characterised by its spectrum (➲ *Figs. 4.1* and ➲ *4.8*). This may be expressed as the *intensity* (i.e., number of photons/s), measured as a function of wavelength *(λ)* or frequency (*f*). These are related by the equation:

$$v = f.\lambda$$

where v is the velocity of light in whatever medium the light is traversing. In air, the velocity of light is close to that within a vacuum: 2.998×10^8 ms^{-1}. EM radiation with a wavelength inside the band 0.38–0.74 μm is visible to the human eye and is therefore known as visible light (VIS). Within this band, different wavelengths produce different sensations of colour to an observer. In practice, there is a multitude of wavelengths in the light reflected from a surface, or emitted

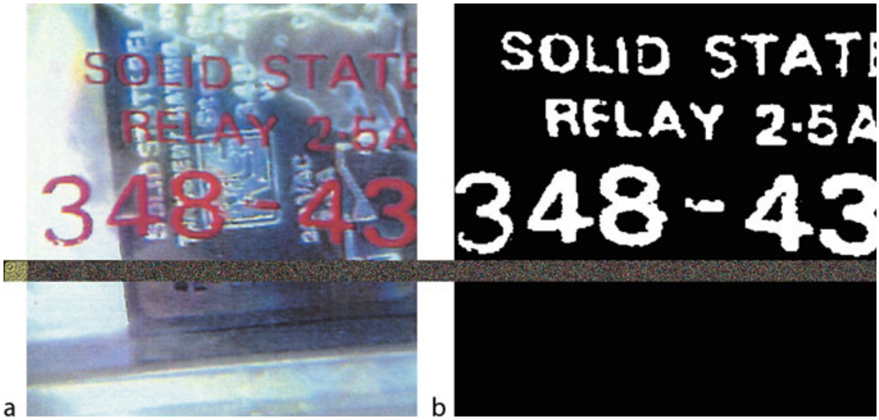


Fig. 4.3
Coarse colour recognition. (a) Original image: black electronic component with silver printing, enclosed in a plastic bag. The latter has the red text printed on it. (b) Output of the programmable colour filter described later in this chapter

by standard ambient lighting. (Lasers and LEDs provide notable exceptions to this.) Notwithstanding the remarks made later, the spectrum is the principal factor in determining the perceived colour of light.

4.2.1 Interaction of Light and Matter

When light strikes a surface, it can be reflected, absorbed, scattered, or re-emitted at longer wavelengths (fluorescence). Several different optical effects can take place simultaneously and combine to form an emergent beam with a spectrum that quite different from that created by any one of them. Some of these effects are orientation specific and hence the combined beam might well produce quite different colour sensations in the observer, depending on the illumination and viewing angles (➋ *Fig. 4.4*). A surface that diffusely reflects all wavelengths in equal measure is seen as being grey or white. A surface that totally absorbs all wavelengths is perceived as black.

In English, it is common practice to refer to the rainbow as having seven colours: violet, mauve, blue, green, yellow, orange and red but, in reality, there is an infinity of “colours.” Subdividing the spectrum like this is an artefact of language, created out of the need to give names to colours and has no physical significance or biological basis. In some languages, fewer colour terms are applied to the rainbow but there are seldom more than seven [10]. The rainbow contains all of those colours that can be formed by visible light of a single wavelength. These are known as *spectral colours* and can be generated optically, using a prism or grating. By selecting light from a very narrow region of a “rainbow,” it is possible to build a *monochromator*. The arrangement shown in ➋ *Fig. 4.5* can be used to select any desired spectral colour from a panchromatic source, or for spectral analysis of the light reflected from a surface.

Interference occurs when a surface has a microscopically repeating structure, where the spatial period is comparable to the wavelength of VIS light. It occurs in nature, on butterfly and beetle wings, and is seen on CDs, DVDs and hologram wrapping paper (➋ *Fig. 4.6*). The brilliant colours it generates are well known and are dependent on the angles of illumination

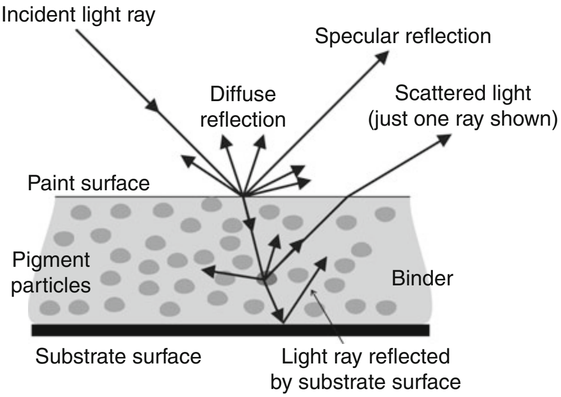


Fig. 4.4

When light falls on a painted/lacquered surface, its spectrum can be altered in several ways. Paint consists of numerous tiny pigment particles suspended in a clear binder. Reflection at the surface of the paint, transmission through the binder, reflection/scattering from the pigment particles and reflection from the substrate all contribute to changes in the colour of the emergent light. If the pigment particles are elongated and aligned in one direction, the apparent colour of the surface is highly directional

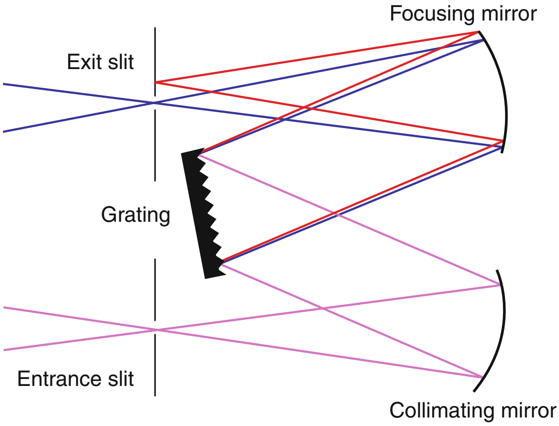


Fig. 4.5

Czerny-Turner monochromator. White light is focussed onto the entrance slit, is collimated by a curved mirror and is then projected onto a grating. The dispersed beam is re-focussed by a second mirror, onto the exit slit. Each wavelength is projected to a different position at the exit slit; the wavelength that is transmitted through it depends on the angular position of the grating

and viewing. Holograms also produce colour by optical interference and the techniques that we shall discuss later in this chapter could be used to verify the authenticity of holograms, such as those found on credit cards (➋ *Fig. 4.6c*). Iridescence is an optical effect in which the colour of a surface varies with viewing angle. It is caused by interference, when multiple reflections occur in thin multi-layered, semi-transparent surfaces. Iridescence is seen on pearls, soap bubbles and

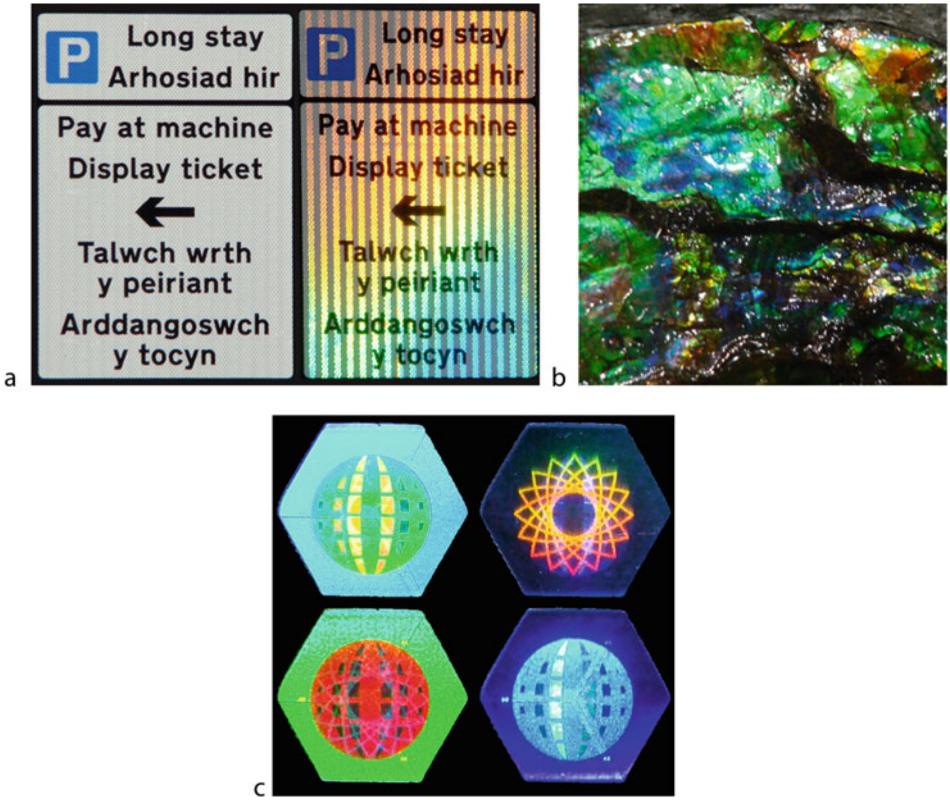


Fig. 4.6

Structural colour. (a) Street sign, variable lighting. (b) Ammolite. (Fossiled ammonite shell) (c) Hologram on a credit card, variable lighting

optical coatings. Dichroic mirrors, such as those used on the integral reflectors on projector lamps show the same effect. Iridescence is sometimes exploited for decorative effects on the packaging of goods such as cosmetics and on jewellery (➋ *Fig. 4.6b*).

Most light sources encountered in everyday life, including the sun, incandescant filament and fluorescent lamps, do not emit pure spectral radiation; we most commonly experience light containing a mixture of wavelengths (➋ *Fig. 4.8*). Sometimes, light sources with quite different spectra may be perceived by the eye in the same way as each other and the same as a pure spectral colour. For instance, a computer screen displaying an "orange" feature does not emit pure spectral light. Instead, the screen emits a mixture of "red" and "green" light, with an intensity ratio of about 2:1. There is little, or no, blue component. To produce the same sensation, pure spectral light of about 0.6 μm would be needed. For sources that emit across a broad spectrum, it is often convenient to refer its *dominant wavelength*. This is the wavelength of pure spectral light that produces the same perceptual sensation as the mixed (i.e., non-spectral) source. The dominant wavelength effectively determines the *hue* of the light.

It is important to note that many colours cannot be represented in this way, using a single pure spectral colour. For example, purple, which is a mixture of red and violet light cannot. The *achromatic colours* (i.e., black, grey and white), pink, brown and magenta do not produce the same visual sensation as any one spectral colour.

4.2.2 Image Acquisition Sub-system

The spectrum of light reaching the retina (of the eye or a camera) is given by the following product:

$$L(\lambda)R(\lambda)O_1(\lambda)O_2(\lambda)...O_N(\lambda)$$

where

L(λ) is the spectrum of the light emitted by the primary source

R(λ) is the reflection characteristic of the surface being viewed

$O_1(\lambda)$, $O_2(\lambda)$,...,$O_N(\lambda)$ are the spectral transmission characteristics of the various components in the optical path. This includes the reflector on the luminaire, the air path, lenses, filters, optical window on the camera.

The most significant elements determining the spectral characteristic of the optical system are normally the light source and any filters that are deliberately introduced to modify the spectrum.

4.2.2.1 Light Sources

Lighting can greatly influence the visual appearance of coloured surfaces and the signals generated by a video camera. ➲ *Figure 4.7* illustrates this by showing the same scene illuminated in five different ways [12]. Notice that, in two cases (➲ *Fig. 4.7c and e*), there is no obvious relationship to the colours that a person would observe when viewing the same scene in natural light. In the other three cases, the "real" colours are identifiable, although there are still clear differences due to the lighting. ➲ *Figure 4.7* was prepared using a camera and the colour changes due to alterations in the lighting are obvious. However, a person viewing the same scene directly would, under certain conditions, observe far smaller changes. This is due to the phenomenon called *colour constancy* and in which the human visual system compensates, in part, for variations in ambient lighting. Neither a photographic nor video camera makes adjustments like this. Hence, in some cases, a human observer's perception of colour changes under varying different lighting conditions is much smaller than a Machine Vision system would experience. The choice of appropriate lighting is therefore of great importance for machine recognition of colour; there are two critical conditions:

(a) The lamp must emit strongly in all relevant parts of the spectrum. (What is relevant depends, of course, on how the surface reflectance function varies with wavelength.)
(b) The lamp emission spectrum must be stable, both in the short and long term.

Lamps differ considerably in their spectral characteristics (➲ *Fig. 4.8*). Daylight has a smooth wide spectrum spanning the whole visible waveband. An incandescent filament lamp has a lower shortwave output than daylight and hence oranges and reds appear to be more prominent than in daylight. Many lamps are based on the emission of light from excited ions of sodium or mercury. These lamps tend to have prominent well-defined peaks in their spectra, due to electron transition from one energy level to another. Strong, spiked spectral peaks can cause problems in certain situations. Since the peaks occur at well-defined

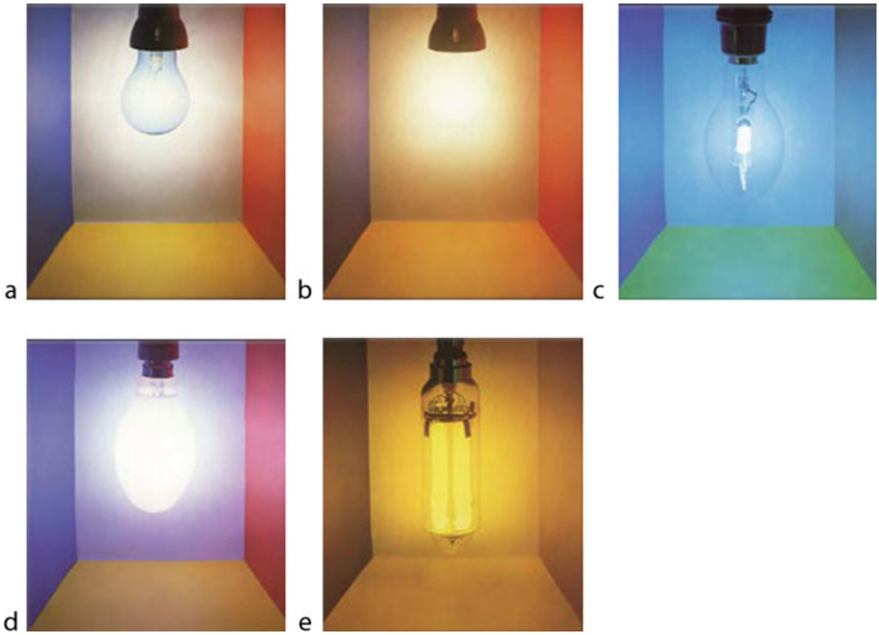


Fig. 4.7
Effects of lighting on the appearance of coloured surfaces. (a) Artificial day-light lamp. (Has a similar spectrum to a cloudy northern sky at mid-day, Northern hemisphere). (b) Household incandescent filament lamp. (c) Mercury lamp. (d) High-pressure sodium lamp. (e) Low pressure sodium lamp (nearly monochromatic). (Reproduced from [12], used with permission)

wavelengths, they can be exploited to good effect in some applications. Some lamps have been designed to match the spectral characteristics of daylight and are widely used in photography and video recording. When lamps have been in use for a long time, the light output usually falls but the spectrum does not normally change much, as this is defined by atomic behaviour. However, there is a special situation that should be borne in mind when using filament lamps, with feedback applied to stabilise the light output. There is a temptation to use the feedback signal to adjust the current through the lamp. However, this alters the filament temperature and hence the spectrum; as the lamp current increases, the filament gets hotter and emits more strongly at shorter wavelengths. To stabilise the light output without altering the colour, it is better to use a variable-size optical aperture, rather than adjusting the lamp current.

4.2.2.2 Adjusting the Spectrum

It is possible to adjust the spectrum of the light projected onto a widget at will, using a liquid crystal light valve. (Lighting-viewing method number 44, ➲ Chap. 40) This is an expensive option but virtually any arbitrary spectrum can be created under software control. A cheaper alternative exists, using a series of LEDs to generate light at different wavelengths. In ➲ Sect. 4.3.4, we show how the contrast between two surfaces of seemingly similar colour

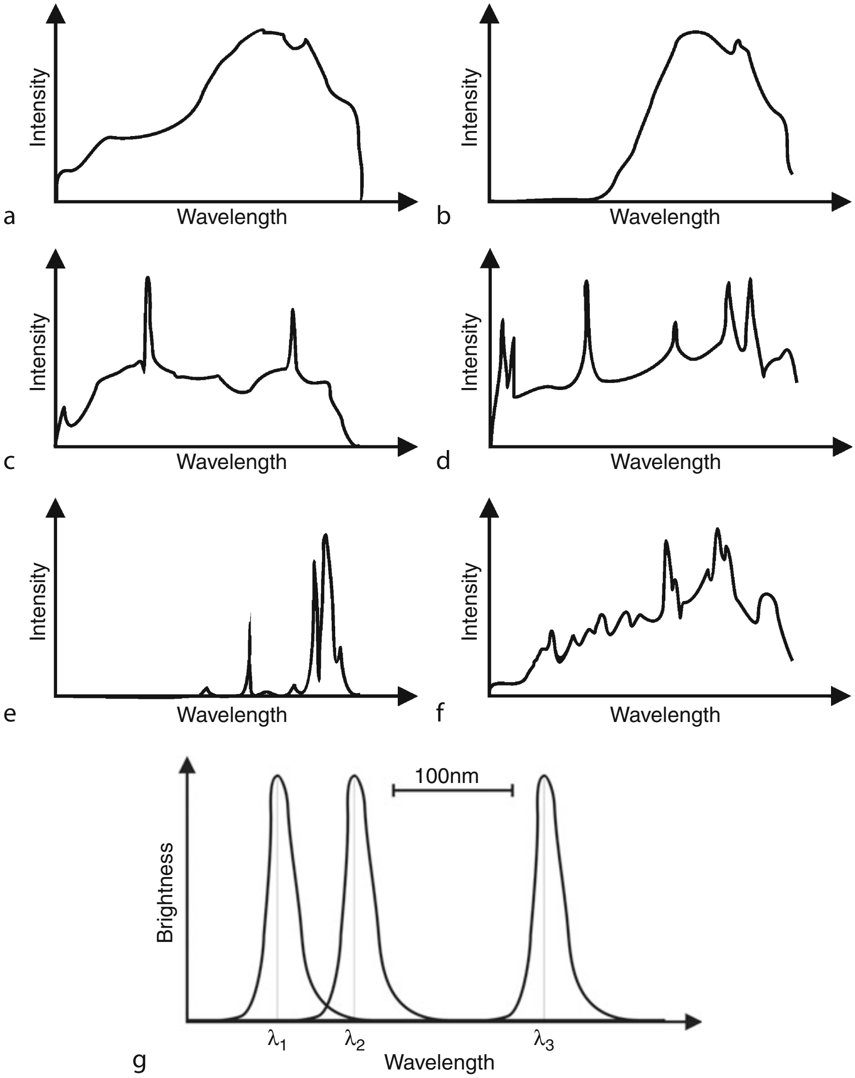


Fig. 4.8

Spectra of common light sources. (a) Daylight. (b) Incandescent filament lamp (c) Fluorescent. (d) Low pressure mercury. This is a strong UV emitter. (UV spectrum not plotted). (e) Low pressure sodium. (f) Xenon flash. (g) LED spectra. A single LED emits light over a narrow band of wavelengths. LEDs can be ordered with spectral peaks at any specified position within the VIS waveband. Hence, illuminants may be built that combine the outputs from several LEDs of different colours. In this illustration, the light from three LEDs, whose peak wavelengths are λ_1, λ_2 and λ_3, is mixed. A tunable light source with a spectrum that consists of a series of spikes can be useful for improving contrast in scenes containing metamers. Individual LEDs can be strobed in sequence, to generate multiple colour-separated images, or always switched ON to obtain a single composite image. Their drive currents can be adjusted individually to alter the balance of their contributions to the spectrum of the mixed beam

(metamers, see ➲ *Fig. 4.18*) can be maximised by using LEDs to generate light with several peaks at appropriate points in the VIS waveband. Using only "off the shelf" LEDs, it is possible to match a given wavelength quite accurately (error less than ±10 nm). Hence, a light source with a multi-peak spectrum can be built using LEDs (➲ *Fig. 4.8*). Moreover, any predefined set of LEDs within an assembly can be strobed. This enables spectral sampling and results in a series of "colour separation" images. Clearly, this is only feasible for illuminating small-medium size objects. Image acquisition is slow, taking several video frame periods.

4.2.2.3 Filters

Another way to modify the spectral characteristic of the image acquisition system is to use filters. These are available to perform of the following functions (➲ *Fig. 4.9*):

- Block short wavelengths (long-pass filter)
- Block long wavelengths (short-pass filter)
- Block all but a very narrow spectral band (band-pass filters)
- Block only a very narrow band (band-stop filter)
- Block ultra-violet and pass VIS light, or vice versa
- Block infra-red and pass VIS light, or vice versa

Interference filters have very sharp cut-off characteristics and can be "tuned" slightly, by adjusting the angle relative the light path (➲ *Fig. 4.10*). Filters can be concatenated. For example, a long-pass filter and a short-pass filter can be combined to select a wide band within the VIS spectrum. Two or more band-stop filters can be used to suppress multiple peaks, for example, in the spectrum of a mercury lamp.

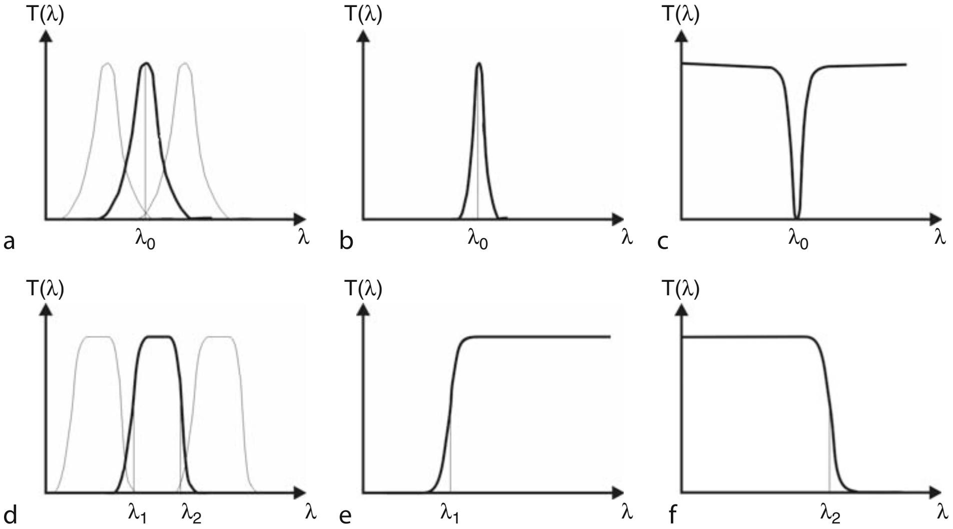


Fig. 4.9
Optical filters. Filters are available for near-UV, VIS and near-IR. (a) Narrow band-pass filter. (b) Very narrow band-pass filter. (c) Very narrow band-stop filter (d) Wide band-pass filter. (e) Long-pass filter. (f) Short-pass filter

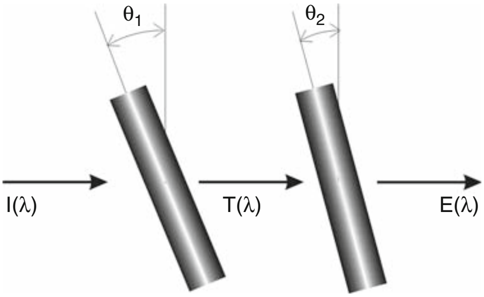


Fig. 4.10
Cascading optical filters. The incident beam (I(λ)) is short-pass filtered to produce beam T(λ). This is, in turn, long-pass filtered to produce the emergent beam E(λ). The exact cut-off wavelengths of the filters can be adjusted by altering the tilt angles θ_1 and θ_2

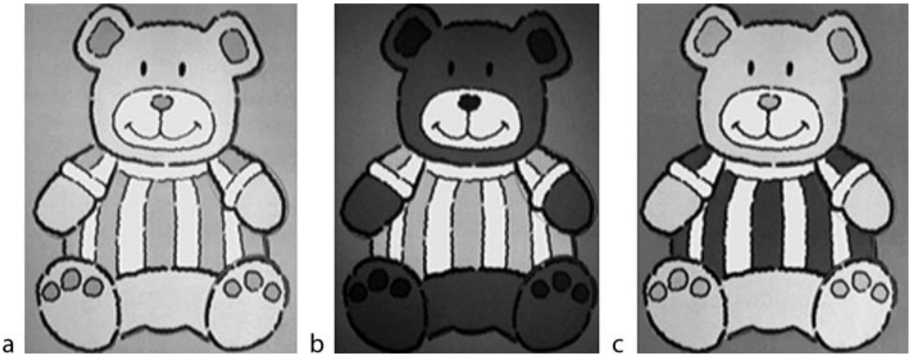


Fig. 4.11
Interference filters have a sharp cut-off and can greatly increase image contrast. (a) Child's toy printed on card. No filter, monochrome CCD sensor. (b) Short-pass filter ($\lambda_{\text{cut-off}} = 0.55\ \mu$m). (c) Long-pass filter ($\lambda_{\text{cut-off}} = 0.55\ \mu$m)

Where the function is to be switched automatically, perhaps to enable a factory-floor inspection system to accommodate a different product line, several filters can be mounted on an indexed wheel. Commercial filter wheels are operated by software, via a low-speed interface (RS232) (Lighting-viewing method 75).

Using interference filters, large increases in image contrast can often be achieved (➲ *Fig. 4.11*).

Filters can be mounted immediately in front of the lamp. However, it is usually better to place them between the object being viewed and the camera's main lens: smaller filters are needed and are not heated by the lamps. Once a filter has been purchased and mounted for a given application, there are no "running" costs. Optical filters are much faster and more reliable than software. The disadvantage is, of course, that the filter function is fixed.

Colour filters can be used, with separate sensors in cascade, to create a multi-spectral image series (➲ *Fig. 4.12*).

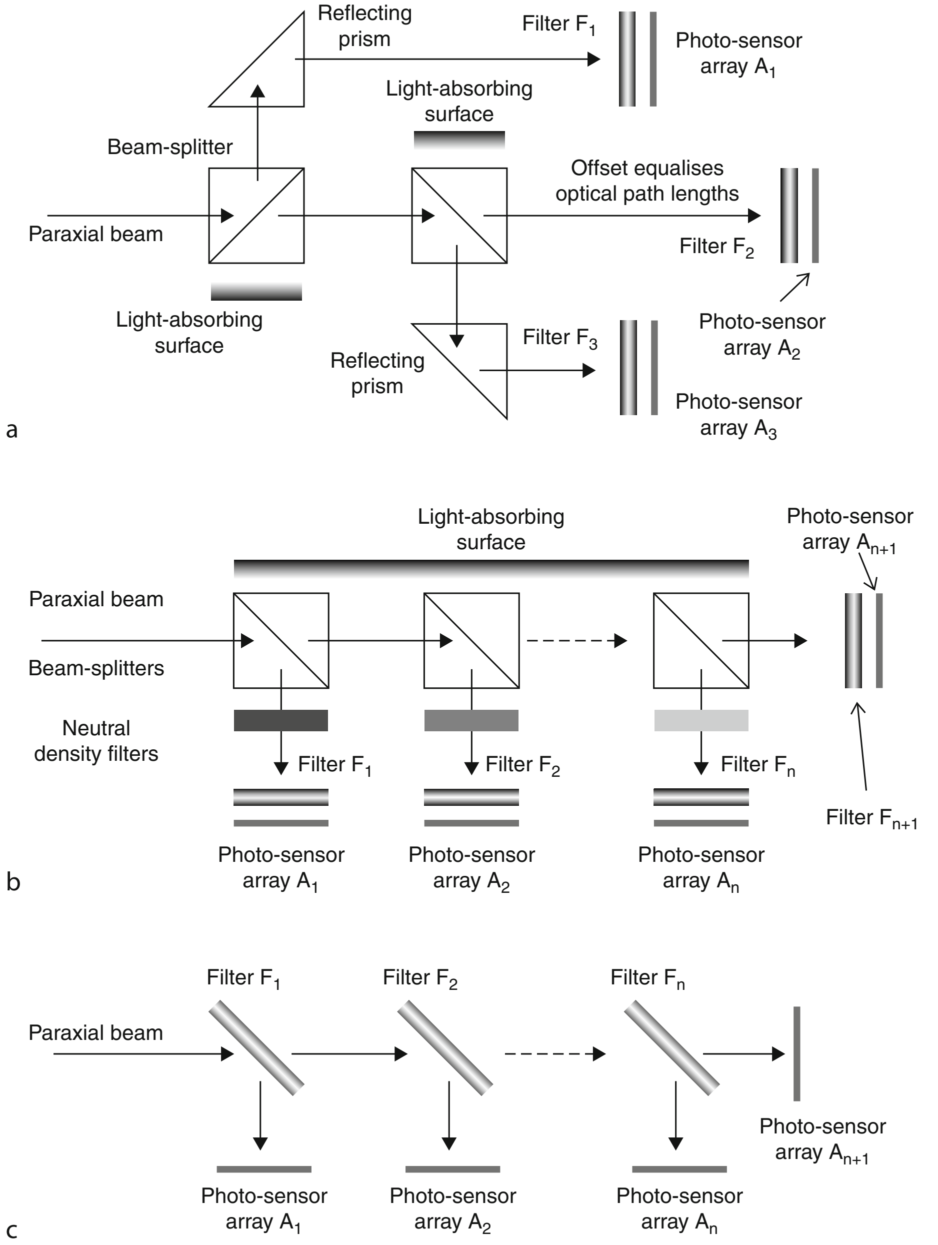


Fig. 4.12

Optical arrangements for multi-spectral imaging. The photo-sensor arrays are capable of detecting light over wide range of wave-lengths. (a) Branching structure of a 3-sensor colour camera. (b) Linear array of beam-splitters. (c) Linear array of dichroic filters. These act as mirrors for a certain range of wavelengths and transmit others. No attempt is made in (b) and (c) to equalise the path lengths, which would ensure that the images received by the photo-sensor arrays are all of equal size and in focus

4.3 Human Colour Vision

It is now widely accepted that the retina of the human eye contains three different types of colour receptor cells, or cones. One type, relatively distinct from the other two and misleadingly called *blue cones* (or more correctly *S* cones), is most responsive to light that we perceive as violet (wavelengths around 0.42 µm). The other two types (*green* and *red cones*, properly called *M* and *L* cones) are more closely related to each other and are most sensitive to green and yellowish-green light (wavelengths of about 0.534 µm and 0.564 µm respectively) (➋ *Fig. 4.13*).

Rod cells in the retina cannot sense differences in wavelength and are reputedly not involved in colour vision, except under very special conditions. Notwithstanding the fact that the "normal" eye actually contains *four* wavelength-selective sensors (A, M and L cones, as well as rods), most colour scientists believe that cones are the principal sensors involved in human colour vision. Based on this idea, the so-called *Tristimulus Theory* has been devised. This leads us to understand that every optical spectrum, however complex, can be represented within the eye by three values (*tristimulus* values) indicating the strength of the response of the S, M and L cones. The sensitivity curves of the cones overlap (➋ *Fig. 4.13*). Notice that some combinations of responses are impossible, irrespective of the shape of the spectrum of the incoming light. For example, it is impossible to stimulate only the M cones, without also stimulating the S and L cones. to some extent. (The reason is, of course, that rods and cones are very small and are packed together very tightly in the retina. Furthermore, there is no easy way to identify whether a given cone cell is S, M or L type.)

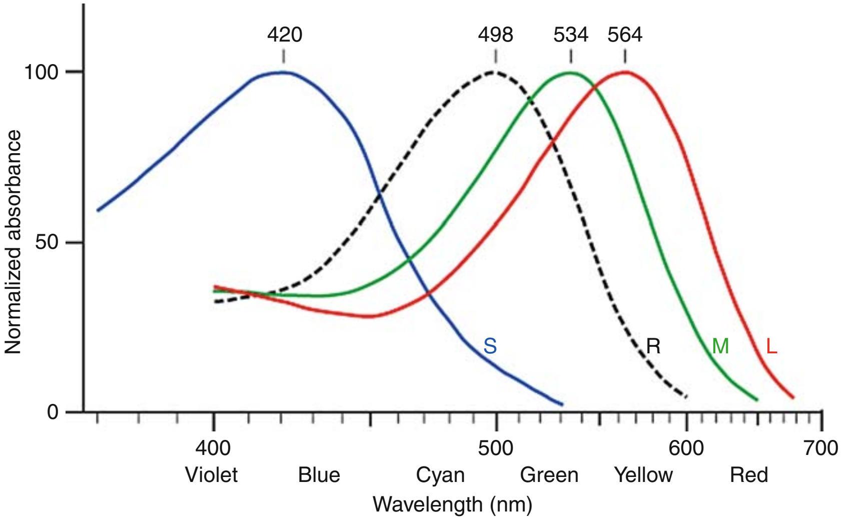


Fig. 4.13
Normalised spectral response of the cones (curves labelled S, M and L) and rods (R) in the human eye. The peak responses are at 0.42 µm (S cones), 0.534 µm (M cones), 0.564 µm (L cones) and 0.498 µm (rods)

Two important facts have been discovered experimentally:

(a) Any colour can be matched by visual observation to a combination of not more than three spectral colours. However, not all colours can be matched to any one set of spectral colours. It should also be noted that it may be necessary to use "virtual" light sources that have no physical existence and to allow negative amounts of light to do this.

(b) Additive relations in colour mixtures have been found to hold under a wide range of conditions.

These are important because they allow us to begin building a mathematical model of colour vision.

4.3.1 Tristimulus Model

Like most children, the author learned at school that *any* colour can be created by mixing red, yellow and blue paint. This is not true! Later, he was taught that mixing red, green and blue light can generate *all* colours, including white (➋ *Fig. 4.14*). This is not true either! A wide range but not all colours can be reproduced in this way, as is evident from colour television; the receiver display screen produces tiny red, green and blue spots of light. Since the viewing conditions are such that these spots cannot be resolved by the eye, they merge to create a wide variety of different colour sensations. In fact, the picture quality is so good that most people are not normally aware of the fact that there are certain colours that television cannot show.

4.3.2 Subtleties of Human Colour Perception

If one or more types of the colour-sensing cones are missing or less sensitive than normal, the person concerned is said to be *colour deficient*, or in popular terms *colour blind*. The latter term can be misleading, since only a small proportion of colour-deficient people are totally unable to detect colour at all. It is of interest to note that some animals (e.g., marsupials, birds, reptiles,

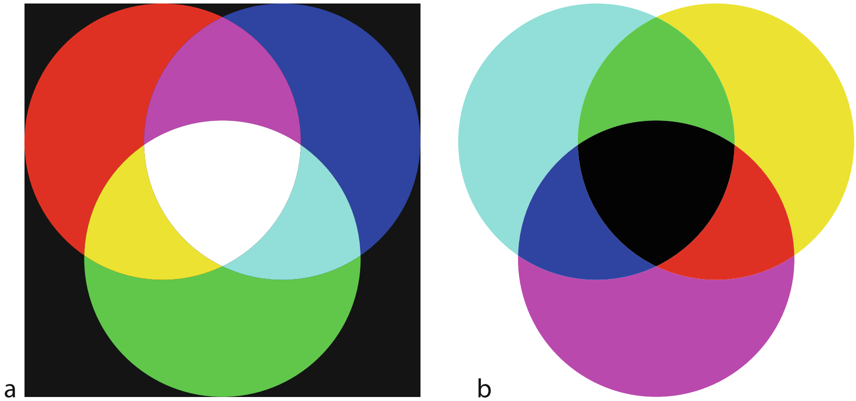


Fig. 4.14

Colour mixing. (a) Additive, produced by mixing coloured lights and hence is relevant to film and video. (b) Subtractive, relevant to painting, dyeing and printing and colour film photography

and fish) are thought to have more than three different types of colour sensor, or fewer (most mammals). Human beings and other old-world primates are actually rather unusual in possessing three kinds of receptors. (*trichromats*) Some women (*tetrachromats*) are thought to have four, rather than three, distinct types of cone cell. The most common form of colour deficiency leads to an inability to distinguish red and green properly and occurs in about 8% of men. A variety of factors can change colour perception, including eye disease, neurological disease, head injury, stroke, dementia, narcotics and severe alcohol abuse.

Numerous optical illusions have been devised that disturb our normal perception of colours. Under certain conditions, alternating red and green stripes on a computer display will "disintegrate" into a shimmering jumble. The naming of a colour can be altered by its context. We are all familiar with the formation of after-images and the changes in colour perception that they introduce. After staring at a coloured patch for a few seconds, an after-image containing the complement of that colour is perceived by the eye for some time after it moves (➲ *Fig. 4.15a*). When a small area of one colour is enclosed within a much larger region of another contrasting colour, a human observer may experience some change (➲ *Fig. 4.15b*). Optical illusions such as these can provide Colour Scientists with insight into the working of the human visual system.

In the 1950s, the BBC Television Service experimentally broadcast grey-scale images that vibrated. Since motion can cause apparent changes of colour, this was done in the hope of exploiting this curious feature of human vision for public television broadcasting. However, the experiment was soon abandoned as the results were unreliable and many people find vibrating images disconcerting. Also see [29].

The apparent brightness of a light source depends upon the number of photons reaching the eye from it each second. The source brightness can alter its perceived colour. For example,

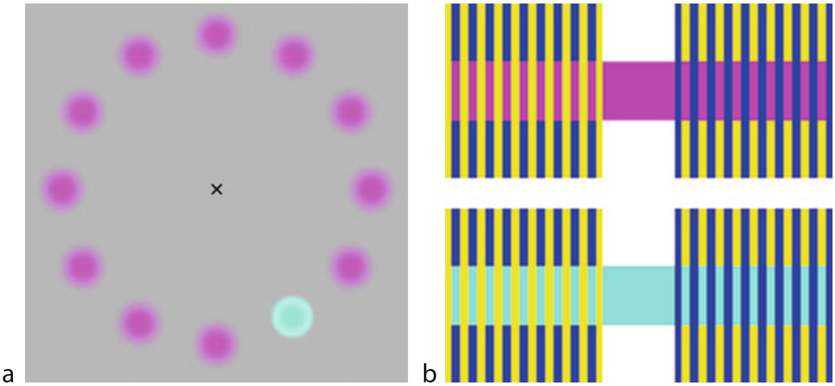


Fig. 4.15

Subtleties of human vision revealed in optical illusions. (a) Troxler fading (due to adaptation of the eye) and the creation of complementary colour as a negative after-effect. In a web-based animated sequence, 11 fuzzy violet spots are shown, with the "empty space" rotating around the centre of the image. When an observer stares at the central cross, a fuzzy green spot appears to rotate instead. After a short while, the violet spots disappear and the green spot continues to rotate but now against a uniform grey background. The animated version can be viewed at [29]. This illusion is capable of creating colours outside the gamut of the display device. (b) Colour perception depends on context. The mauve/turquoise bands seem to change colour when they are viewed against different backgrounds

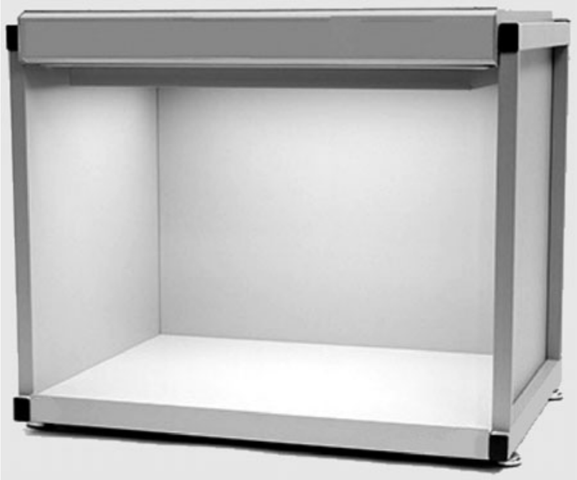


Fig. 4.16

Viewing booth for colour matching experiments. All surfaces are matt grey. The coloured background is shown here merely to emphasise this point and would normally be grey too. In addition, subdued room lighting would be used. In order to match colours under standard daylight conditions, the booth should use 5,000 K lighting

a low-intensity orange-yellow appears to be brown, while at low-intensity yellow-green appears to be olive-green.

In human beings, the polarisation of the light reaching the eye does not significantly affect visual sensation. However, in some fish and mollusks, it does. Of course polarising sun-glasses can alter perceived colours.

An important issue that affects colour perception is ambient lighting. Within quite broad limits, people are able to compensate for changes in the spectral distribution of the light falling on a multi-colour surface. This effect is known as *colour constancy*. However, it can sometimes be misleading and can lead to spectacular mistakes, when witnesses in court differ in their descriptions of the colour of a car seen under sodium or mercury street lighting.

An experimental arrangement, based on a viewing booth in which the total viewing experience is both simplified and held constant, allows a person to see large blocks of colour in a standard way (➋ *Fig. 4.16*). This is necessary to obtain reliable "naming" of colours for teaching a vision system. Under carefully controlled lighting-viewing conditions, many of the complications outlined in this section can be avoided.

4.3.3 Language and Culture

Different cultures have different terms for colours. This is not simply a matter of translation; there are no equivalent words. For example, in Welsh, there is no word corresponding exactly to grey, green or blue (➋ *Fig. 4.17*). French *jaune* has a slightly different meaning from English *yellow*, although the author was taught otherwise when he was at school! The Han character 青 (*qīng* in Mandarin and *ao* in Japanese) covers both blue and green in English. Italians refer to

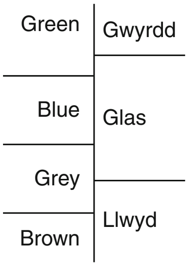


Fig. 4.17
Colour terms in English and Welsh do not correspond exactly, despite their having been in close proximity to one another for 1,400 years

the "red" of an egg, when refering to its yolk. Russian and English differ in the terms used for light blue and dark blue. In some languages, there are fewer colour terms than we are able to employ in English [10]. As a result, it is often impossible to translate a colour term exactly into another language. For example, this chapter cannot be translated exactly into Japanese, Chinese, Russian, Italian, French or Welsh! Colour terms change over time. Colour terms that have long been familiar to English speakers have been adopted in Japan only since 1945.

People working in the printing and clothing industries develop the ability to discriminate fine colour differences that are often missed by the general population. Gardeners, farmers, vetinary surgeons and doctors are all acutely aware of the importance of palour as an indicator of health in living things. Workers in steel and other hot-metal industries learn to sense the temperature of the incandescent, or molten, material from its colour. Apple Macintosh Computers are used frequently in printing, fashion and other colour-based industries. For this reason, they are provided with a non-coloured operating environment, called *Graphite*. This avoids users' colour vision being upset by gazing at vividly coloured screen displays for long periods of time.

4.3.4 Metamerism

In order to understand the processes involved in colour vision better, Colour Scientists often use mathematical notation. We shall adopt this approach to explain another interesting feature of the human vision: *metamerism*. If two surfaces appear to be similar under one set of lighting conditions but different under another, their colours are said to be *metamers*. As we shall see, metamerism is inevitable for any system, including Machine Vision, that relies on an RGB sensor. Metamerism can sometimes be beneficial in Machine Vision.

The following analysis is based on the tristimulus theory and *Grassmann's Law* [15]. The latter is essentially a statement that, over a wide range, the subjective sensation, achieved when light beams are added, can be predicted by adding their individual effects. Consider two monochromatic light beams *P* and *Q*. Suppose that a human observer chooses $[R_P,G_P,B_P]$ as the mixture of the RGB primaries that match beam P and $[R_Q,G_Q,B_Q]$ as the strengths of the primaries that match beam Q. Then, if P and Q are combined by adding them together, the matching values for the combined beam will be $[R,G,B]$ where:

$$[R, G, B] = [R_P, G_P, B_P] + [R_Q, G_Q, B_Q]$$

Consider an additive mixture of pure spectral components with wavelengths λ_1, λ_2 and λ_3, of intensities K_1, K_2 and K_3 respectively. This will excite the cones in the retina by the following amounts

$$\text{S cones} \quad S = K_1S(\lambda_1) + K_2S(\lambda_2) + K_3S(\lambda_3)$$

$$\text{M cones} \quad M = K_1M(\lambda_1) + K_2M(\lambda_2) + K_3M(\lambda_3)$$

$$\text{L cones} \quad L = K_1L(\lambda_1) + K_2L(\lambda_2) + K_3L(\lambda_3)$$

In these equations $S(\lambda)$, $M(\lambda)$ and $L(\lambda)$ are the spectral response functions of the S, M and L cones respectively (see ❯ *Fig. 4.13*). Let us now consider what happens when the eye receives light with a continuous spectral function $F(\lambda)$. This will excite the cones by the following amounts:

$$E_S = \int F(\lambda)S(\lambda)d\lambda$$

$$E_M = \int F(\lambda)M(\lambda)d\lambda$$

$$E_L = \int F(\lambda)L(\lambda)d\lambda$$

(These three integrals are computed over the range of visible wavelengths: 0.38–0.74 µm). Clearly, if $F(\lambda)$ is such that

$$E_S = S$$

$$E_M = M$$

and

$$E_L = L,$$

the spectra $F(\lambda)$ and $[K_1\delta(\lambda - \lambda_1) + K_2\delta(\lambda - \lambda_2) + K_3\delta(\lambda - \lambda_3)]$ are indistinguishable. ($\delta(.)$ is the impulse function.) If two different spectral functions $F(\lambda)$ and $G(\lambda)$ both satisfy these same conditions, they too are indistinguishable from one another and are called *metamers* (❯ *Fig. 4.18*). In addition, note that if $F(\lambda)$ and $G(\lambda)$ are metamers, so too is

$$H(\lambda) = \alpha \cdot F(\lambda) + \beta \cdot G(\lambda)$$

where α and β are scaling parameters. (This is true only if the resulting function, $H(\lambda)$, is always positive.)

Before describing the details of some of the more important colour models, we shall digress for a short while to consider metamerism in more detail. This is important in understanding human vision but, as we have already seen is an inevitable consequence of using only three different colour sensing channels.

Colour scientists studying human vision use the term metamerism in four ways, to include

- *Illuminant metamerism* which occurs when two spectra match according to an observer when using one light source, but fail to do so with another. This is often referred to simply as *metamerism*. We shall use this terminology, except where the term *metamerism* is explicitly qualified.
- *Geometric metamerism* which occurs when two surfaces match when viewed from one angle, but then fail to do so when viewed from a different angle. Geometric metamerism occurs when the spectral reflectance characteristics vary with viewing angle.
- *Observer metamerism* occurs when two spectrally dissimilar surfaces may match for one observer but fail to do so when a second observer views them under the same light source.

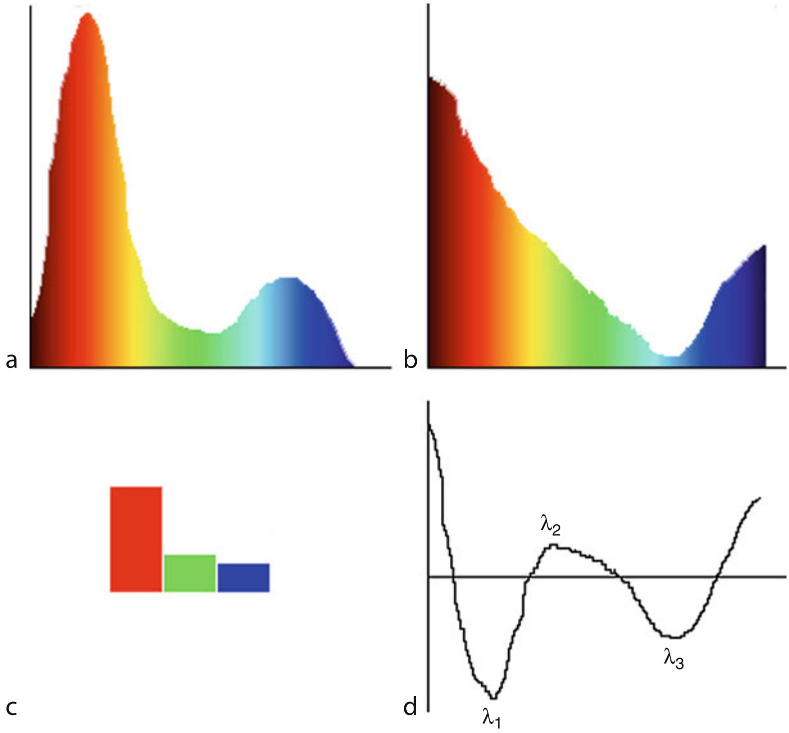


Fig. 4.18
Metamers. (a) and (b) Reflectance characteristics for two surfaces, S_1 and S_2 respectively. When S_1 and S_2 are viewed in perfect white light, the spectra of the reflected light have the same form. (c) RGB values generated by these spectra are identical. Hence, the visual sensation they create, in an observer or camera, is identical; S_1 and S_2 are metamers. (d) Differences between the spectra. By illuminating S_1 and S_2 with light having strong spectral peaks at λ_1, λ_2 or λ_3 the visual contrast is enhanced. These spectra were verified as being metamers, using a Java program [17]

This phenomenon occurs when two people have S, M and L cones in different proportions in the retina.

- *Field-size metamerism* occurs because the proportions of the three cone types varies across the retina. In the central part of the retina where there are no S cones. It is possible, therefore, for two surfaces to match when viewed from a distance but then fail to match when viewed from close up.

Metamerism is an inevitable consequence of using a small number (3) of wavelength-selective sensory channels. (Metamerism is akin to *aliasing*, caused when we fail to satisfy the Nyquist Sampling Criterion.) As a consequence, metamerism is encountered in human beings, animals and artificial vision systems. Despite the apparent crude coding for colour implicit in the tristimulus theory, healthy "normal" human beings can distinguish roughly 10^7 different colours. (This was established experimentally.) Furthermore, we are able to function very effectively as we interpret everyday colour scenes without our being continuously made aware of metamerism. However, the effect is of great significance and sometimes, of considerable benefit, when we are building colour recognition systems.

Since people normally view the world in nearly white light, they are often unaware of the presence of metamers in our everyday lives. However, when designing a Machine Vision system, it is a good idea to keep metamerism in mind. To understand how we can exploit metamerism, reconsider ➲ *Fig. 4.18*, which shows the reflectance characteristics of two metameric surfaces. Illuminating them with three monochromatic light sources of wavelength λ_1, λ_2 or λ_3, perhaps generated by carefully selected LEDs, can produce a significant improvement in contrast, compared to white light illumination.

4.3.5 Emulating Human Colour Recognition in a Machine

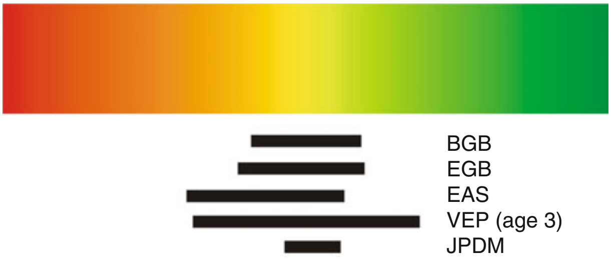
The exact meaning of "colour" is a matter that continues to vex philosophers. However, it is almost universally agreed that it is a psychophysical phenomenon that only exists within the human mind. The naming of colours is highly subjective, being dependent on experience, state of health, culture and language. My concept of colour is different from yours and yours is different from everybody else's. Even people who have lived together for many years do not always agree on the naming of colours. A French proverb implicitly acknowledges this:

- *Les Goûts et les coleurs ne se discutent pas.*
  (One should not argue about taste and colour.)

The subjective and individual nature of colour classes can be demonstrated quite easily. When people are asked to define the limits of what they perceive to be the limits of a certain colour (e.g., "yellow") on a continuous spectrum, they do not agree perfectly (➲ *Fig. 4.19*).

Given that they both have no obvious colour deficiency, no one person's definition of "yellow" is more/less valid than any other. This has significant repurcussions for building colour vision systems. Our subject is concerned with building machines that we hope can emulate a human being in associating a given set of surfaces with a particular colour name (e.g., "sulphur yellow"). We can formalise our objective in the following way.

A human teacher can separate colours into a large set of categories, *S*. On the other hand, a factory-based Machine Vision system will normally be expected to classify colours into just a few categories. (The exact number depends upon the application requirements.) Let the



■ Fig. 4.19
Subjective nature of colour. Several people were asked to define the limits of what they perceive to be *"yellow."* For this experiment, the colour bar was displayed on an LCD computer display (Printing the colour bar has altered its colours slightly from those seen on the monitor)

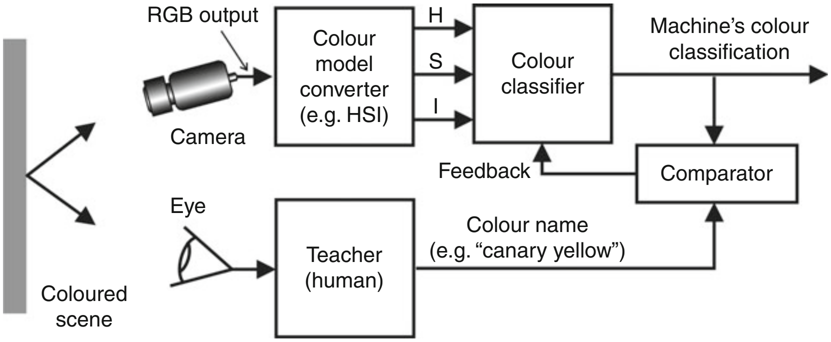

RGB output
Camera
Colour model converter (e.g. HSI)
H
S
I
Colour classifier
Machine's colour classification
Feedback
Comparator
Eye
Teacher (human)
Colour name (e.g. "canary yellow")
Coloured scene

Fig. 4.20
Archetypal colour recognition system; learning colours from a human teacher (This idea is developed later in this chapter)

teacher output in response to receiving light with a certain spectrum, $\boldsymbol{X}$, be $T(\boldsymbol{X})$. Notice that $T(\boldsymbol{X}) \in S$, for all spectra $\boldsymbol{X}$ within $\boldsymbol{Y}$, the set of spectra that our vision system may reasonably be expected to experience through its working life. We would like to build a Machine Vision system that, on observing light with a spectrum $\boldsymbol{X}\in\boldsymbol{Y}$, produces an output $M(\boldsymbol{X})$, such the following equality is true.

$$M(X) = T(X)$$

We have not specified precisely how the teacher reaches his/her decision. Indeed, the *teacher* may not be a single person but may, in fact, be a committee of experts, who generate $T(\boldsymbol{X})$, perhaps on a majority-voting basis. In theory, one possible way to build a machine to recognise colours would be to show the teacher an example of light with every possible spectrum and then build a machine that simply remembers the colour labels that the teacher ascribes to each one (➋ *Fig. 4.20*). In this way, we circumvent many of the philosophical questions surrounding the concept of colour. We can do this by simply copying an expert (i.e., the teacher) whose opinion about the naming of colours is well respected and authoritative. This pragmatic engineering approach to colour vision in machines neatly avoids many of the more subtle aspects of human vision; as stated earlier, the traditional approach to Colour Science has little relevance to the design colour recognition machines for Automated Visual Inspection. Nevertheless, we need to explain some of the fundamentals of this subject.

4.4 Standards and Models

Philosophers including Aristotle, had speculated on the nature and origins of colour many centuries before Isaac Newton used a prism to separate sun-light into its spectral components (➋ *Fig. 4.21*). Nowadays, it is common practice to use a diffraction grating instead, as this is more efficient. The theory of diffraction gratings is given in [2, 16].

Decomposing light into its spectral components was a crucial step towards understanding the real nature of colour as a psycho-physical phenomenon. By using a second prism to reconstruct a single light beam, it is possible to modify the spectrum at will and thus create whatever colour we desire (➋ *Fig. 4.22*. Also see lighting-viewing method number 44,

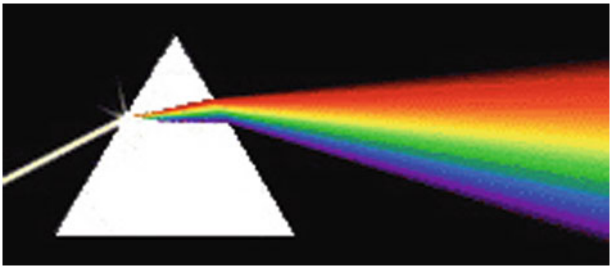


Fig. 4.21

Decomposing white light into its spectral components using a prism

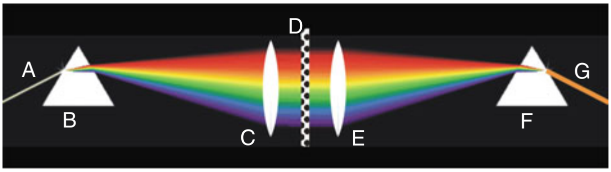


Fig. 4.22

Generating light with an arbitrary spectrum. Prism B decomposes the incident beam (A, white light). Lens C forms a set of parallel rays, which are then filtered by the transparent mask (D). Lens E refocuses the rays, which are then recombined by prism F to form a coloured emergent beam (G). The optical system is symmetrical about the mask (monochrome grey), which may be a photographic transparency or LCD spatial light modulator

Chap. 40) The physics of colour is well understood but its neurophysiological-psychological basis is not. The reason is it is not possible to measure the internal response within the human brain to a colour stimulus. Hence, we have to resort to indirect means of probing its internal working. One of the prime experimental techniques used in Colour Science is that of visually comparing a test sample to a set of standard reference surfaces. This is called *colour matching*. This section discusses both mathematical models of colour vision. These have been established in large part by colour matching experiments. In addition, we shall describe a pragmatic method of describing a given colour by visually comparing it to a set of standard reference surfaces that have previously been placed on an ordered scale.

4.4.1 Tristimulus Theory

At the beginning of the nineteenth century, Thomas Young observed that three primary colours (of either paint or light beams) could be added to create a wide variety of other colours. This led him to formulate the *Tristimulus Theory*. This postulates that any colour (C) can be obtained by mixing (adding) light from three primary sources. Let us call these P_1, P_2 and P_3. (Originally, these were taken to be red, green and blue.) Although nearly all visible colours can be matched by additive mixing, some cannot. Suppose that one of the primaries (say, P_3) is added to C and the resulting mixture can be matched by a combination of P_1 and P_2. Then, C may be considered to have a positive mixture of P_1 and P_2, and a negative amount of P_3. By allowing subtractive mixing of colours, a greater range of colours can be created. We must be careful to distinguish between superimposing coloured light beams (additive mixing) and mixing paint or other pigments (subtractive mixing). The primaries used in these two situations to create the same colour are quite different.

A *colour model* is a mathematical aid to understanding human colour vision and is usually expressed in terms of a three- or four-element vector. Colour models are inherently approximate; each one is valid only over a finite range of conditions and has its own special merits and disadvantages. In this respect, a colour model is no different from any other model of a complex physical system Most colour models were developed to assist in the production/reproduction of colour images, on paper, film or video. Ideally, a colour model should provide

- A means of describing visible light or surface colour, as a mixture of fundamental colour components, called *primary colours.*
- Clues as to how a given colour can be reproduced using physically available media: pigments, paints, inks, light sources, optical filters, phosphors.
- A framework that relates colours to each other, so that, for example, complimentary and harmonising colours can be identified. Many such frame-works are based on a geometric representation, such as a segmented wheel.
- A systematic nomenclature enabling colour information to be communicated between people who share no direct visual history. For example, it should be possible to purchase paint and wallpaper by telephone and be totally sure that they will harmonise with one another.

Thirty-seven different models are mentioned at [23], reflecting the different purposes they are intended to serve, as well as advances in knowledge. So called *Colour Order Systems* are represented within a standardised geometric framework, while surface colours are judged visually under standardised lighting and viewing conditions. We shall describe some of these below. However, many colour models are irrelevant to us, as industrial inspection systems are only expected to verify that colours that have previously been seen are correctly placed on the object being examined. This presents quite different requirements compared to reproducing colours and harmonising the parts of a multi-colour ensemble. For Machine Vision, the usual requirement is for a system that can compare the colours in two scenes and identify differences that a person would find significant. Differences in the spectral signatures of defined points on two surfaces can be measured with ease using a spectrophotometer [26]. Machine Vision demands the analysis of large areas, not single points. We are therefore constrained to using available technology, most commonly CCD/CMOS cameras that generate RGB signals.

In this section, we shall also describe how colour standards have been established, using so called colour atlases. These provide a systematic nomenclature based on a set of precisely tinted swatches, which can be matched (normally by eye) to artefacts/features of interest. Both "general purpose" and specialist colour atlases exist. The latter are intended for diverse activities, such as printing, horticulture, grading fruit, baking and dermatology, fashion, furnishing and interior decorating.

4.4.1.1 RYB Model

Historically, the RYB model is the oldest, having been established empirically by painters, long before colour scientists discovered the response characteristics of the retinal cone cells. It is based on a subtractive set of primary colours: red, yellow and blue (➜ *Fig. 4.23a*). The RYB

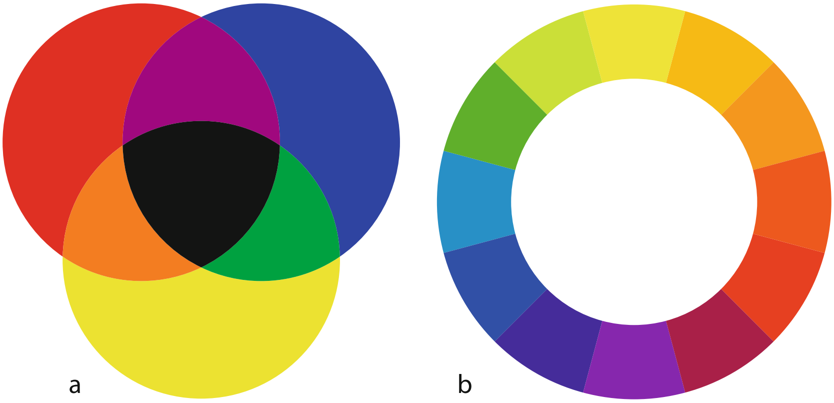


Fig. 4.23
RYB Colour Model (a) Colour subtraction, creates violet, orange and green. (b) Colour wheel

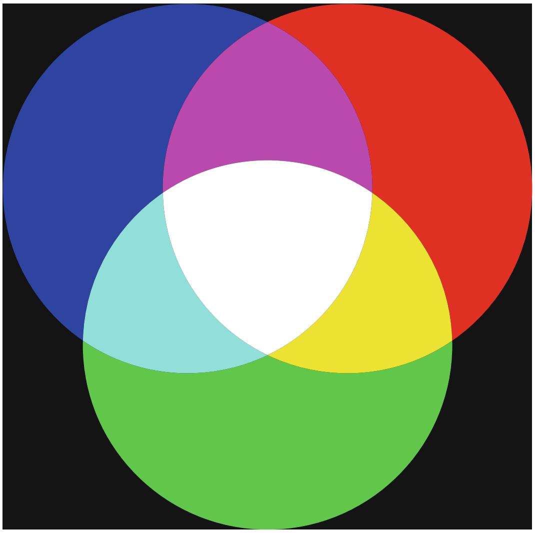
model places the primary colours and the secondary colours (violet, orange, green) around a circle, a so-called *colour wheel* (*Fig. 4.23b*). This allows complementary and harmonising pairs of colours to be identified. The RYB model is not able to reproduce bright shades of green, cyan or magenta, since these cannot be created from any combination of red, yellow, and blue paints. Furthermore, mixing a colour with its complement (i.e., the sector 180° around the RYB colour wheel), produces a shade of brown, rather than the ideal, black. As far we are concerned, the importance of the RYB model is limited to historical interest. It is still used when teaching children to paint.

4.4.1.2 RGB/sRGB

James Clerk Maxwell first devised the RGB colour model and demonstrated colour addition in an experiment conducted with Thomas Sutton The latter photographed a tartan ribbon three times, first with a red filter, then a green filter and finally a blue filter, placed in the optical path. The three images so obtained were then projected onto a white screen with three different projectors fitted with the same red, green and blue colour filters. When they were properly registered, the three monochrome images reproduced a full colour image of the tartan. Colour addition of RGB primaries is represented diagrammatically in *Fig. 4.24*.

There are several RGB colour models, based on different definitions of three primaries: "red," "green" and "blue." These can be added together in various proportions to reproduce other colours. One commonly used version is *sRGB* (*standard RGB*), which was originally devised by Hewlett-Packard and Microsoft. It has since been widely applied by companies including Intel, Pantone and Corel. sRGB defines red, green, and blue primaries in terms of the CIE xy chromaticity coordinates (see below), as follows

Red is at [0.6400, 0.3300]
Green at [0.3000, 0.6000]
Blue is at [0.1500, 0.0600]
White is at [0.3127, 0.3290]



■ Fig. 4.24
Additive mixing of RGB primaries

sRGB also defines a non-linear transformation between the numbers stored in an image and the intensities of the primaries (light beams) that are actually added together. This enables sRGB to provide the basis for efficient short-integer computation, while displaying images that are acceptable to a human observer. sRGB was designed to match the performance of CRT monitor technology as it existed in 1996. The sRGB model does not naturally fit more modern hardware, such as liquid crystal displays, digital cameras, and printers, although they incorporate compensating circuitry/software to accommodate the differences.

sRGB remains of great importance, despite the fact that it is sometimes criticised, especially by printers because it has a limited colour gamut compared to CMYK (see below). Another variant, Adobe RGB is sometimes preferred. There is one feature of the sRGB colour model that makes it very important for Machine Vision: it is ubiquitous, being the basis for a large majority of colour cameras. Even cameras that generate signals conforming to other models, such as HSI or YUV, begin by sensing RGB data. The reason for this is simple: video cameras work on the same principle as that first devised by Maxwell, using colour filters to separate the RGB channels. The overwhelming forces of finance, availability, versatility and familiarity combine to make RGB the dominant colour model for Machine Vision. The RGB model is the nearest we currently have to one based on direct measurement of the spectrum. Conceptually, it is attractive because it provides a direct analogue of the colour sensing achieved by retinal cone cells. Moreover, other models, including HSI and YUV, can be related directly to the RGB model.

4.4.1.3 CMYK

CMYK (cyan, magenta, yellow, black) is a subtractive colour model. Its great importance arises from the fact that it is used extensively in colour printing. Magenta plus yellow produces red, magenta plus cyan makes blue and cyan plus yellow generates green. Theoretically, a mixture of

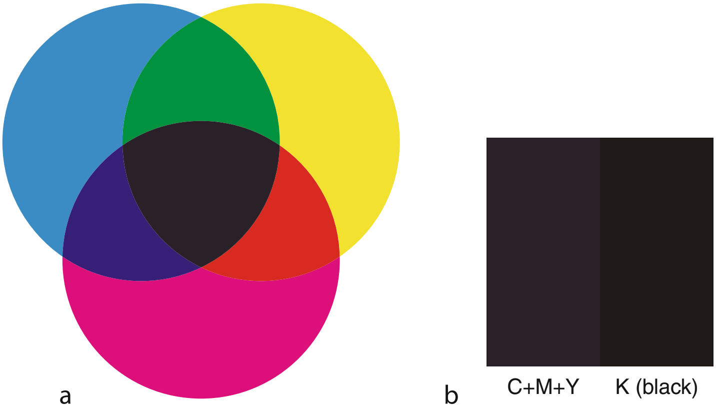


Fig. 4.25
CMYK (a) CMY colour subtraction. (b) Comparison of "black" produced by CMY and K. A "rich black" is sometimes produced by imprinting all four of the CMYK channels

ideal cyan, magenta, yellow pigments, printed together on white paper, would result produce black. However, in practice, the colour generated by mixing the CMY primaries is not ideal, being dark murky brown, rather than black. Using four-colour (CMYK) printing generates a better result with high contrast (➋ *Fig. 4.25*). To obtain really high quality images, orange and green imprints are added, requiring a six-stage printing process designated as CMYKOG. Hexachrome® (Pantone Corporation) was created in 1994 to provide a wider colour gamut for the printing industry and is claimed to provide brighter greens, more vibrant purples, and more accurate rendering of skin tones.

When printing computer-generated (RGB) images, they must be converted to CMYK format first. This process is imperfect, because CMYK and RGB have different gamuts. It is possible to produce an invertible conversion from RGB to a subset of CMYK, although conversion in the reverse direction is not always possible. It is a common experience to see quite different colours on a computer screen and on a paper print-out. Pseudo-code for converting between RGB, CMYK and a variety of other colour models is given at [13].

4.4.1.4 HSI (Hue, Saturation, Intensity)

Consider the RGB colour space, assuming that each of the RGB variables is limited to the same range: [0,W] (➋ *Fig. 4.26*). This defines a cube of side W in which

White is [W,W,W];	Black is [0,0,0]	
Pure red is [W,0,0];	Pure green is [0,W,0];	Pure blue is [0,0,W]
Pure cyan is [0,W,W];	Pure magenta is [W,0,W];	Pure yellow is [W,W,0].

Notice that we have emphasised that these are pure colours. Points close to [W,W,0], such as [0.95 W, 0.95 W,0.05 W], would probably be called "yellow" too. The plane defined by the points [W,0,0], [0,W,0], [0,0,W] intersects this cube to produce a triangle, called the *Colour Triangle*, or *Maxwell Triangle*. The point P is at the intersection of the vector [R,G,B] and the

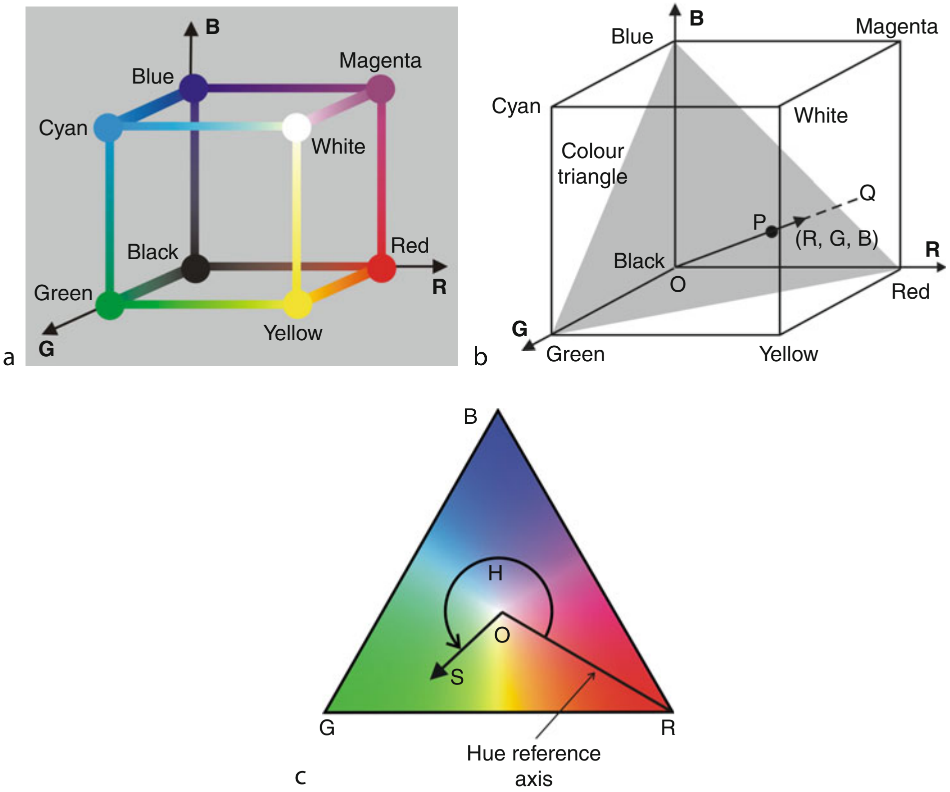


Fig. 4.26

The colour cube in RGB space and the Colour triangle. (a) Colour cube. (b) Relationship between RGB space and the colour triangle. (c) The colour triangle represents hue (H) and saturation (S) but not intensity (Colours are approximate)

Colour Triangle. This line is extended if necessary. Q is the point of intersection of that vector and the external surface of the cube. P can lie on any one of three faces, defined by the equations

$$R = W; \quad G = W; \quad B = W.$$

Apart from points very close to the origin, O, where scotopic (low light) vision prevails, all points along the line OQ have the same chromaticity (i.e., the same hue and saturation) but with different intensities. That is, they are perceived as being the same colour. The orientation of the line OQ can be determined by two angles, not shown, or the position of the point P in the colour triangle. Within the Colour Triangle, the chromaticity can be defined using polar coordinates (*H* and *S*) or Cartesian coordinates.

Conversion between HSI and RGB is invertible using the following transformations:

$$H = \cos^{-1}\left((2R - G - B)/\left(2\sqrt{\left(R^2 + G^2 + B^2 - GB - BR - RG\right)}\right)\right)$$

$$S = 1 - 3\mathrm{MIN}(R, G, B)/(R + G + B)$$

$$I = (R + G + B)/3.$$

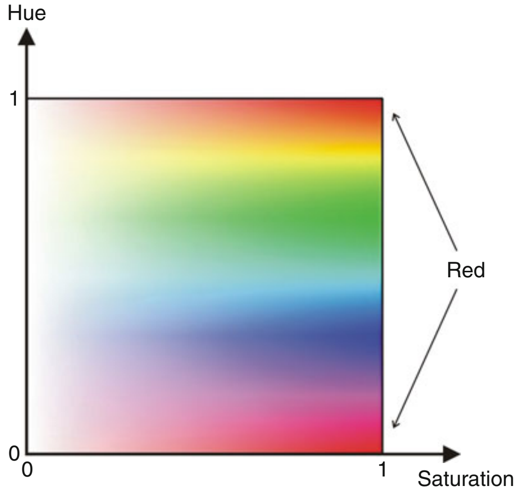


Fig. 4.27
HS plane rendered in colour. Notice that "*red*" is split into two distinct regions (Colours shown are approximate)

Later, we shall find it convenient to plot HS using Cartesian coordinates (➋ *Fig. 4.27*). This merely represents a geometric warping of the Colour Triangle.

4.4.2 CIE 1931 Colour Model

The CIE Colour Model [3] was one of the first mathematically defined colour models. It has been up-dated since its inception to accommodate new thinking and knowledge. It forms the basis upon which many other colour models have subsequently been defined. However, it remains of great importance in its own right, because it is based on direct measurement of the human visual system. The CIE XYZ Colour Model was derived from experiments performed in the late 1920s by W. D. Wright and J. Guild and is based firmly on the tristimulus theory. The CIE 1931 Chromaticity Diagram is central to understanding this colour model (➋ *Fig. 4.28*).

Despite the ubiquitous appearance of this diagram in text-books on photography, video and Colour Science, its conceptual and experimental basis is often not explained clearly. As a result, it is frequently misunderstood and mis-used. The model is based on the assumption that there are three separate colour channels in the human visual system and that their responses to a given colour input, *C*, are values that we shall denote by *X*, *Y* and *Z*. These parameters are actually the amounts of monochromatic light at wavelengths of 0.7000 μm, 0.5461 μm, and 0.4358 μm that are needed to match *C*. Notice that *X*, *Y* and *Z* correspond to specific instances of red, green and blue, respectively. The spectral responses of these hypothesised colour channels are shown in ➋ *Fig. 4.13* and are closely related to the responses of the L, M and S cones in the eye. In order to normalise for varying intensity, it is sensible to define three new variables thus:

$$x = X/(X + Y + Z)$$

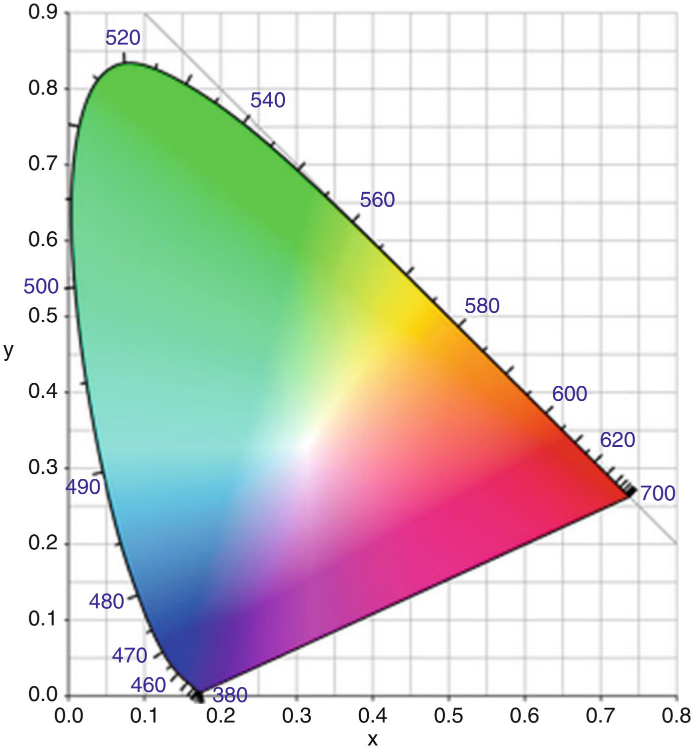


Fig. 4.28
CIE (1931) Chromaticity diagram. This represents the full range of colours that can be seen by a human being. The colours shown here are approximate as the range of visible hues that can be represented in printing (gamut) is smaller than the diagram requires. The numbers around the edge indicate the wavelengths of the spectral colours, measured in nanometres. The x and y axes are explained in the text

$$y = Y/(X + Y + Z)$$

$$z = Z/(X + Y + Z)$$

Since

$$x + y + z = 1$$

we need only specify two of these numbers in order to calculate the third one. The famous CIE Chromaticity Diagram (➋ *Fig. 4.28*) has an inverted U-shaped arc, which is the locus of [x, y] values that match the pure spectral colours. (Numbers around this arc indicate the wavelength.)

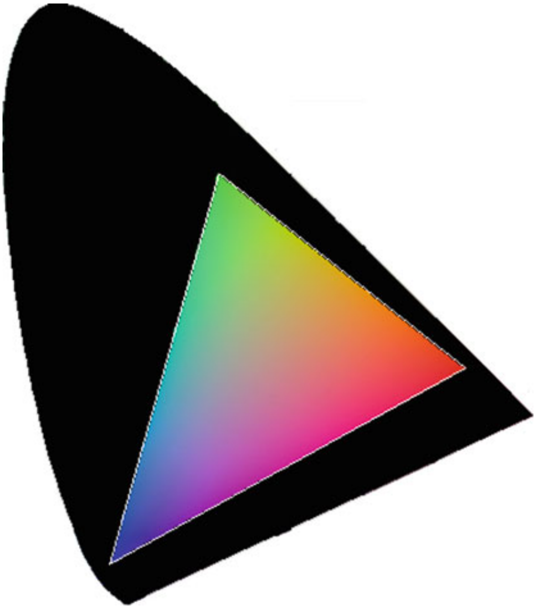
- The straight line across the bottom of the inverted U is called the *Line of purples*. It represents the non-spectral colours that can be obtained by mixing red and blue light. Despite its representation in the diagram, the line of purples is not a hard limit and should properly be drawn as having a fuzzy boundary.

- Colours on the periphery of the locus are saturated; colours become progressively desaturated and tend towards white near the middle of the plot. The point at

  $$x = y = z = 0.333$$

  represents white.
- Suppose we choose three points within the shaded region, to define three primaries. Any colour within this triangle can be created by adding various proportions of those primaries. The range of colours that can be reproduced in this way is called the gamut.
- The gamut of colours that can be reproduced on a video monitor, or by photography, is often defined by drawing a triangle in the CIE Chromaticity Diagram. See, for example, ➲ *Fig. 4.29*.
- The 1931 CIE chromaticity diagram is not *perceptually uniform*. That is to say the *area* of any region of the plot does not correlate at all well with the number of perceptually-distinguishable colours in that region. In particular, the vast area of green-turquoise inaccessible to televisions and monitors is not quite as serious as it appears.

Since it is so important, the CIE Chromaticity Diagram has often been attributed with meaning that was not intended when it was first introduced. One potentially misleading use has been the addition of spots to indicate colours of familiar objects, such as lettuce, tomato, etc. By measuring one point, on one tomato, on one particular day, a set of XYZ coordinates was found and a point labelled “tomato” was added to the diagram. This was intended, quite correctly, to assist people to understand the diagram. The danger is that it can be



■ Fig. 4.29
Gamut for a computer/television display; colours in the black region cannot be displayed properly

misinterpreted as meaning that the colour set with the name *"tomato"* can be identified with a point. It cannot! *"Tomato"* is a set and should therefore be represented on the CIE Chromaticity Diagram as a blob. If this convention were to be adopted, we could, in theory, use membership of that set of points as a basis for colour recognition. As this would not lead to a very efficient recognition procedure, we shall seek other methods for doing this.

4.5 CIE L*a*b* (CIELAB)

This is one of the most complete colour models yet devised. It was developed for by the CIE (*Commission Internationale d'Eclairage)* in an attempt to describe all of the colours visible to the human eye. Notice that $L^*a^*b^*$ and *Lab* are two different colour models; the asterisks indicate the fact that the parameters L^*, a^* and b^* are derived from parameters derived for the earlier model and denoted by *L, a* and *b* [19]. These are, in turn, based on a mathematical approximation to the Munsell colour ordering system (see below), which was devised empirically after detailed colour matching experiments.

L* measures lightness between black (L* = 0) and white (L* = 100)
a* indicates position on the magenta – green continuum ($a^* < 0$ indicates green; $a^* > 0$ indicates magenta)
b* indicates position on the yellow – blue continuum ($b^* < 0$ indicates blue; $a^* > 0$ indicates yellow)

Since it purports to represent all colours perceived by human beings, the full gamut of colours allowed by the $L^*a^*b^*$ model cannot be represented perfectly accurately, either on paper or a video display. It can only be represented properly in a three dimensional space. However, since L^* measures lightness, just a few a^*b^* plane slices allow the whole gamut to be represented reasonably satisfactorily.

The $L^*a^*b^*$ colour model was devised specifically to serve as a device independent, absolute reference. It is one of the most important models for Colour Science but it does not provide us with any practical help in building industrial inspection systems.

4.6 YUV/YIQ

The YUV model is used to define the PAL and NTSC television broadcasting standards, which are both widely used throughout the world. YUV defines a colour space with one luminance (brightness Y) and two chrominance components (U and V). These can be calculated from the RGB parameters using the following equations:

$$\mathrm{Y} = 0.299\mathrm{R} + 0.587\mathrm{G} + 0.114\mathrm{B}$$

$$\mathrm{U} = 0.492(\mathrm{B} - \mathrm{Y}) = -0.147\mathrm{R} - 0.289\mathrm{G} + 0.436\mathrm{B}$$

$$\mathrm{V} = 0.877(\mathrm{R} - \mathrm{Y}) = 0.615\mathrm{R} - 0.515\mathrm{G} - 0.100\mathrm{B}$$

where, R, G and B are assumed to be in the range [0,1]. Another standard, YIQ, is closely related to YUV and can be calculated thus:

$$Y = 0.299R + 0.587G + 0.114B (\text{i.e., same as for } YUV)$$

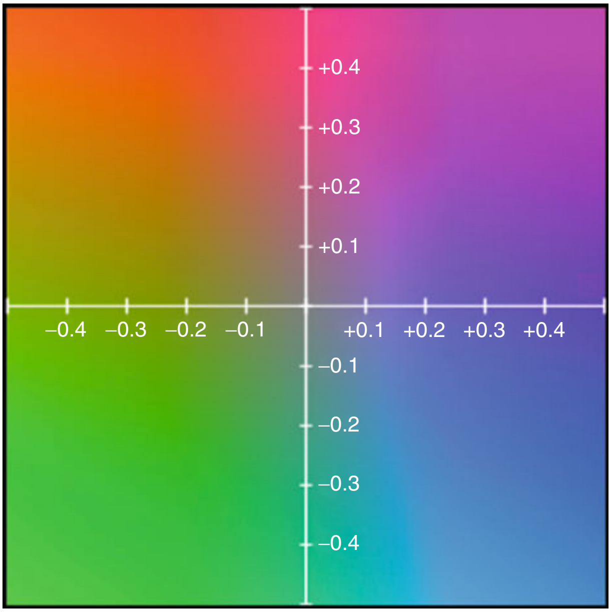


Fig. 4.30
UV colour space. (In this context UV does *not* refer to ultra-violet.) Abscissa, U. Ordinate, V. IQ colour space is similar but the coordinate axes are rotated counter-clockwise by 33°

$$I = 0.735514(R - Y) - 0.267962(B - Y) = 0.595716R - 0.274453G - 0.321263B$$

$$Q = 0.477648(R - Y) + 0.412626(B - Y) = 0.211456R - 0.522591G + 0.311135B$$

The importance of YIQ has diminished in recent years as it was formerly used to define the NTSC television standard, which is now based on the YUV model. YUV and YIQ are compatible with black and white analogue systems. (UV/IQ are simply discarded.) The calculations given above can be performed in real time, using low cost analogue or digital hardware. IQ and UV represent different coordinate systems in the same plane (➋ *Fig. 4.30*).

YUV and YIQ do not define absolute colour spaces. Notice that the actual colour generated depends on the RGB colourants used to display the image. That is, the colour generated by given values of YUV, or YIQ, is only predictable if known RGB colourants are used in the display hardware. Of particular importance for us in building colour recognition systems is the fact that any two sets that are separable in either YUV or YIQ space are also separable in RGB space.

4.7 Munsell Colour Standard

The Munsell System is widely recognised as providing a standard for colour notation. It is used throughout the world in painting, printing, advertising, commerce, manufacturing, science and education. It provides a means for selecting, specifying, and controlling colour. The Munsell System is recognised as a standard system of colour specification in Standard Z138.2

of the American National Standards Institute; Japanese Industrial Standard for Colour, JIS Z 8721; the German Standard Colour System, DIN 6164; and several British national standards.

In the Munsell System, every colour is represented by three numeric quantities: *hue (H)*, *value (V)* and *chroma (C)*. At the beginning of the twentieth century, A. H. Munsell first produced numerical scales, with visually uniform steps, for each of these attributes. The *Munsell Book of Colour* [21, 28] displays an ordered collection of coloured swatches, each of which is identified numerically by a three-element vector [H,V,C]. It is used in a very simple way: the colour of a surface can be identified by visually matching it to the swatches. Illumination and viewing conditions are closely controlled during the matching process. Proprietary viewing booths are available for this purpose (➔ *Fig. 4.16*) [14].

Hue is that attribute of a colour by which we normally attribute names, such as red, yellow, green, blue, purple. Munsell exploited the fact that there is a natural order of hues. We can mix paints of adjacent colours in the series just mentioned to obtain intermediate colours. For example, red and yellow light may be mixed in varying proportions to generate all of the hues on the red-orange-yellow continuum. Similarly, we can obtain intermediate colours by mixing yellow and green, green and blue, blue and purple, or purple and red. Notice that the sequence is cyclic. Hence, it is natural to arrange these colours around a circle. Munsell termed red, yellow, green, blue, and purple the *principal hues* and therefore placed them at equal intervals around the circle. He then interposed the five intermediate hues: yellow-red, green-yellow, blue-green, purple-blue and red-purple between them. Hence, the Hue circle has a total of making ten hues (➔ *Fig. 4.31*).

Munsell arbitrarily sub-divided the hue circle into 100 steps of equal visual change. Zero is referred to the edge of the red sector. Hence, *hue* can be represented by an integer in the range [0, 99]. An alternative representation uses a letter (sector name) followed by an integer

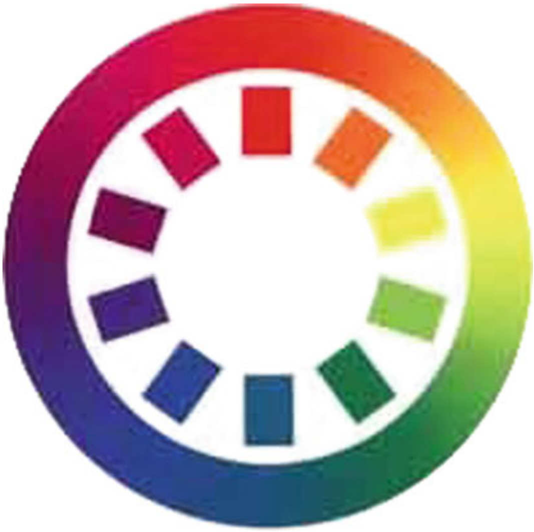


Fig. 4.31
The Hue circle

(position within the sector). The hue in the middle of the red sector is called *five red (5R)*. Hence orange is 5YR and 8BG is blue-green (blue turquoise) (see *Fig. 4.32*).

Minor adjustments in the nomenclature for hue have been introduced to compensate for an anomaly that exists in the blue region of the Munsell System. Awkward features like this are not unusual in Colour Science; as we indicated earlier, no colour model yet devised perfectly matches human vision.

Value indicates the lightness of a surface. Zero (0) represents pure black and 10 pure white. Achromatice colours (or “neutral” colours: black, white and grey) have zero hue. The term *chromatic colours* refers to colours where the hue (H) is not zero. The value scale applies to both chromatic and achromatic colours. The value scale is illustrated for a series of neutral colours in *Fig. 4.33*.

Chroma measures the strength of a colour in terms of the degree of departure from that achromatic colour with the same value. Colours with high chroma are termed *highly saturated*,

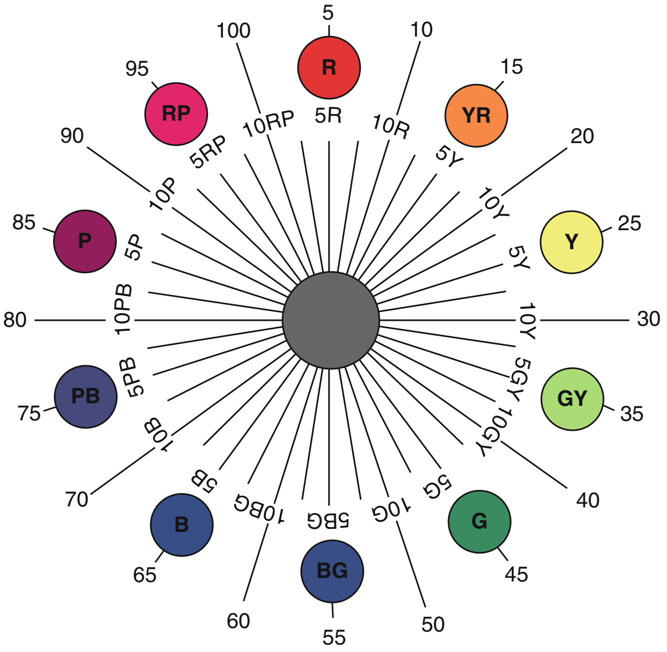


Fig. 4.32
Munsell Hue designations

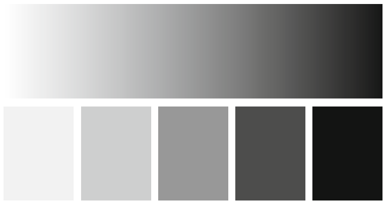


Fig. 4.33
Munsell value parameter, achromatic colours

or *vivid*. To understand the nature of chroma, imagine mixing yellow and grey paints that have the same value. Starting with pure grey and gradually adding more yellow until saturated yellow is achieved, the chroma of the paint mixture increases progressively (➋ *Fig. 4.34*).

The chroma scale was designed to be visually uniform, although the units are arbitrary. Achromatic colours have zero chroma but there is no arbitrary maximum. As new pigments have been manufactured, Munsell colour swatches with higher chroma have been made.

4.7.1 Munsell Notation

The complete Munsell notation is written as follows:

- *hue value/chroma*

Saturated red with a hue of 5R, a value of 6 and a chroma of 14, is represented as 5R 6/14. When greater precision is needed, decimal parameters are used (e.g., 5.1R 5.2/12.9.) The Munsell colour parameters (hue, value and chroma) are often presented graphically using cylindrical coordinates (➋ *Fig. 4.35*). Hue is a cyclic parameter and is therefore equated with

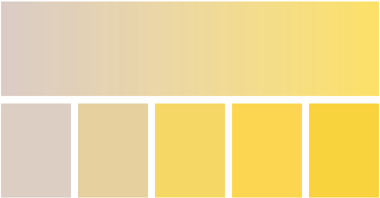


Fig. 4.34
Varying chroma

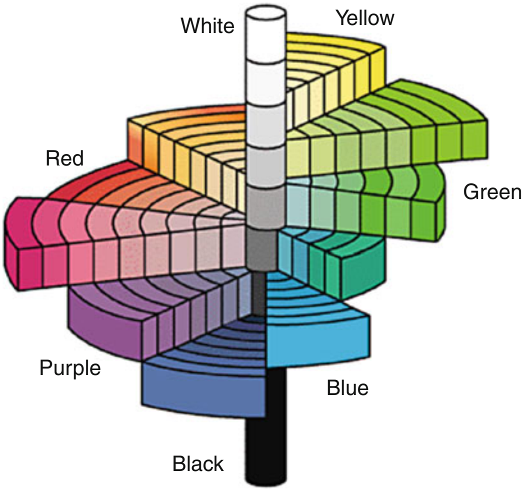


Fig. 4.35
Munsell colour solid

angular position. Chroma is measured by the radius from the central axis, which represents the value scale. This leads to the *Munsell Colour Solid.* This is not a perfect cylinder, since the scales are not defined on an absolute scale. Moreover, the model is modified to include only those pigments for which a swatch can be made. As a result, the Munsell colour solid is irregularly shaped figure that is enlarged from time to time, as new pigments are developed.

4.7.2 Colour Matching

The apparent colour of a surface depends on the spectrum and polarisation of the illumination, the directions of illumination and viewing, the reflection, polarisation and scattering characteristics of that surface, its surrounding area/background and the state of adaptation of the eyes of the observer. To ensure precise colour matching, it is standard practice to view specimens illuminated by daylight from a lightly overcast northern sky in the northern hemisphere. An artificial source that produces light with a very similar spectrum may be used instead. (Lamps providing an approximation to daylight have been available for photographers and television program makers for many years.) Viewing booths that employ controlled artificial daylight and are normally recommended for colour matching because the lighting conditions that they produce are much more consistent over the working day than natural light. To reduce the effects of glinting to a minimum, the surface to be examined should be viewed along its normal and illuminated at 45° to the normal. Before a person is allowed to match colours, he/she should be tested to detect obvious colour deficiencies. (These are quite common, occurring at a rate of about 5% for men and 2% for women.) Among normal observers, the ability to match colours varies significantly and gradually diminishes with age. In this area of Colour Science, it is not possible to guarantee precise measurements, because subjective judgements are involved. Colour swatches are available with matt and glossy surfaces in the Munsell System.

4.7.3 Pantone® Colour Reference System

The Pantone Matching System® is used very extensively in the printing industry. It is used by artists and commercial printers to select, specify and match colours. Given a colour sample, visual matching, against a large set of standard swatches, allows the ink formulation that will reproduce it to be identified. Numerous company and other organisation logos are created with specific reference to the Pantone system. It originally included about 500 colours and has since been expanded to include over 3,000 colours, together with their CMYK ink formulations. Recently, Pantone Inc. introduced a six-colour printing system called Hexachrome®, which includes cyan, magenta, yellow, black, orange and green inks. This substantially increases the colour gamut for reproducing printed photographic images and simulated spot colours. The Pantone system also includes many special colours to be specified, including metallic finish and florescent paints/inks. Recently, the Pantone system has also been linked to screen-based colours (RGB). It has also been used to specify the colours used in flags and company logos. The Union Jack (flag of the United Kingdom) contains PMS186(red) and PMS280 (blue), while the BP logo consists of elements with colours PMS355 (dark green), 368 (light green) and yellow (109 coated, or 108 uncoated). Knowing the Pantone index numbers enables a person to order paints or inks to reproduce their colours accurately.

The Pantone Matching System is the intellectual property of Pantone Inc. [24]. Since it is a proprietary system, PMS lacks the transparency that the Munsell system enjoys. This is frequently cited as a reason why Pantone colours are not supported in open source software, such as GIMP, and are not often found in low-cost software. The Pantone company asserts that its collection of colour swatches is a database that could be copyrighted. However, Pantone palettes supplied by other companies, such as paint and printer manufacturers, are freely distributed, without undue restriction as to their use. The Pantone Matching System is intended specifically for the print industry, whereas the Munsell system is more appropriate for other, more general applications. For example, to specify the range of colours that might be expected on a manufactured food product, the Munsell system would normally be preferred. However, to print a photograph of it, the Pantone system would be used. The Munsell system has been matched, as far as is possible, to mathematically based colour models, such as $L^*a^*b^*$.

4.7.4 Colour Gamut

The following notes indicate the sizes of gamuts for a number of colour systems

- *Photographic film* This is one of the best systems for reproducing colour as it has a large gamut. It far exceeds the gamut for television.
- *Laser light projectors* produce images containing just a few nearly monochromatic components, which are therefore highly saturated. However, it is more difficult to generate less saturated colours.
- *CRT video displays* have a gamut that covers a significant portion of the visible colour space. The gamut is limited by the phosphors built into the display screen. In practice, the video sensor also has a limiting effect, as it is the intersection of the gamut of the camera and that of the monitor that determines the gamut of a complete camera-display system.
- *Liquid crystal display (LCD) screens* filter the light emitted by a backlight. Hence, the gamut of an LCD display based on fluorescent tube back-lighting is smaller than that for a CRT screen, while one using LED back-lighting has a larger gamut.
- *Broadcast television* using a CRT display has a gamut limited by transmission coding, rather than the screen. High definition television (HDTV) is better, but the gamut is still less than that for a computer display employing the same display screen.
- *Paint* can produce some highly saturated colours. Artists have long exploited the huge range of pigments available from animal, plant and mineral sources. More recently, synthetic dyes have been used. Paints are available with highly saturated colours that cannot be generated by video displays. Paint colours for industrial and commercial applications are often specified with reference to the Pantone® Matching System [24].
- *Spot Colour Printing* refers to a process in which each colour is created by a different ink. It is most cost-effective when there are only three or different colours, or when special-purpose inks (e.g., UV reflective, fluorescent, photo-chromic, chemically sensitive, etc.) are required. The same remarks apply here as for paint.
- *Process Colour Printing* is used for “general purpose” image reproduction in newspapers, books, magazines, etc. and usually relies on CMYK, CMY or CMYKOG imprints. The gamut for CMYK printing is smaller than that for a video monitor.

A Java program for calculating display gamuts is on-line available at [4].

References

1. Byrne A, Hilbert D. Glossary of Color Science. (1997) http://tigger.uic.edu/~hilbert/Glossary.html. Accessed 1 December 2009
2. CAC Colour standards and models. Color Academy. http://www.coloracademy.co.uk/Online/index.html. Accessed 21 February 2011
3. CIE Colour Model, Wikipedia. (2007) http://en.wikipedia.org/wiki/Color_model. Accessed 1 December 2009
4. CGA Color Gamut Applet, Rochester Institute of Technology. http://www.cs.rit.edu/~ncs/color/a_chroma.html. Accessed 1 December 2009
5. Color Science, IBM Research. http://www.research.ibm.com/image_apps/colorsci.html. Accessed 1 December 2009
6. Color, Wikipedia. http://en.wikipedia.org/wiki/Color. Accessed 1 December 2009
7. Colour Academy http://www.coloracademy.co.uk/. Accessed 1 December 2009
8. Colour, Color matters http://www.colormatters.com/science.html. Accessed 1 December 2009
9. Colour Science, Colour-Science AG. http://www.colour-science.com/. Accessed 1 December 2009
10. Crystal D (1997) The Cambridge encyclopedia of language, 2nd edn. Cambridge University Press, Cambridge. ISBN 0 521 55967-7
11. Dickinson C, Murray I, Carden D (eds) (1997) John Dalton's colour vision legacy. Taylor & Francis, London. ISBN 0-7484-010-8
12. Douma M (curator), WebExhibits, Light bulb. Cause of Color. http://www.webexhibits.org/. Accessed 22 September 2008
13. EAS Color conversion math and formulas, EasyRGB. http://www.easyrgb.com/math.html. Accessed 1 December 2009
14. GEN Colour Matching Booth, Geneq. http://www.geneq.com/catalog/en/colormatcher_color_match.html. Accessed 1 December 2009
15. GRA Grassmann's Law, Wikipedia. http://en.wikipedia.org/wiki/Grassmann%27s_law_(optics). Accessed 1 December 2009
16. HOR Diffraction Gratings, Horiba Scientific Ltd. http://www.jobinyvon.com/usadivisions/OOS/oos1.htm. Accessed 1 December 2009
17. Hughes JF, Bell JE, Doppelt AM. http://www.cs.brown.edu/exploratories/home.html. Accessed 25 September 2008
18. Introduction to Colour Science. http://www.techmind.org/colour/. Accessed 1 December 2009
19. LAB Lab Colour Space, Wikipedia. http://en.wikipedia.org/wiki/Lab_color_space. Accessed 1 December 2009
20. Lamb T, Bourriau J (1995) Colour art & science. Cambridge University Press, Cambridge. ISBN 0-521-49963-1
21. MUN Munsell book of colour. http://en.wikipedia.org/wiki/Munsell_color_system. Accessed 1 December 2009
22. Munsell Color Science Laboratory, Rochester Institute of Technology. http://www.cis.rit.edu/mcsl/. Accessed 1 December 2009
23. MUS Colour Models, Color model museum. (2000) http://www.colorcube.com/articles/models/model.htm. Accessed 1 December 2009
24. PAN Pantone matching system, Pantone. http://www.pantone.com/aboutus/aboutus.asp?idArticle=53. Accessed 1 December 2009
25. Biv RG Color Science. http://www.midnightkite.com/color.html. Accessed 1 December 2009
26. SPE Spectrophotometry, Wikipedia. (2008) http://en.wikipedia.org/wiki/Spectrophotometry. Accessed 1 December 2009
27. The science of light, Teachers' Lab. http://www.learner.org/teacherslab/science/light/lawslight/index.html. Accessed 1 December 2009
28. TPS Munsell book of colour, Trade print supplies. http://www.tradeprintsupplies.co.uk. Accessed 1 December 2009
29. Hinton's Lilac Chaser, Bach M. http://www.michaelbach.de/ot/col_lilacChaser/index.html. Accessed 21 February 2011

5 Light and Optics

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_5,

5.1 Basic Concepts

The image acquisition sub-system is responsible for collecting information in optical form about the scene that is to be examined and then projecting it onto the image sensor (the camera's "retina"). It is important that the very best image possible is presented to the camera; lack of attention to the design of the optical "front end" is tantamount to throwing away information and/or introducing noise into the system deliberately. The well-worn adage "*Rubbish in, rubbish out*" is worth remembering. It is also worthwhile bearing in mind that, once an effective optical system has been set up, good quality images will be produced indefinitely without further expense (apart from minimal costs for regular cleaning of optical components). In many cases, complicated digital processing may be avoided altogether. Good optical design will very often make the overall system faster, cheaper and more reliable.

5.1.1 Image Acquisition Sub-System

➋ *Figure 5.1* shows how information flows in the optical "front end" of a typical Machine Vision system; information is carried via electro-magnetic radiation in the visible wave band

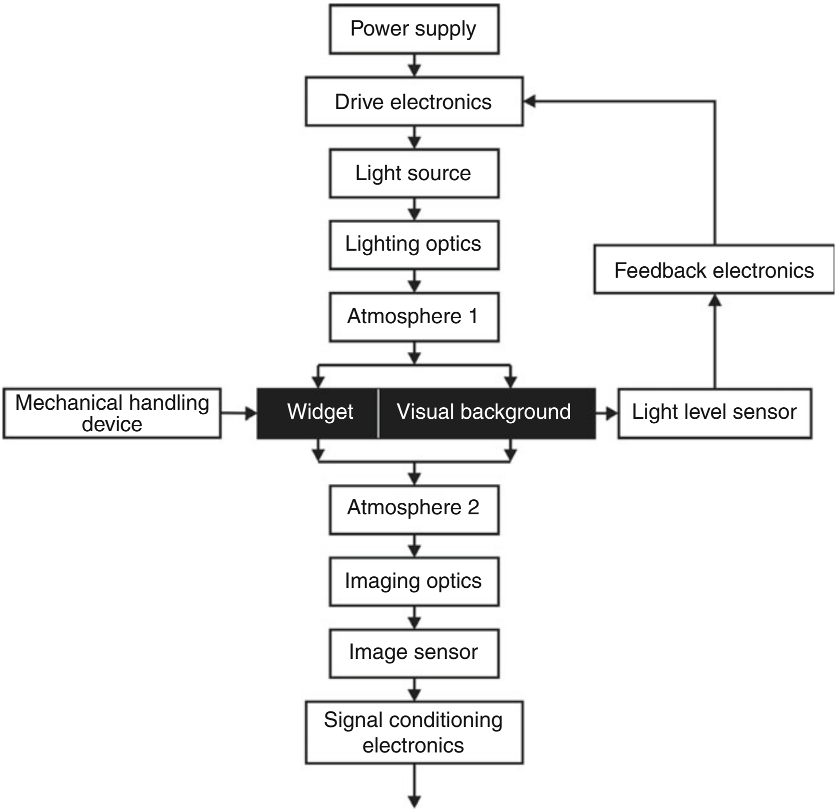


Fig. 5.1
Information flow in the image acquisition sub-system of a Machine Vision system

(roughly 0.36–0.74 μm). Occasionally, we may use the ultra-violet, infra-red, or x-ray regions of the spectrum instead. An imaging sub-system may include a variety of components: lenses, filters, mirrors, prisms, polarisers, gratings, apertures, fibre optics, beam-splitters, diffusers, protective windows, light absorbing backgrounds and shrouding. This diagram also indicates that the image presented to the image sensor (the camera) is affected by a series of what might be regarded as secondary factors: the power supplied to the lamps, the means of presenting the widget for inspection, and the atmosphere in the relevant part of the factory. If we do not control each of these properly, the quality of the image reaching the camera will be degraded, possibly seriously. In fact, these "secondary" factors are all critical for success and we neglect them at our peril. We concentrate in this chapter on imaging optics. The other "primary" topics (i.e. illumination sources, lighting-viewing methods and image sensors) are covered elsewhere (➋ Chaps. 7 and ➋ 40).

5.1.2 Waves, Particles and Rays

Physicists conventionally represent light, in one of three ways:

(a) Waves
(b) Particles
(c) Rays

For almost all purposes in Machine Vision, the ray model is perfectly adequate. Occasionally, we must take into account the wave nature of light, so that we can understand interference effects caused by very small obstacles/apertures. (This is not necessary if we are only considering the colour of large objects, even though we frequently use the term wavelength.) The concept of light travelling in finite packets (photons) is hardly ever of significant interest when considering vision, whether it be in an animal, human being or machine. The reason is that so many photons are required for vision that we may conveniently measure light levels as analogue, rather than digital quantities. A light ray is merely a convenient mental model that we can use when designing normal (i.e. non-coherent) optical systems. We can also make several simplifying assumptions:

- In an homogenous medium (e.g. air, glass, water), light travels in a straight line.
- Light effectively travels with zero delay from one end of an optical system to the other. The system is small enough to allow us to ignore the fact that the speed of light is finite.
- Each ray can be characterised by two numbers:
  - Wavelength, which determines the sensation of colour that a person will perceive when that ray enters his/her eye.
  - Intensity, which determines how bright that ray will appear when it is viewed by a human being.

A ray has zero width. A "bundle" of rays travelling in the same general direction is called a *beam.*

This simple model is not unduly restrictive and is very convenient for out purposes. For example, it is possible to represent a very narrow beam with a complicated spectrum as a collection of rays superimposed upon one another and with different wavelengths.

There is one more parameter that a ray may have: *polarisation.* This does not alter its perception by either the eye, or a camera, but it does affect the way that a ray behaves when it

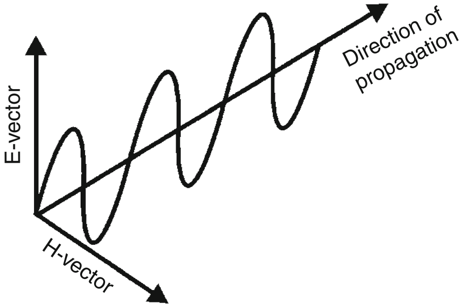


Fig. 5.2
The polarisation of light is described by referring to the E-vector (electric-field vector). The magnetic field vector (H-vector) is orthogonal to both the E-vector and the direction of propagation but is not normally considered when designing optical systems. The sinusoidal graph represents the variation of the electric field in space at a given moment. The magnetic field (not shown) also has a sinusoidal form and is 90° out of phase with the electric field. As time progresses, these two graphs move along the axis labelled "direction of propagation"

strikes a surface. Polarisation is determined by the orientation of the E-vector, which located in the plane orthogonal to the direction of travel (➋ *Fig. 5.2*). Light from an incandescent filament lamp, fluorescent tube or LED contains rays with no preference for the angle of polarisation. However, after specular reflection from a dielectric surface, or transmission through a polarising device, the emergent rays will almost all be polarised with their E-vectors orientated in the same direction. (Rays with their E-vectors in the orthogonal direction are attenuated, not rotated.)

There are six major physical effects that affect the behaviour of optical components:

(a) Absorption. (This leads to olive oil being coloured yellow, emeralds being green, etc.)
(b) Reflection (This property is exploited in making mirrors.)
(c) Scattering (This is caused when light strikes tiny particles suspended in a clear medium such as glass, water or air. Fog and milk are prime examples of light scattering media.)
(d) Diffraction (This effect is responsible for the colouration of beetles' wings and is exploited in optical gratings.)
(e) Refraction (This property is exploited in lenses and prisms.)
(f) Polarisation (Polarisation of light, either occurring naturally due to the nature of the object being viewed or deliberately introduced into the optical system, can often be used to good effect.)

5.1.3 Absorption, Transmittance, and Optical Density

5.1.3.1 Absorption

All materials absorb radiation in some parts of the electromagnetic spectrum. The amount of absorption depends on the wavelength, the amount of absorbing material in the radiation path, and the absorption of that material at that wavelength. Materials that absorb some visible wavelengths appear coloured. For purposes of this chapter, the term coloured glass also refers to material that is a wavelength-selective absorber in the near-ultraviolet to the near-infrared region.

Absorption occurs when the electric field of a light wave interacts with absorbing atoms or molecules that form an oscillating dipole. The photon is absorbed and the atom or molecule is placed in an excited state. This process occurs only at resonant wavelengths, which alters the spectrum of the light emitted from that material. In a solid or liquid absorber, excitation energy is dissipated as heat (vibrations of particles). Therefore, filters that rely mainly on absorption are not ideal for high-power laser applications. The intense local heating can lead to structural damage.

5.1.3.2 Transmittance

As a beam of light passes through an absorbing medium, the amount of light absorbed is proportional to the intensity of incident light times the absorption coefficient. Consequently, the intensity of an incident beam drops exponentially as it passes through the absorber. This is often expressed as *Beer's law*:

$$T_{\mathrm{i}} = 10^{-\alpha c x} \tag{5.1}$$

where T_{i} is internal transmittance, α is the absorption coefficient, c is the concentration of absorbers, and x is the overall thickness of the absorbing medium. Clearly α, and T_{i} are wavelength dependent. For solid absorbing mediums, $c = 1$.

Internal transmittance is the transmittance of an optical element when surface (coated or uncoated) losses are ignored. The measured transmittance of the element (including surface effects), transmittance, is called external transmittance, T.

Alternatively, a filter is defined by the amount of light it blocks, as opposed to the amount of light it transmits. This parameter is opacity, which is simply the reciprocal of the transmittance, $1/T$.

The transmittance of a series of filters is the product of their individual external transmittance, $T_1 \times T_2 \times T_3$, etc. Because transmittance (and hence opacity) is multiplicative, and since transmittance may extend over many orders of magnitude, it is often more convenient to use a logarithmic expression to define transmittance.

5.1.3.3 Optical Density

Optical Density (or *density*) is the base-10 logarithm of opacity:

$$D = \log(1/T). \tag{5.2}$$

As optical density increases, the amount of light blocked by the filter (by reflection and/or absorption) increases. The most important point to note is that optical density is additive. If several filters are stacked in series, their combined optical density is the sum of the individual optical densities.

Optical density is particularly useful for neutral-density (ND) filters. These filters, which have a very flat wavelength response, are used to attenuate light in a calibrated, chromatically invariant fashion. In this catalogue, ND filters are supplied in sets of various calibrated densities. Combinations of these filters can be used to produce many different calibrated optical densities.

5.1.4 Reflection of Light

5.1.4.1 Reflections at Uncoated Surfaces

Whenever light is incident on the boundary between two media, some light is reflected and some is transmitted (undergoing refraction) into the second medium. Several physical laws govern the direction, phase, and relative amplitude of the reflected light. For our purposes, it is necessary to consider only polished optical surfaces. Diffuse reflections from rough surfaces are not considered in this discussion. The law of reflection states that the angle of incidence (θ_1) equals the angle of reflection (θ_r). This is illustrated in ➲ *Fig. 5.3*, which shows reflection of a light ray at a simple air/glass interface. The incident and reflected rays make an equal angle with respect to the axis perpendicular to the interface between the two media.

5.1.4.2 Intensity

At a simple interface between two dielectric materials, the amplitude of reflected light is a function of the ratio of the refractive index of the two materials, the polarization of the incident

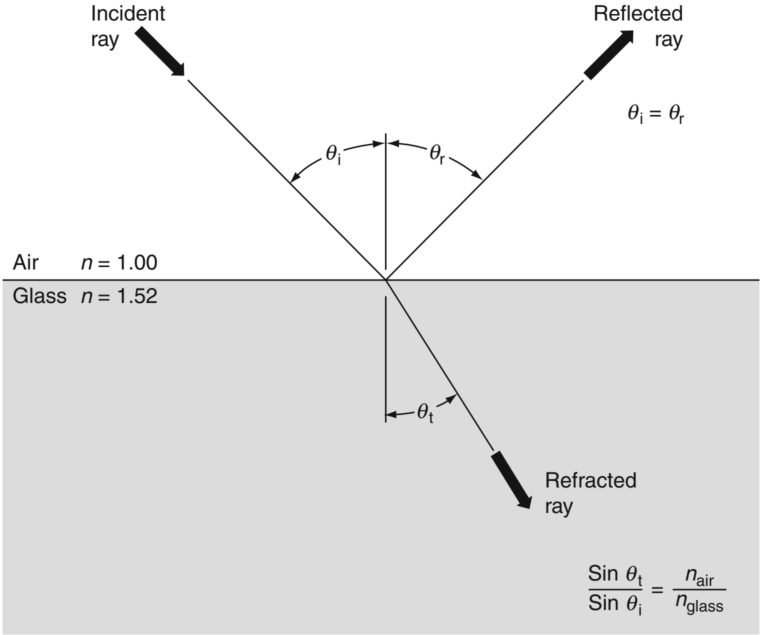


Fig. 5.3
Reflection and refraction at a simple air-glass interface

light, and the angle of incidence. When a beam of light is incident on a plane surface at normal incidence, the relative amplitude of the reflected light, as a proportion of the incident light, is given by

$$\frac{(1-p)}{(1+p)} \tag{5.3}$$

where p is the ratio of the refractive indexes of the two materials ($n1/n2$). Intensity is the square of this expression. The greater the disparity between the two refractive indexes, the greater the reflection. For an air/glass interface, with glass having a refractive index of 1.5, the intensity of the reflected light will be 4% of the incident light. For an optical system containing ten such surfaces, the transmitted beam will be attenuated to approximately 66% of the incident beam due to reflection losses alone, emphasizing the importance of antireflection coatings to system performance.

5.1.4.3 Incidence Angle

The intensity of reflected and transmitted beams at a surface is also a function of the angle of incidence. Because of refraction effects, it is necessary to differentiate between external reflections, where the incident beam originates in the medium with a lower refractive index (e.g., air in the case of an air/glass or air/water interface), and external reflection, where the beam originates in the medium with a higher refractive index (e.g., glass in the case of a glass/air interface, or flint glass in the case of a flint/crown-glass interface), and to consider them separately.

5.1.4.4 External Reflection at a Dielectric Boundary

Fresnel's laws of reflection precisely describe amplitude and phase relationships between reflected and incident light at a boundary between two dielectric media. It is convenient to think of the incident radiation as the superposition of two plane-polarized beams, one with its electric field parallel to the plane of incidence (p-polarized), and the other with its electric field perpendicular to the plane of incidence (s-polarized). Fresnel's laws can be summarized in the following two equations, which give the reflectance of the s- and p-polarized components:

$$r_{\mathrm{s}} = \left[\frac{\sin(\theta_1-\theta_2)}{\sin(\theta_1+\theta_2)}\right]^2 \tag{5.4}$$

$$r_{\mathrm{p}} = \left[\frac{\tan(\theta_1-\theta_2)}{\tan(\theta_1+\theta_2)}\right]^2. \tag{5.5}$$

In the limit of normal incidence in air, Fresnel's laws reduce to the following simple equation:

$$r = \left(\frac{n-1}{n+1}\right)^2. \tag{5.6}$$

It can easily be seen that, for a refractive index of 1.52 (crown glass), this gives a reflectance of 4%. This important result reaffirms that, in general, 4% of all illumination incident normal

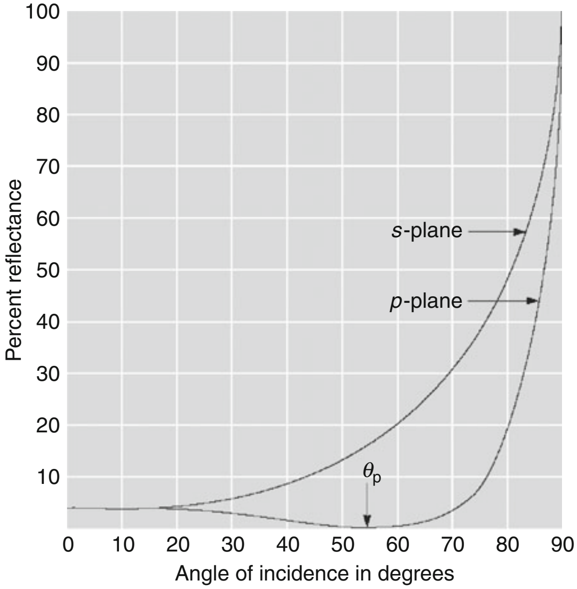


Fig. 5.4
External reflection at a glass surface (n = 1.52) showing *s*- and *p*-polarized examples

to an air-glass surface will be reflected. The variation of reflectance with angle of incidence for both the *s*- and *p*-polarized components, plotted using the formulas above, is shown in *Fig. 5.4*.

It can be seen that the reflectance remains close to 4% over about 25° incidence, and that it rises rapidly to nearly 100% at grazing incidence. In addition, note that the *p*-component vanishes at 56° 39′. This angle, called *Brewster's angle*, is the angle at which the reflected light is completely polarized. This situation occurs when the reflected and refracted rays are perpendicular to each other ($\theta_1 = \theta_2 = 90°$), as shown in *Fig. 5.5*.

This leads to the expression for Brewster's angle, θ_B:

$$\theta_1 = \theta_B = \arctan(n_2/n_1). \tag{5.7}$$

Under these conditions, electric dipole oscillations of the *p*-component will be along the direction of propagation and therefore cannot contribute to the reflected ray. At Brewster's angle, reflectance of the *s*-component is about 15%.

5.1.4.5 Internal Reflections at a Dielectric Boundary

For light incident from a higher to a lower refractive index medium, we can apply the results of Fresnel's laws in exactly the same way. The angle in the high-index material at which polarization occurs is smaller by the ratio of the refractive indices in accordance with Snell's law. The internal polarizing angle is 33° 21′ for a refractive index of 1.52, corresponding to the Brewster angle (56° 39′) in the external medium, as shown in *Fig. 5.6*. The angle at which the emerging

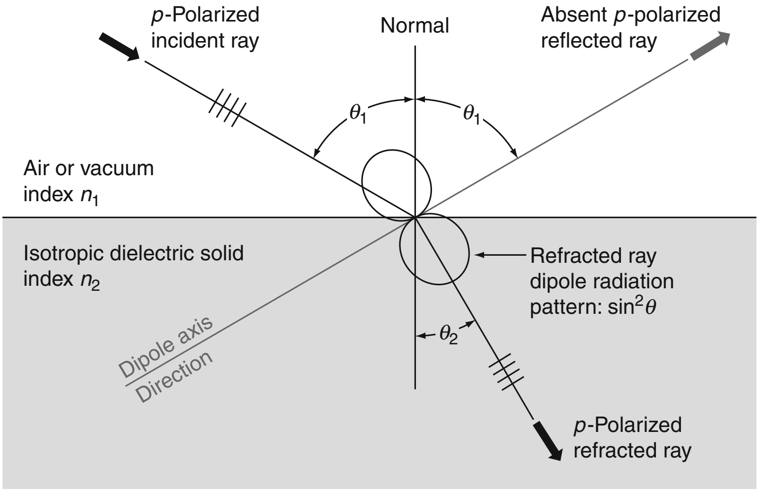


Fig. 5.5
Brewster's angle. The *p*-polarized component is completely absent in the reflected ray

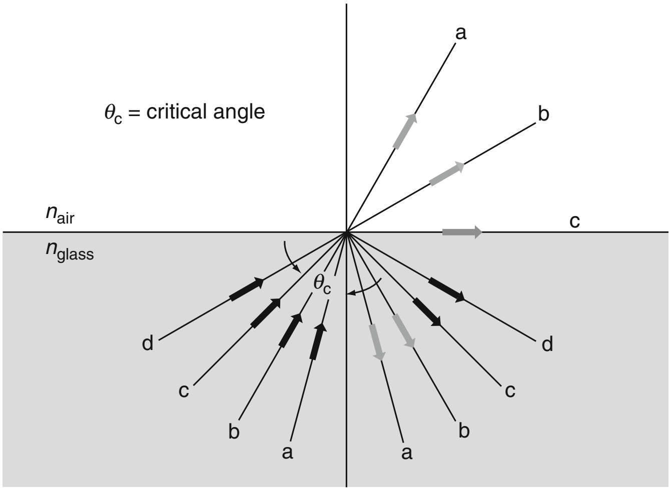


Fig. 5.6
Internal reflection at a glass-air interface (n = 1.52) showing *s*- and *p*-polarized components

refracted ray is at grazing incidence is called the critical angle (see Fig. 5.7). For an external medium of air or vacuum ($n = 1$), the critical angle is given by

$$\theta_c(\lambda) = \arcsin\left(\frac{1}{n(\lambda)}\right) \tag{5.8}$$

and depends on the refractive index *n*1, which is a function of wavelength. For all angles of incidence higher than the critical angle, total internal reflection occurs.

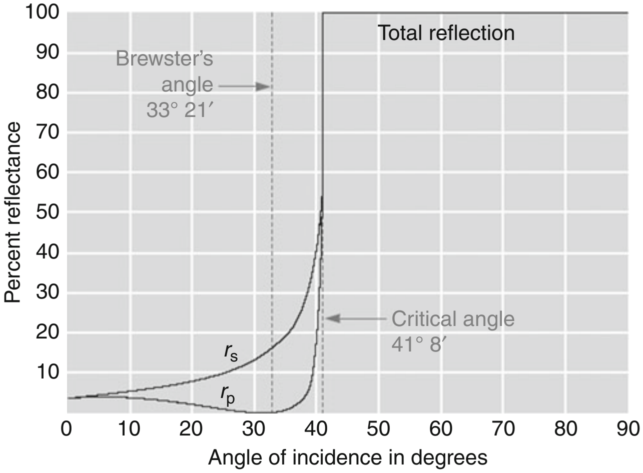


Fig. 5.7
Critical angle. The emerging ray is at grazing incidence

5.1.4.6 Phase Changes on Reflection

There is another, more subtle difference between internal and external reflections. During external reflection, light waves undergo a 180° phase shift. No such phase shift occurs for internal reflection (except in total internal reflection). This is one of the important principles on which multilayer films operate.

5.1.5 Diffraction

Electromagnetic waves are described by electric and magnetic field components. They travel in the form of a wave. A wavelength can be associated with these fields. A relatively small part of the electromagnetic *spectrum* is physiologically detected as colour by our eye. We commonly describe this part of the spectrum as *light*, or visible light and denoted by *VIS*. The spectrum spans shorter wavelengths: ultraviolet (*UV*), X-rays and Gamma rays. Longer wavelengths span the infrared (*IR*), microwaves, radio and television. Electric and magnetic field components can be physically measured. The height (amplitude) of the wave varies with time. If we were to observe the wave-like propagation of the electric field, $\mathbf{E}(x,t)$, we would find it can be described mathematically as:

$$E(x,t) = E_0 \sin(\omega t - kx) \tag{5.9}$$

where

E_0 Maximum amplitude of the electric field (Vm^{-1})
ω $2\pi/T$ where T is the period of oscillation (s).
$1/T$ is referred to as the frequency υ (Hz). (Frequency of light: $4 \times 10^{14} < \upsilon < 7.9 \times 10^{14}$ Hz)
k $2\pi/\lambda$ where λ is the wavelength (m). ($0.380\ \mu\mathrm{m} < \lambda < 750\ \mu\mathrm{m}$). $1/\lambda$ is the wave number.
x position coordinate
t time

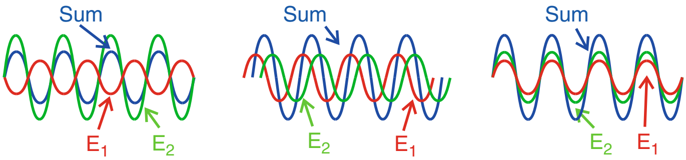


Fig. 5.8

Two propagating sinusoidal wave. (a) The sum is always a sinusoid. (b) Constructive interference. (c) Destructive interference

The frequency of light and the corresponding wavelength are related by:

$$\lambda = c/\upsilon \tag{5.10}$$

where c is the speed of light in a vacuum: $2.9979 \times 10\ \mathrm{m}^{-1}$.

The expression in parenthesis on the right hand side of Eq. 5.9 should be noted. The first term, ωt, is the time component and the second, kx, the spatial component of propagation. The following example shows the effect these components have on wave propagation. *Figure 5.8* shows two independent waves propagating. At $t = 0$, the one has an electric field strength, $E_1 = 0$, while the other E_2, has a non-zero amplitude. At time $t = 0$, the argument of the sine function in (1) becomes kx. We therefore have $E1(x{,}0) = 0$. This means the product, kx, must equal $N\pi$, (where $N = 0{,}+1{,}+2{,}\ldots$). E_2 has a negative amplitude at $t = 0$, suggesting that kx is not a multiple of π. In this case we refer to the E_2 wave as being "phase shifted" from E_1. The condition of this shift in position between the two waves will not change as the waves propagate further in space, so long as the waves are undisturbed. The contribution of kx describes the spatial phase of a wave at position x.

5.1.6 Interference

If two or more waves meet at a point x, the contribution and position of the field components, E_1 and E_2, can be superimposed on each other. Superposition (*Positive* or *Constructive Interference*) does not take place if the field amplitudes are oriented perpendicular to each other. *Figure 5.8a* shows the superposition of two sine waves. Their sum is a sinusoid, in the general case when the phases differ from each other. Comparing the resultant field amplitude, E, with the individual components, E_1 and E_2, we observe in *Fig. 5.8b* an increase in amplitude. This is constructive interference: the phase difference between E_1 and E_2 is zero or multiples of 2π. ($\Delta\varphi = N \times 2\pi$, $N = 0{,}\pm1$, ±2, ...) Destructive interference is shown in *Fig. 5.8c* where there is a decrease in the resultant field amplitude. The phase difference between the two waves is half of one wavelength ($\Delta\varphi = (2\ N + 1) \times \pi$). If the constituent amplitudes would have the same amplitude and differ in phase by half a wavelength ($\Delta\varphi = (2\ N + 1) \times p$), the resultant amplitude would be zero. Superposition of a constructive nature leads to an increase in light intensity (high intensity), whereas destructive interference leads to a decrease in intensity, or even zero intensity. This effect can be seen in the case to two beams propagating in different directions, which simultaneously impinge on a screen at the same coordinate. One can observe the periodic pattern of light and dark stripes (double slit interference). This pattern will appear everywhere the two beams superimpose. The ability of light to interfere, along with the observations of Christian von Huygens, that wavefronts themselves can become wave centres of point source propagation explains the effect of diffraction.

5.1.6.1 Diffraction

A plane wave impinging on a geometric structure with an opening (wall with a round opening aperture) will emit elementary point source waves from each opening. The projected pattern of these waves oscillates between bright and dark areas, which are not sharply separated. This oscillatory transition is the result of interference of these elementary waves. The phase difference between neighbouring maxima or minima is always 2π.

Diffraction therefore means that wave fronts travelling in a straight line, which are then restricted by some opening, divert off their straight path into light/dark patterns. This assumes, of course, that the waves don't experience other phenomena such as reflection or refraction. In *Fig. 5.9*, one can see deviation of the diffracted light from the original angle impinging the wall. The optical path difference, ($N\lambda$), between rays emanating from the edges of the opening and the size of the geometrical opening, D, will determine the Nth order diffraction angle.

Each $N = \pm1, \pm2$, defines a direction α, or for small angles, $N\alpha$, of an intensity maximum. The intensity pattern at the edges due to contributions from non-diffracted rays leads to a reduced sharpness of bright/dark transition. We refer to this as *reduced contrast.*

The *direction* of rays and therefore of diffracted rays is always perpendicular to their wave surface. Geometrical optics neglects diffraction in progressive ray calculations. One assumes = 0 to be a boundary condition of wave optics. All cross rays in the system have a diffraction angle of zero ($\lambda = 0 = > \alpha = 0$).

5.1.7 Scattering and Diffusion

In a perfect vacuum, a ray of light travels, without loss, in a straight line. In a defect-free, homogenous, transparent material, such as optical-quality glass, there is a little loss, due to a combination of absorption and scattering. In glass of lower quality, such that used for making bottles and windows, there is likely to be a greater level of scattering but still at a tolerable level for non-critical optical functions. In media such as smoke, fine water/oil mist, aerosols, milk and mayonnaise, scattering is the dominant optical property.

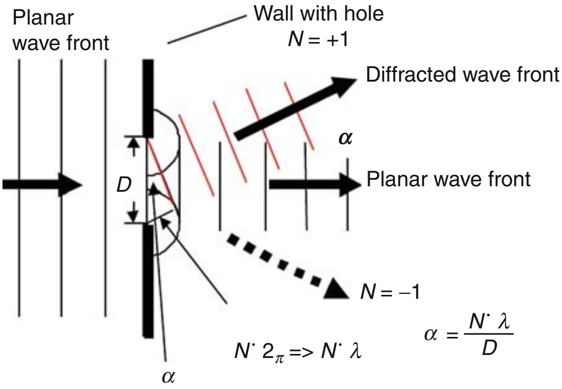


Fig. 5.9
Diffraction through a small aperture of width D

Scattering can occur either as a surface effect, or one that involves the bulk material and can be caused in various ways:

- *Refractive index changes in a transparent material.* Light deviates from a straight-line trajectory when it passes through localized non-uniformities in the refractive index of a transmissive medium. This type of scattering is caused by aerosols, as well as tiny air bubbles in water and glass. These behave like numerous tiny lenses, inevitably causing parallel beams to diverge.
- *Tiny opaque particles suspended in a transparent medium.* Airborne dust and smoke particles scatter light by diffraction (➲ *Fig. 5.10*). Inclusions and crystal-structure defects in gems also cause scattering.
- *Tiny opaque particles adhering to a reflective surface.* Particles deposited from fumes and smoke scatter light by diffraction.
- *Contamination of a surface by liquid droplets.* These may be deposited from steam, water/oil sprays, fingerprints, spittle, etc.
- *Rough surfaces on a transparent material.* Ground glass provides a good example of this type of scattering. Scratches and pits on glass surfaces form sites for localised scattering.
- *Rough surfaces on an opaque material.* Textiles, cut unpolished wood, unglazed china are examples. Cells in animal/plant tissue also cause scattering.

Due to their complex structural and chemical nature, biological materials scatter light by a variety of means. Indeed, they do it so well, that it can be difficult to obtain good contrast images, as it is often impossible to control the diffusion of light from one part of a sample to another. As evidence of this, recall that the ends and edges of the fingers glow when they are held in front of a bright light.

5.1.7.1 Physical Origins of Scattering

To understand diffuse reflection, consider a rough surface as consisting of a series of very small facets, with random orientations. Each of these provides a mirror-like (specular) reflection.

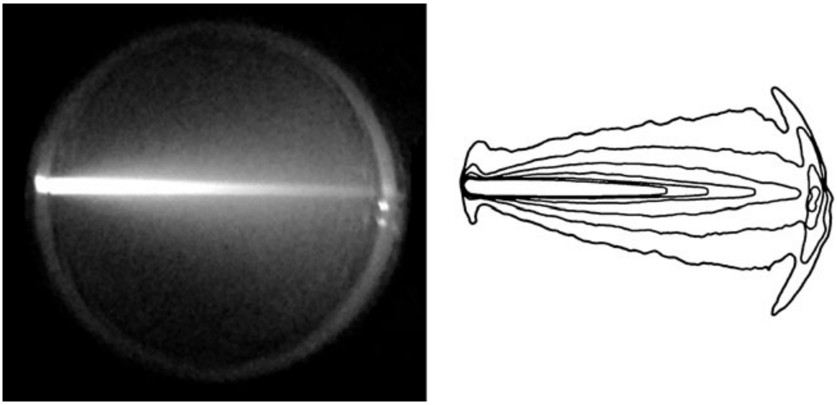


Fig. 5.10

Scattering of light in milk diluted with water. (a) Laser beam enters the glass container from the left. (b) Intensity contours (isophotes) emphasise the conical spreading of the beam

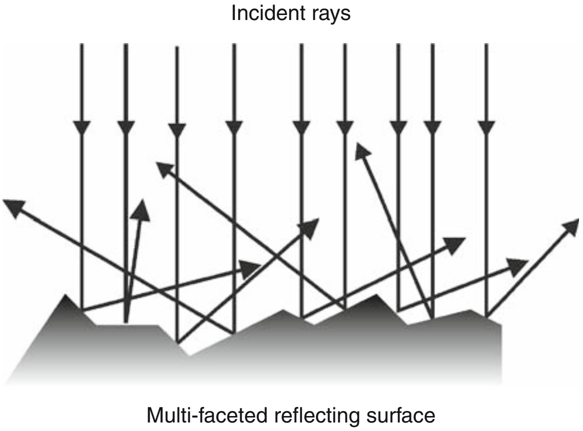


Fig. 5.11

Diffuse reflection. Each ray may be regarded as undergoing specular reflection from a tiny mirror with random orientation

The macroscopic effect is to see light emerging from the surface in all directions (➋ *Fig. 5.11*). A surface exhibits *Lambertian reflectance* if light falling on it is scattered such that its apparent brightness is the same regardless of the observer's angle of view. That is, the luminance is isotropic. The reflection from a cross-grain wood surface is approximately Lambertian, as it is from the surface of unglazed bone china and a clean button mushroom and an egg. The Lambertian reflection model is often used as the basis for generating realistic images in Computer Graphics. Of course, if a matt surface has a glossy coating (e.g. polyurethane or lacquer), a mixture of effects combine to provide the total effect that we see (➋ *Fig. 5.12*). (This diagram does not take into account the partial polarisation of light reflected from a dielectric surface.

There are two principal types of scattering that have been subjected to detailed mathematical analysis:

- *Rayleigh scattering* occurs when light is deflected by very small objects, usually taken to be of diameter less than λ/10, where λ is the wavelength. For objects of this size or less, the exact shape is not significant; mathematical analysis is reasonably accurate if we replace complex (real) particles with spheres of equivalent volume. Rayleigh scattering is responsible for the sky appearing to be blue at midday and reddening at sun-rise and sunset. It also occurs in fog, industrial mists, smoke, aerosols and perhaps most importantly low-grade glass.
- For larger particles, *Mie scattering* occurs. In this case, the shape of the scattering objects is more important; no general closed-form mathematical analysis has been discovered for particles of arbitrary shape.

The interaction of light with particles of diameter greater than about 10λ can usually be described adequately using geometric (i.e. ray) optics. This is not usually described as scattering. In order to model scattering from large irregularly shaped particles, computational methods are used.

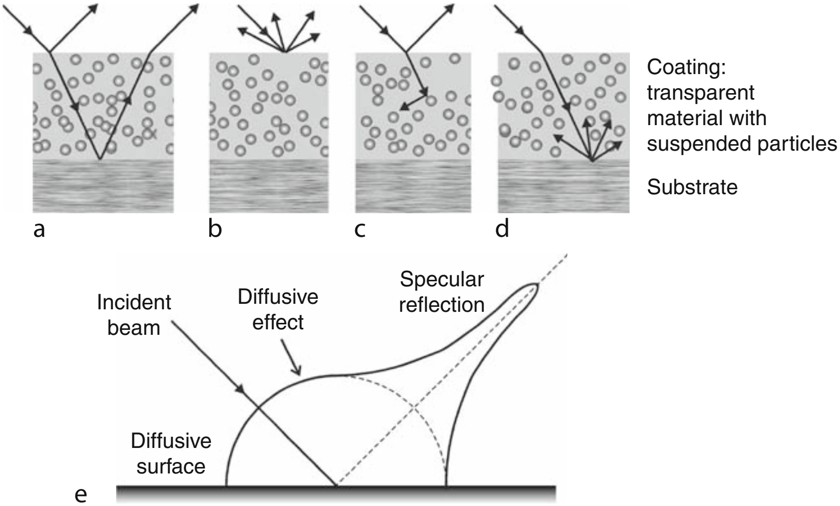


Fig. 5.12

Diffusion and reflection from a painted surface. The paint is a thin coating containing minute suspended particles. In practice, the reflection and scattering effects shown here all occur to some extent. (a) Some light rays penetrate the coating and undergo specular reflection from the interface between it and the substrate. Some rays are reflected from the outer surface of the coating. (b) Some rays are scattered by the external surface of the coating. (c) Some light rays are scattered by particles suspended in the coating. Some of these scattered rays re-emerge from the coating (not shown). (d) Scattering occurs at the interface. (e) Polar energy distribution. Again, some of these scattered rays re-emerge from the coating (not shown). The polar energy plot for a perfect Lambertian reflector is an hemisphere

Light that has already been scattered by one diffusing reflection may fall onto another, then onto another and so on. This gives rise to *multiple scattering*. *Single scattering*, as described above, can be treated as a random phenomenon and can conveniently be represented in terms of probability distributions. On the other hand, multiple scattering is more deterministic; it can be analysed on a macroscopic level using intensity distributions (➲ *Fig. 5.13*). Multiple scattering is analogous to the diffusion of heat in a solid and is treated mathematically in the same way. *Diffusers* are optical elements designed to produce multiple scattering and are useful in improving the homogeneity of lighting for Machine Vision. They may be either transmitting or reflecting devices.

5.2 Optical Materials

5.2.1 Criteria for Optical Materials

There are several important criteria for optical materials:

- The *index of refraction* (or *refractive index*) is the ratio of the speed of light in that material to the speed of light in a vacuum. By definition, the refractive index for a vacuum is 1.00. The refractive index of all materials varies to some extent with wavelength.

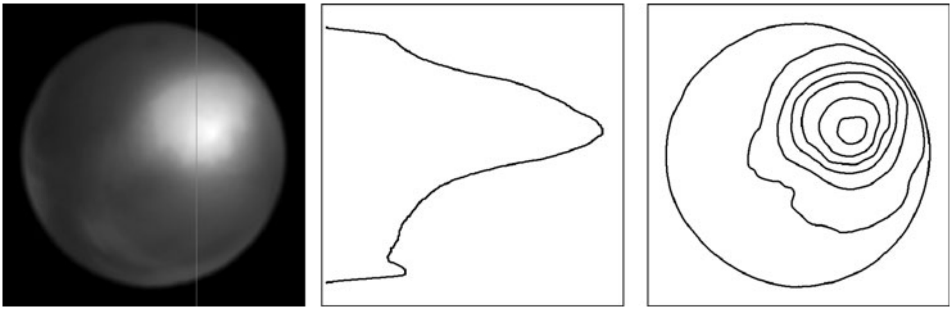


Fig. 5.13
Diffuse and specular reflection from a sphere with a small ("point") light source. (a) Original image, computer simulation. (b) Intensity profile along the vertical line in (a). (c) Intensity contours (isophotes)

- The *Abbé number* of a material quantifies the amount of dispersion (i.e., variation in refractive index) over a specific spectral range. A high Abbé number generally gives less colour dispersion and reduces colour aberration.
- The density of a glass is a factor determining the weight of an optical component. It is unlikely to be critical for many industrial applications.
- The hardness of an optical material indicates its resistance to scratching. Surface damage can seriously degrade the performance of an optical component.
- The coefficient of thermal expansion is important for applications involving rapid changes of temperature and large spatial temperature differentials.

A list of refractive index values for some common optical and other transparent materials is given in *Table 5.1*, while *Fig. 5.14* shows the variation in refractive index for a range of optical materials. Typical values for the reflectance from opaque materials are provided in *Table 5.2*.

5.2.2 Important Characteristics

Glass manufacturers provide hundreds of different glass types with differing optical transmissibility and mechanical strengths. The most important material properties to consider in regard to an optical element are as follows:

- *Spectral transmission characteristics* A material must be transparent at the wavelength of interest, if it is to be used for a transmissive component. A transmission curve allows the optical designer to estimate the attenuation of light, at various wavelengths, caused by internal material properties. For mirror substrates, the attenuation is not important.
- *Refractive index* The index of refraction, as well as the rate of change of index with wavelength (dispersion), might require consideration. High-index materials allow the designer to achieve a given power with less surface curvature, typically resulting in lower aberrations. On the other hand, most high-index flint glasses have higher dispersions, resulting in more chromatic aberration in polychromatic applications. They also have poorer chemical characteristics than lower-index crown glasses.

Table 5.1
Refractive index values

Material	Refractive index
Acetone	1.36
Acryilic (Polymethyl methacrylate, PMMA)	1.490–1.492
Air	1.00029
Alcohol, ethyl	1.36
Alcohol, methyl	1.329
Amethyst	1.532–1.554
Benzene	1.501
Beryl	1.57–1.60
Calcite	1.486–1.658
Crown glass (impure)	1.485–1.755
Crown glass (pure)	1.50–1.54
Diamond ($\lambda = 0.6563$ μm)	2.417
Emerald	1.560–1.605
Flourite, CaF_2	1.434
Glass, arsenic trisulfide	2.04
Glass, crown (spectacles)	1.523
Glass, flint, 29% lead	1.569
Glass, flint, 55% lead	1.669
Glass, flint, 71% lead	1.805
Glass, fused silica	1.459
Glass, lanthanum flint	1.82–1.98
Glass, pyrex (borosilicate)	1.474
Glycerol	1.4729
Lithium niobate	2.283
Magnesium fluoride (MgF_2)	1.3777
Mineral oil, $\lambda = 0.4678$ μm	1.47685
Mineral oil, $\lambda = 0.4800$ μm	1.47583
Mineral oil, $\lambda = 0.5086$ μm	1.47373
Mineral oil, $\lambda = 0.5461$ μm	1.47149
Mineral oil, $\lambda = 0.5893$ μm	1.46945
Mineral oil, $\lambda = 0.6438$ μm	1.46744
Mylar (boPET)	1.65
Vegetable oil (50°C)	1.47
Poly(ethylene terephthalate) (PET)	1.5750
Plastic	1.46–1.70
Polycarbonate	1.584–1.586
Polystyrene	1.55–1.59
Quartz	1.543–1.545

Table 5.1 (Continued)

Material	Refractive index
Ruby	1.76–1.77
Sodium chloride (NaCl, salt)	1.544
Sapphire	1.76–1.77
Sugar solution, 30%	1.38
Sugar solution, 80%	1.49
Turpentine	1.472
Vacuum (by definition)	1.00
Water ice	1.309
Water, 0°C	1.33346
Water, 20°C	1.33283
Water, 100°C	1.31766
Poly(tetrafluoroethylene) (PTFE, teflon)	1.35–1.38

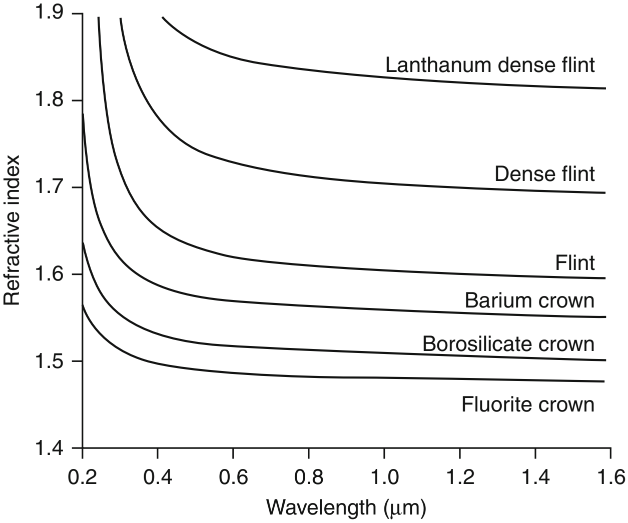


Fig. 5.14

Variation of refractive index versus wavelength, for a various optical glasses

- *Thermal expansion* The thermal expansion coefficient can be particularly important in applications in which the part is subjected to high temperatures, such as high-intensity projection systems. This is also of concern when components must undergo large temperature cycles, such as in optical systems used outdoors.
- *Mechanical characteristics* The mechanical characteristics of a material are significant in many areas. They can affect how easy it is to fabricate the material into shape, which affects

Table 5.2

Reflectance values for common materials (Hill DA (1977) Fibre optics. Business Books, London)

Material	Light reflected (%)
Silver-bright	94
Polished aluminium	90
Fresh whitewash	90
Ceiling paint	80
Porcelain	75
Drawing paper	70
Bright chrome	70
White linen	75
Nickel	60
Matt steel-bright	55
Matt brass	50
Black paper	5
Black velvet	1

product cost. Scratch resistance is important if the component will require frequent cleaning. Shock and vibration resistance are important for military, aerospace, or certain industrial applications. The ability to withstand large pressure differentials is important for windows used in vacuum or pressure chambers. An important mechanical property of glass is microhardness. A precisely specified diamond scribe is placed on the glass surface under a known force. The indentation is then measured. The Knoop and the Vickers microhardness tests are used to measure the hardness of a polished surface and a freshly fractured surface, respectively.

- *Chemical Properties* The chemical properties of materials are important to manufacturers of optical equipment and users, especially when the component will be used in a harsh environment. To quantify the chemical properties of glasses, manufacturers rate each glass according to four categories: climatic resistance, stain resistance, acid resistance, and alkali and phosphate resistance.
- *Climatic Resistance* Humidity can cause a cloudy film to appear on the surface of some optical glass. Climatic resistance expresses the susceptibility of a glass to high humidity and high temperatures. In this test, glass is placed in a water-vapour-saturated environment and subjected to a temperature cycle which alternately causes condensation and evaporation. The glass is given a rating from 1 to 4 depending on the amount of surface scattering induced by the test. A rating of 1 indicates little or no change after 30 days of climatic change; a rating of 4 means a significant change occurred in less than 30 hours.
- *Stain Resistance* Stain resistance expresses resistance to mildly acidic water solutions, such as fingerprints or perspiration. In this test, a few drops of a mild acid are placed on the glass. A coloured stain, caused by interference, will appear if the glass starts to decompose. A rating from 0 to 5 is given to each glass, depending on how much time elapses before stains occur. A rating of 0 indicates no observed stain in 100 hours of exposure; a rating of 5 means that staining occurred in less than 0.2 hours.

- *Acid Resistance* Acid resistance quantifies the resistance of a glass to stronger acidic solutions. Acid resistance can be particularly important to lens manufacturers because acidic solutions are typically used to strip coatings from glass or to separate cemented elements. A rating from 1 to 4 indicates progressively less resistance to a pH 0.3 acid solution, and values from 51 to 53 are used for glass with too little resistance to be tested with such a strong solution.
- *Alkali and Phosphate Resistance* Alkali resistance is also important to the lens manufacturer since the polishing process usually takes place in an alkaline solution. Phosphate resistance is becoming more significant as users move away from cleaning methods that involve chlorofluorocarbons (CFCs) to those that may be based on traditional phosphate-containing detergents. In each case, a two-digit number is used to designate alkali or phosphate resistance. The first number, from 1 to 4, indicates the length of time that elapses before any surface change occurs in the glass, and the second digit reveals the extent of the change.

5.2.3 Material Types

Crown glass is a low-index commercial-grade glass. The index of refraction, transmittance, and homogeneity are not controlled as carefully as they are in some other optical-grade glasses. Optical crown glass is suitable for applications in which component tolerances are fairly loose and is a substrate material for mirrors. Transmittance characteristics are shown in ➲ *Fig. 5.15*.

Fused silica (SiO_2) is an ideal material for many optical applications. It is transparent over a wide spectral range (➲ *Fig. 5.16*). It has a low coefficient of thermal expansion, and is resistant to scratching and thermal shock. Synthetic fused silica (amorphous SiO_2) should not be confused with fused quartz, which is made by crushing and melting natural crystals, or by fusing silica sand. Both of these processes result in a granular microstructure with trapped impurities and bubbles. These lead to local variations of refractive index and reduced

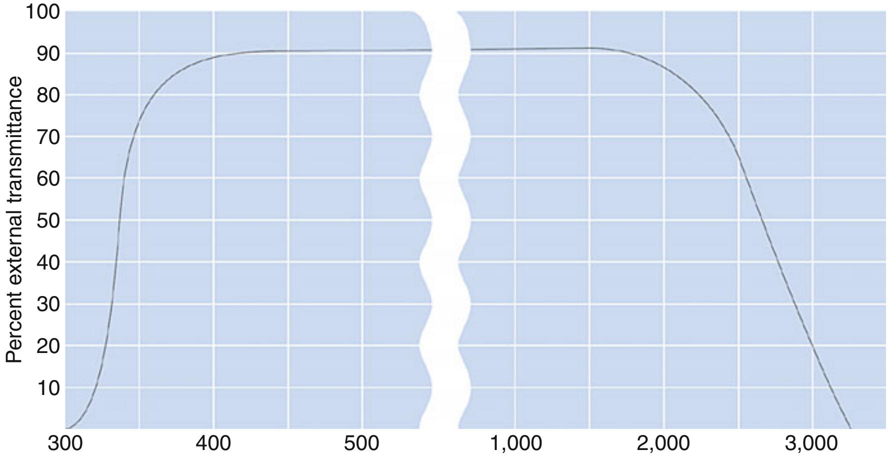


Fig. 5.15
Transmittance of crown glass versus wavelength (nm)

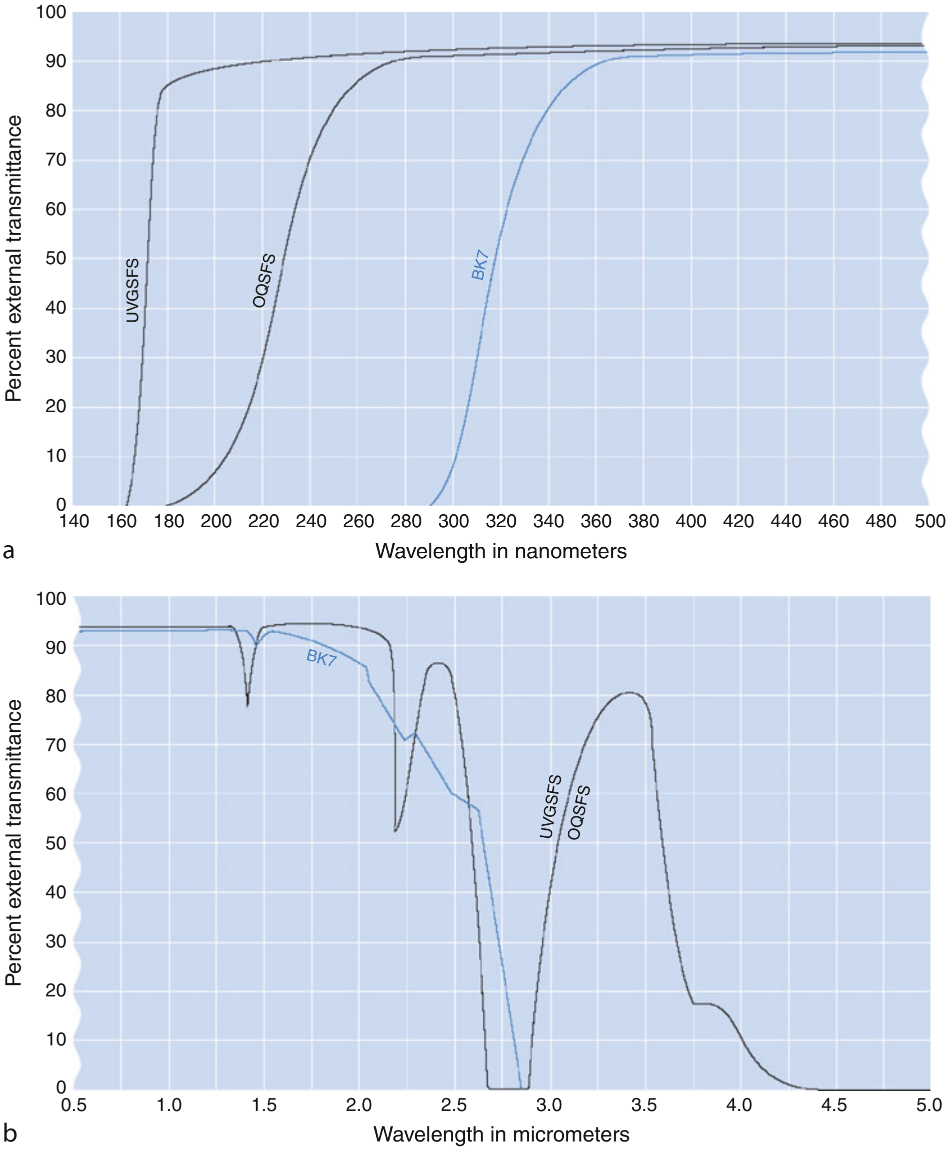


Fig. 5.16

Transmittance of optical quality synthetic fused silica (OQSFS) compared to ultra-violet grade synthetic fused silica. (a) UV waveband. (b) Visible and IR wavebands

transmission throughout the spectrum. Synthetic fused silica is far purer than fused quartz and provides a number of advantages:

- Greater ultraviolet and infrared transmission
- Low coefficient of thermal expansion, which provides stability and resistance to thermal shock
- Wider thermal operating range

- Increased hardness and resistance to scratching
- Much higher resistance to radiation darkening from ultraviolet, x-rays, gamma rays, and neutrons
- Reduced chromatic aberration
- Very low fluorescence levels (approximately 0.1% that of fused natural quartz excited at 0.254 μm. UV-grade synthetic fused silica does not fluoresce in response to wavelengths longer than 0.290 μm)

Glass transmittances are affected by thermal history after manufacture, as well as during the manufacturing process. Depending on the manufacturer and subsequent thermal processing (coating, annealing, or tempering), it is possible for the internal transmittance of any optical glass to be reduced by several percent across the entire spectrum. External transmittance is correspondingly affected. Transmittance of all glass is especially uncertain at wavelengths approaching the water absorption band at 2.7 μm. Synthetic fused silica also exhibits batch-to-batch transmittance variations, especially in the deep ultraviolet and infrared spectral regions. Visible batch-to-batch transmittance variations in synthetic fused silica are insignificant.

Sapphire (Al_2O_3) is an almost ideal material for optical windows, since it is very strong and extremely hard. It can be scratched by only a few substances: diamond or boron nitride and itself. As a result, it can be cleaned with impunity. It is chemically inert and insoluble in almost everything, except at highly elevated temperatures. Even hydrogen fluoride fails to attack sapphire at temperatures below 300°. Sapphire exhibits high internal transmittance from 0.150 to 6 μm. The external transmittance of sapphire is shown in ➔ *Fig. 5.17*. It has an exceptionally high thermal conductivity, so thin windows can be cooled by a forced air jet. Conversely, sapphire windows can be heated to prevent condensation. Sapphire exhibits birefringence. Most transmission losses of sapphire are due to surface reflections. Magnesium fluoride is an

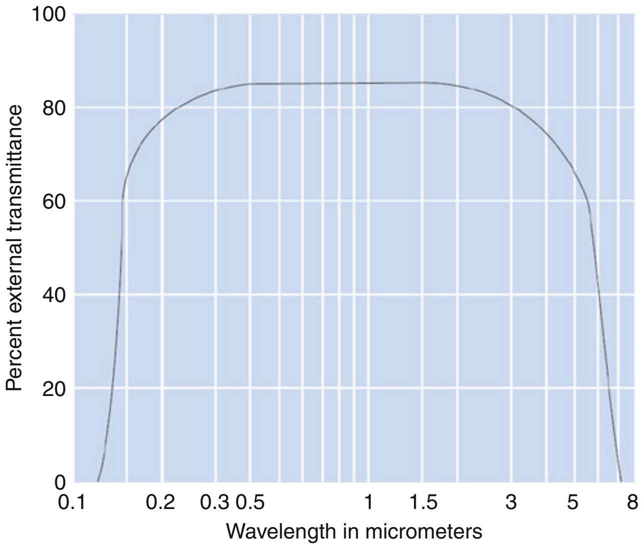


Fig. 5.17
Transmittance for sapphire

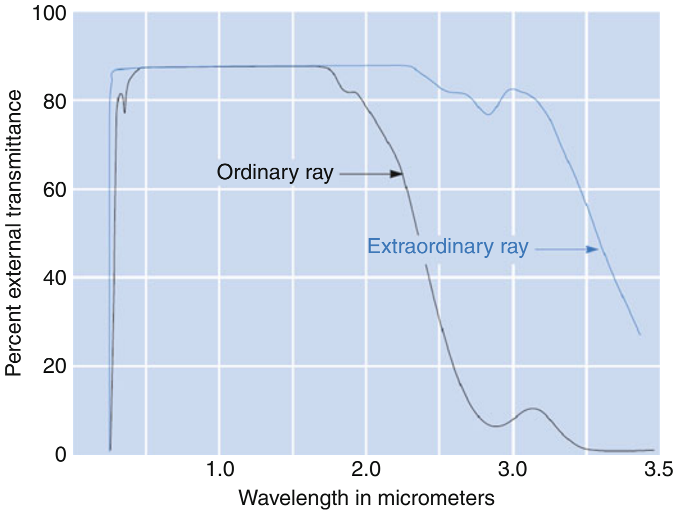


Fig. 5.18
Transmittance of calcite

almost ideal anti-reflection coating; a single layer of MgF_2 can ensure that the total transmission losses through a sapphire window are below 2%, over the entire visible spectrum.

Calcite ($CaCO_3$) exhibits pronounced birefringence. Strong birefringence and a wide transmission range make mineral popular for making polarising prisms. Most optical calcite is mined, rather than synthesised. It is a soft crystal and tends to cleave easily. These factors all contribute to the high cost of calcite prisms, compared to other types of polarisers. *Figure 5.18* shows the external transmittance of calcite. Since calcite is a natural crystal, the transmission is likely to vary from one sample to another.

Calcium fluoride (CaF_2) transmits readily over a wide spectrum (*Fig. 5.19*). As a result, it has widespread applications in the ultraviolet and infrared regions of the spectrum. It can be manufactured into windows, lenses, prisms, and mirror substrates. CaF_2 occurs naturally and can be mined. This yields a product with some impurities. A more homogenous material can be synthesised using an expensive manufacturing process.

α-Barium Borate (α-$Ba_2B_2O_4$) exhibits pronounced birefringence, is harder than calcite, has a high damage threshold, has a low coefficient of thermal expansion and is transparent over the visible and IR parts of the spectrum (*Fig. 5.20*). These properties make it an attractive alternative to calcite for polarising prisms.

ZERODUR® (Schott-AG) Many optical applications require a substrate material with a very low coefficient of thermal expansion and/or excellent thermal shock resistance. ZERODUR® is a glass ceramic material that is claimed to have the following characteristics:

- Very low coefficient of thermal expansion
- Good 3D homogeneity
- High internal quality
- Good machining properties
- Can be polished to a high accuracy
- Can be coated easily

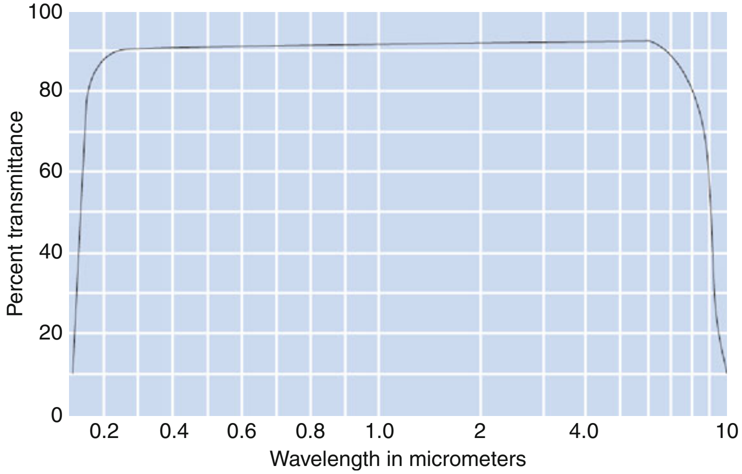


Fig. 5.19

Transmittance of calcium fluoride

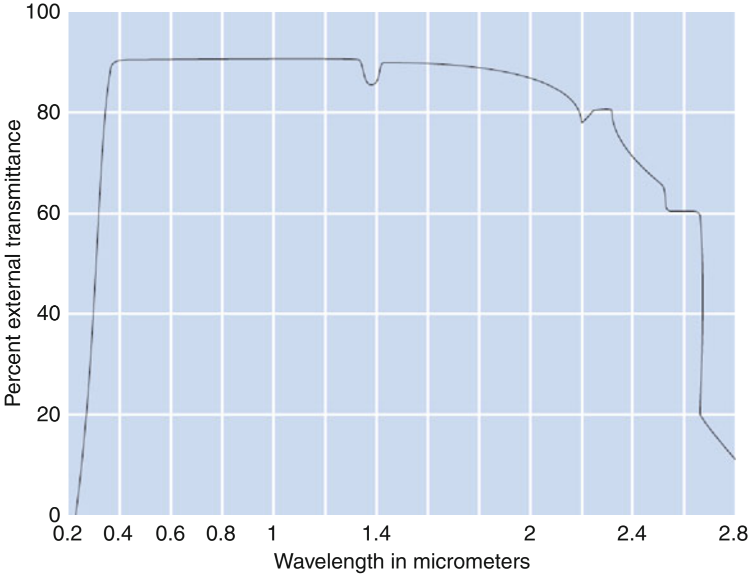


Fig. 5.20

Transmittance of α-barium borate

- Non-porous
- Good chemical stability

ZERODUR® is a glass-ceramic composite class of materials that has both amorphous (vitreous) and crystalline components. At room temperatures, the coefficients of thermal expansion of the vitreous phase (positive) coefficient and crystalline phase (negative) nearly cancel, resulting in a low overall coefficient of thermal expansion; within the temperature range 20–300°C this is $0.05 \pm 0.10 \times 10^{-6}$/°C. As a result, this type of material is well suited as a mirror substrate for high-precision applications, such as interferometry, holography and reflectors for large telescopes.

5.2.4 Light Absorbing Materials and Structures

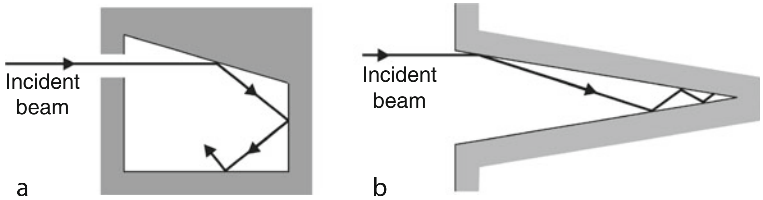
When a 50:50 beamsplitter is used in the illumination path of a vision system, half of the light it receives from the lamps is immediately discarded and must be disposed of properly. To absorb the large quantities of unwanted light that this type of system produces requires that special techniques be used. The naïve approach to producing a light absorbing surface is to use commercially available matt black paint. However, this still reflects a significant amount of light. Black velvet provides an efficient light absorbing surface but this is hardly practical for use in a dirty factory environment. Although amorphous carbon (soot) is a good absorber, it is friable. A standard approach to improving matters in photographic lens hoods, is to use rifling on the inside surface. A closed box with a small aperture in its side wall and painted matt black inside is a good absorber. A deep hollow cone is too (➋ *Fig. 5.21*). Any light reflected by the inside surface of the cone is reflected deeper into it, where more of the unwanted light will be aborbed. This leads to the idea of building a surface consisting of a series of small conical depressions. A similar idea is to use a stack of razor blades; the side of the stack is a good light absorber. Creating a sponge-like structure from a material that itself has a low reflection coefficient seems to offer the best approach to building a really efficient light absorber. One recently announced coating material aborbs 10–20 times more less light than conventional techniques [1]. (It is claimed by its inventors to be 25 times blacker than conventional black paint.) This coating consists of an alloy of nickel and phosphorus. The surface is pitted, with tiny craters. At the time of writing (April 2008) this material is expensive and not yet available in large quantities.

5.3 Optical Coatings

There are three reasons for using coated optics:

(a) To minimise reflections and thereby increase overall light transmission through an optical system

(b) To maximise the reflection of light from mirror surfaces



■ Fig. 5.21
Light absorbing structures. In both cases, the internal surface is coated with absorbent material, such as matt black paint. (a) Box, (b) Deep conical pit

(c) To protect the optical surfaces (both reflective and transmissive) from scratching and staining due to contamination

We discuss coatings for reflective surfaces in ➲ Sect. 5.5. Protective coatings require little explanation, except to emphasise that the ability to operate an optical system in a hostile environment is an essential requirement for many Machine Vision systems. The scientific basis of anti-reflection coatings is far from simple. We shall therefore concentrate here on the rationale for and use of coated optics.

The vast majority of optical components are made of various types of glass, coated with thin layers of special materials. The purpose of a coating is to modify the reflection and/or transmission properties of the component's surface. Whenever light passes from one type of material into a medium with different optical properties (most notably refractive index), part of the light is reflected and part of the light is transmitted. The intensity ratio of the reflected and transmitted light is primarily a function of the change in refractive index between the two media. The angle of incidence of the light at the interface is also important but, for the sake of simplicity, we shall assume normal incidence. Let n_1 be the refractive index of the material that the incident beam traverses prior to meeting the interface. Also, let n_2 be the refractive index after the interface. Then, the reflection coefficient is given by the equation

$$R = \frac{(n_2 - n_1)^2}{(n_2 + n_1)^2} \tag{5.11}$$

If the first medium is air, $n_1 \approx 1$, this equation reduces to

$$R = \frac{(n_2 - 1)^2}{(n_2 + 1)^2} \tag{5.12}$$

As the light beam emerges from the same optical component, the reflection coefficient is again equal to R. The insertion loss for that component is therefore R^2. For standard optical quality glass, the refractive index is about 1.5. (➲ *Table 5.1*) Hence, the insertion loss is 7.84%.

If the reflection coefficient of the ith surface is R_i and there are N surfaces in a transmissive (i.e., no mirrors) optical system, the intensity of the emergent beam is

$$I_0 \prod\nolimits_{i=1}^{N} (1 - R_i) \tag{5.13}$$

where I_0 is the intensity of the incident beam. An uncoated surface of clean polished optical quality glass might well reflect 4% ($R = 0.04$) of the incident light. Suppose that $N = 10$, a typical value in a simple system. Then, the intensity of the transmitted light is $0.665I_0$. In most industrial vision systems, this degree of attenuation is not critical, as there is power available to flood a small arena with plenty of light. However, what is important is where the "lost" light goes. In our example, on third of the light is reflected back into the optical system, where it has many opportunities to degrade the image contrast. In ➲ *Fig. 5.22*, it is clear that even a single layer coating of magnesium fluoride (MgF_2) can reduce the reflection coefficient at each surface to about 0.015. In this case, the intensity of the emergent beam is $0.86I_0$. The "lost" light has however, been reduced from 33% to 14%.

The only suitable applications for uncoated optics are those where only a few optical components are in the optical path, and significant transmission inefficiencies can be tolerated. In principle, the surface of any optical element can be coated with thin layers of various materials (called thin films) in order to achieve the desired reflection/transmission ratio. With the exception of simple metallic coatings, this ratio depends on the nature of the material from which the optical device is fabricated, the wavelength of the incident light, and the angle of

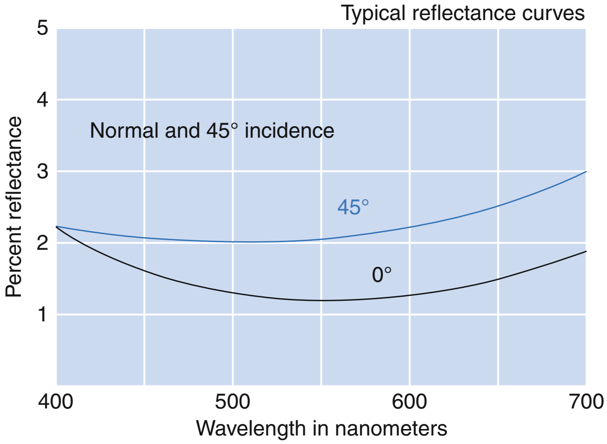


Fig. 5.22
Reflectance characteristics from a single layer of MgF_2

incidence of the light (measured from the normal to the optical surface). There is also a polarisation dependence to the reflection/transmission ratio when the angle of incidence is not normal to the surface. A multilayer coating, sometimes made up of more than 100 individual fractional-wavelength layers, may be used to optimise the reflection/transmission ratio for a specific wavelength and angle of incidence or to optimize it over a specific range of conditions. Today's multilayer dielectric coatings are remarkably hard and durable. With proper care and handling, they can have a long life. In fact, the surfaces of many high-index glasses that are soft or prone to staining can be protected with a durable antireflection coating. Several factors influence coating durability. Coating designs should be optimized for minimal overall thickness to reduce mechanical stresses that might distort the optical surfaces or cause detrimental polarisation effects. The most resilient materials should be used. Great care must be taken in coating fabrication to produce high-quality, nongranular, even layers.

5.3.1 Antireflection Coatings

Magnesium fluoride (MgF_2) is commonly used for single-layer antireflection coatings because it has an almost ideal refractive index (1.38 at 0.550 μm) and high durability. These coatings are insensitive to wavelength and angle of incidence. Single-layer antireflection coatings are routinely available for almost any angle of incidence and any wavelength in the range 0.20–1.6 μm. (However, as the upper wavelength limit is approached, the angle of incidence should be restricted to near-normal.) The performance of MgF_2 antireflection coatings improves dramatically as the substrate's index of refraction increases.

5.3.2 Multi-Layer Coatings

Although MgF_2 does not offer the same performance as multilayer coatings, it is preferred in some circumstances. On lenses with very steep surfaces (e.g., close to the edges of some aspheric lenses), MgF_2 will actually perform better than a multi-layer coating. Whatever the angle of incidence, MgF_2 always performs at least as well as an uncoated lens but, at very high angles of incidence, multi-layer coatings may be worse than no coating at all.

The so-called quarter/quarter coating was developed to circumvent the lack of a material with the index of refraction needed to optimise the performance of a single-layer coatings. The basic problem associated with single-layer antireflection coatings is that the refractive index of the coating material is generally too high, resulting in too strong a reflection from the first surface which cannot be completely cancelled through destructive interference with the weaker reflection from the substrate's top or first surface. In a two-layer coating, the first reflection is cancelled through destructive interference with two weaker out-of-phase reflections from underlying surfaces. A quarter/quarter coating consists of two layers, both of which have an optical thickness of a quarter wave at the wavelength of interest. The outer layer is made of a low-refractive-index material, and the inner layer is made of a high-refractive-index material, compared to the substrate. As illustrated in ➋ *Fig. 5.23*, the second and third reflections are both exactly 180° out of phase with the first reflection. The interference effect in thin films is, of course, sensitive to wavelength. In other words, the reflected waves only cancel one another exactly at one specific wave-length.

When considering a two-layer quarter/quarter coating optimized for one wavelength at normal incidence, the required refractive indexes for minimum reflectivity can be calculated easily by using the following equation:

$$n_0 = \frac{n_1^2 n_3}{n_2^2} \tag{5.14}$$

Here n_0 is the refractive index of air (approximated as 1.0), n_3 is the refractive index of the substrate material, and n_1 and n_2 are the refractive indices of the two film materials. If the substrate is crown glass with a refractive index of 1.52 and if the first layer is the lowest possible refractive index, 1.38 (MgF_2), the refractive index of the high-index layer needs to be 1.70. Either beryllium oxide or magnesium oxide could be used for the inner layer, but both are soft materials and will not produce very durable coatings. Although it allows some freedom in the choice of coating materials and can give very low reflectance, the quarter/quarter coating is constrained in its design owing to the lack of materials with suitable refractive index and physical or durability properties.

In principle, it is possible to deposit two materials simultaneously to achieve layers of almost any required refractive index, but such coatings are not very practical. As a consequence, thin-film engineers have developed multilayer and special two-layer antireflection coatings that allow the refractive index of each layer and, therefore, coating performance to be optimized.

5.3.3 Two-Layer Coatings of Arbitrary Thickness

Optical interference effects can be characterized as either constructive or destructive interference, where the phase shift between interfering wavefronts is 0° or 180° respectively. For two wavefronts to cancel each other completely, as in a single-layer antireflection coating, a phase shift of exactly 180° is required. Where three or more reflecting surfaces are involved, complete cancellation can be achieved by carefully choosing the relative phase and intensity of the

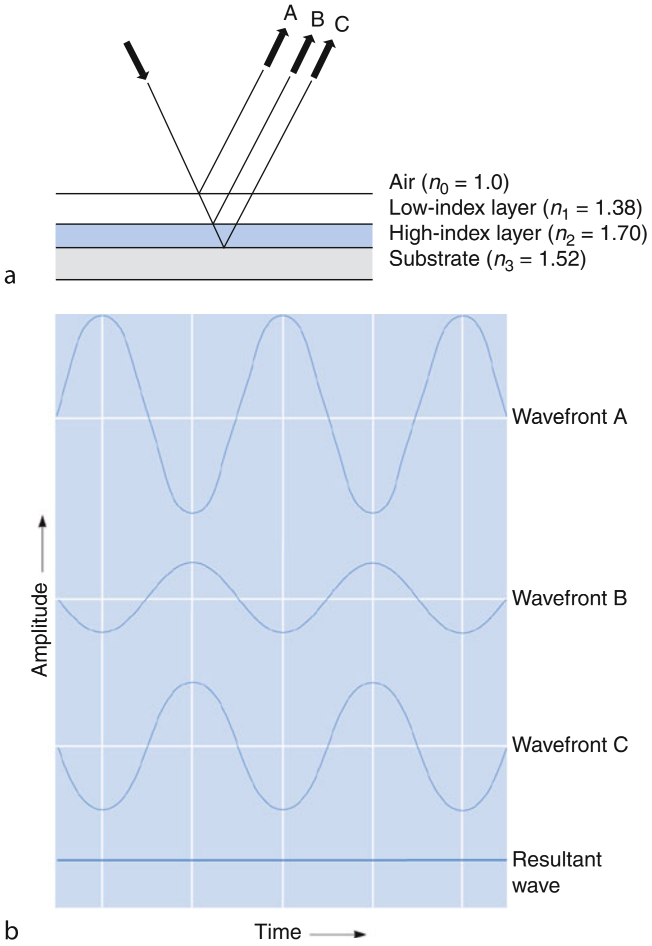


Fig. 5.23
Interference in a typical quarter/quarter coating

interfering beams (i.e., optimizing the relative optical thicknesses). This is the basis of a two-layer antireflection coating, where the layers are adjusted to suit the refractive index of available materials, instead of vice versa. For a given combination of materials, there are usually two combinations of layer thicknesses that will give zero reflectance at the design wavelength. These two combinations are of different overall thickness. For any type of thin-film coating, the thinnest possible overall coating is used because it will have better mechanical properties (less stress). A thinner combination is also less wavelength sensitive. Two-layer antireflection coatings are the simplest of the so-called V-coatings. The term V-coating arises from the shape of the reflectance curve as a function of wavelength, as shown in *Fig. 5.24*, which resembles a skewed V shape with a reflectance minimum at the design wavelength. V-coatings are very popular, economical coatings for near monochromatic applications.

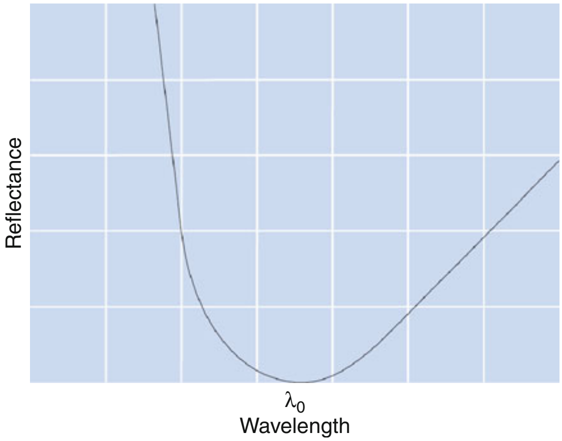


Fig. 5.24
Characteristic performance curve of a V-coating

5.3.4 Broadband Anti-Reflection Coatings

Imaging systems typically use polychromatic (more than one wavelength) light. In order for the system to have a flat response over an extended spectral region, transmitting optics are coated with a dichroic broadband antireflection coating. The main technique used in designing antireflection coatings that are highly efficient at more than one wavelength is to use “absentee” layers within the coating. Additional techniques can be used for shaping the performance curves of high-reflectance coatings and wavelength-selective filters, but these are not applicable to antireflection coatings.

5.3.5 Absentee Layers

An absentee layer is a film of dielectric material that does not change the performance of the overall coating at one particular wavelength. Usually that particular wavelength is the wavelength for which the coating is being optimized. The absentee layer is designed to have an optical thickness of a half wave at that specific wavelength. The “extra” reflections cancel out at the two interfaces because no additional phase shifts are introduced. In theory, the performance of the coating is the same at that specific design wavelength whether or not the absentee layer is present. At other wavelengths, the absentee layer starts to have an effect for two reasons: the ratio between physical thickness of the layer and the wavelength of light changes with wavelength, and the dispersion of the coating material causes optical thickness to change with wavelength. These effects give the designer extra degrees of freedom not offered by simpler designs.

The complex, computerized, multilayer antireflection coating design techniques used by manufacturers are based on the simple principles of interference and phase shifts described above. In view of the properties of coherent interference, it is meaningless to consider individual layers in a multilayer coating. Each layer is influenced by the optical properties of the other layers in the multilayer stack. A complex series of matrix multiplications, in which each matrix corresponds to a single layer, is used to model the performance of multilayer thin-film coatings mathematically. There also are multiple reflections within each layer of a coating. In the previous discussions, only first-order or primary reflections were considered. This oversimplified approach is unable to predict accurately the true behaviour of multilayer coatings. Second-, third-, and higher-order terms must be considered if the real behaviour of a coating is to be modelled accurately.

5.4 Lenses

Most lenses have spherically curved surfaces; either surface can be convex, concave, or planar. A lens is classified by the curvature of its optical surfaces (➋ *Fig. 5.25*).

- B*iconvex*, both surfaces are convex
- *Biconcave*, both surfaces are concave
- *Plano-convex*, it has one plane and one convex surface
- P*lano-concave*, it has one plane and one concave surface
- *Convex-concave, it has* one convex and one concave surface
- M*eniscus lens*, convex-concave type, where both surfaces have the same radius of curvature

The terminology used when discussing lenses is summarised in ➋ *Fig. 5.26*.

If a lens is biconvex, or plano-convex, a paraxial, collimated beam of light will be focussed behind the lens to a point on its axis called the *focal point*. (A *collimated beam* consists of a set of parallel rays. *Paraxial* means that all rays are parallel to the optical axis.) Such a lens is said to be a positive or *converging lens* (➋ *Fig. 5.27*). The distance between the lens and its focal point is called the *focal length* and is usually denoted by *f*.

A biconcave, or plano-concave, causes a paraxial beam to diverge as it emerges from the lens. This is called a negative or *diverging lens*. The beam emerging from the lens appears to be emanating from a point on the axis in front of the lens (➋ *Fig. 5.28*). This distance from this virtual focal point to the lens is also known as the focal length. In this case, it has a negative value.

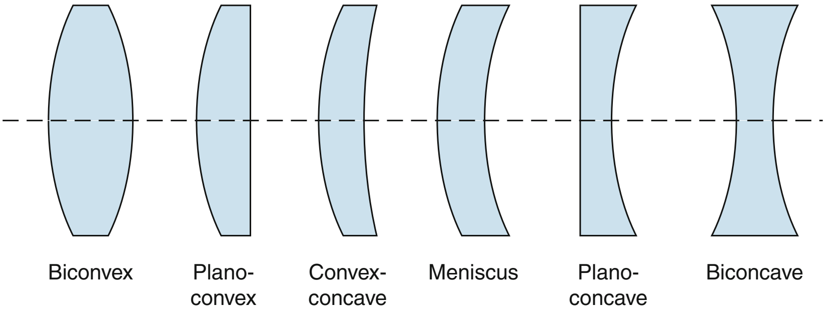
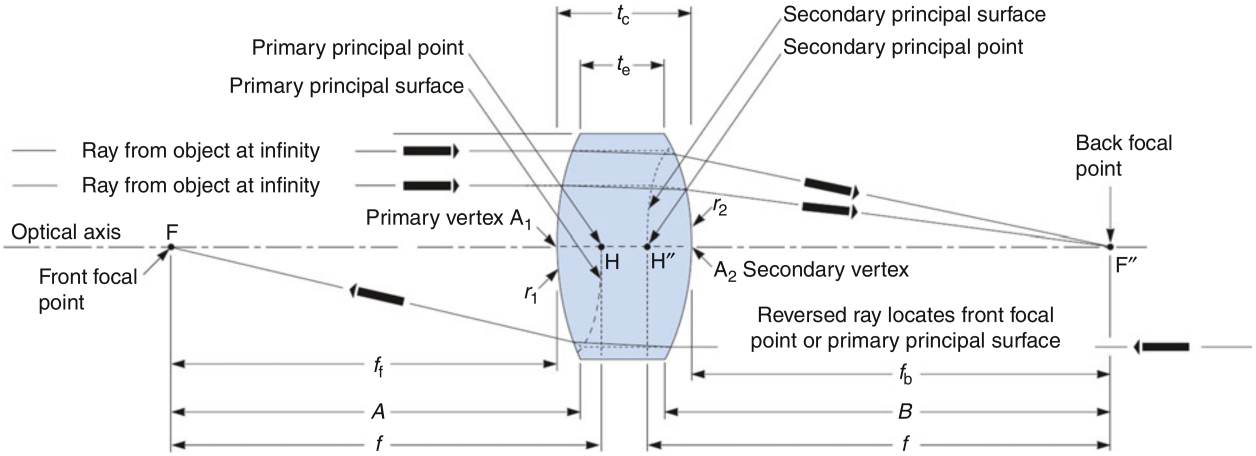


Fig. 5.25
Types of lenses


t_c
t_e
Primary principal point
Primary principal surface
Secondary principal surface
Secondary principal point
Ray from object at infinity
Ray from object at infinity
Optical axis
F
Front focal point
Primary vertex A_1
r_1
r_2
H
H″
A_2 Secondary vertex
Back focal point
F″
Reversed ray locates front focal point or primary principal surface
f_f
A
f
f_b
B
f

A = Front focus to front edge distance

B = Rear edge to rear focus distance

f = Effective focal length; may be positive (as shown) or negative

f_f = Front focal length

f_b = Back focal length

t_e = Edge thickness

f_c = Center thickness

r_1 = Radius of curvature of first surface (positive if center of curvature is to right)

r_2 = Radius of curvature of second surface (negatove if center of curvature is to left)

Fig. 5.26
Terminology for lenses

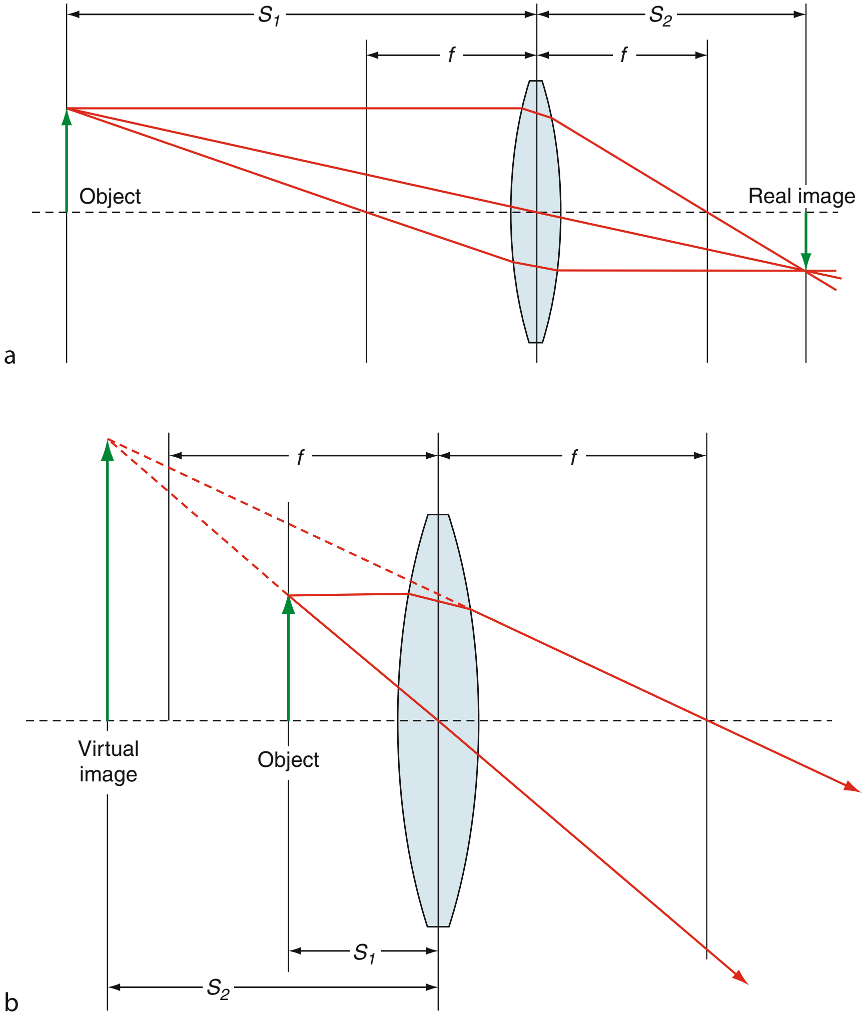


Fig. 5.27

Creating real and virtual images with a thin biconvex (i.e., converging) lens. (a) The object lies beyond the focal point. The image is real, inverted and is reduced in size. (b) The object lies between the vertex and focal point. The virtual image is not inverted and is magnified

A convex-concave lens may be converging or diverging, depending on the relative curvatures of the two surfaces. The beam is neither converged nor diverged by a meniscus lens. A paraxial beam remains paraxial but its diameter changes as a result of travelling through such a lens.

The line joining the centres of the two spherical lens surfaces is called the *axis*, or *optical axis*. Most lenses are symmetrical about the optical axis, so the focal points both lie on it. The *vertices* are those points where the axis intersects the surfaces of the lens.

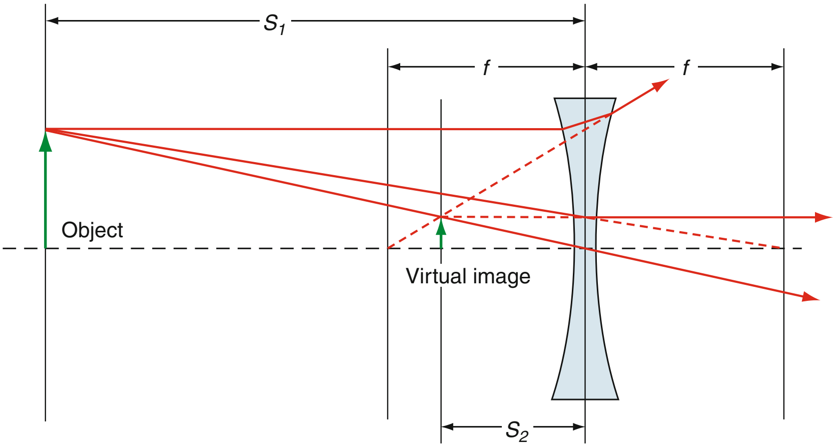


Fig. 5.28

Diverging biconcave lens. The object lies beyond the focal point. The image is virtual, upright and smaller than the object. If the object is closer than the focal point (not shown), the image is virtual, upright and enlarged. (To appreciate this, mentally reverse the direction of travel of the light rays and interchange the image and object.)

5.4.1 Lens Maker's Equation

The focal length of a lens in air can be calculated from the *lens maker's equation*:

$$\frac{1}{f} = (n-1)\left[\frac{1}{R_1} - \frac{1}{R_2} + \frac{(n-1)d}{nR_1R_2}\right], \tag{5.15}$$

where *f* is the focal length of the lens *n* is the refractive index of the lens material. (The refractive index of a vacuum is defined as being 1. The refractive index of air is 1.0008, so for most practical purposes this is taken to be 1. The refractive index of most optical glasses is about 1.5.) R_1 is the radius of curvature of the lens surface closer to the light source, R_2 is the radius of curvature of the lens surface farther from the light source*d* is the thickness of the lens (i.e., the distance between the two vertices).

The signs of the radii of curvature indicate whether the lens surfaces are convex or concave. Several sign conventions exist but in this chapter R_1 is positive if the first surface is convex and negative if it is concave. The signs are reversed for the second surface of the lens: if R_2 is positive the surface is concave, and negative if it is convex. If either R_1 or R_2 is infinite, the corresponding surface is flat.

If d is small compared to R_1 and R_2 (*thin lens condition*), we obtain the following approximation to ➋ Eq. 5.1:

$$\frac{1}{f} \approx (n-1)\left[\frac{1}{R_1} - \frac{1}{R_2}\right]. \tag{5.16}$$

Notice that the focal length *f* is positive for converging lenses, negative for diverging lenses, and infinite for meniscus lenses. The value of 1/*f* indicates the *optical power* of the lens and is measured in *diopters* (or m^{-1}).

5.4.2 Imaging Properties of Lenses

The plane perpendicular to the lens axis and situated at a distance f from the lens is called the *focal plane*. A we have already stated, a converging lens focuses a paraxial beam to a spot at a distance f from the lens. Conversely, a point source placed at the focal point will form a collimated (paraxial) beam of light as it emerges from the lens.

Let us denote the distance from the object to the lens by S_1 and that from the lens to the image by S_2. Then, for a lens of negligible thickness, in air, we can use the so-called *thin lens formula*:

$$\frac{1}{S_1} + \frac{1}{S_2} = \frac{1}{f} \tag{5.17}$$

If an object is placed at a distance $S_1 > f$ along the axis in front of a positive lens of focal length f, a *real image* is created. This means that it can be used to project an image onto a screen placed at a distance S_2 behind the lens (➋ *Fig. 5.27a*). ➋ Equations 5.16 and ➋ 5.17 are based on the assumption that the lens has zero thickness and are useful for approximate calculations.

If $S_1 < f$, S_2 is negative and a *virtual image* is formed. This cannot be projected onto a screen. However, an observer looking through the lens will see the magnified upright image, apparently located on the far side of the lens. A magnifying glass creates this kind of image.

The *magnification* of the lens is given by M, where:

$$M = -\frac{S_2}{S_1} = \frac{f}{f - S_1} \tag{5.18}$$

If the absolute vale of M (i.e., $|M|$) is greater than 1, the image is larger than the object. If M is negative, the image is inverted. For virtual images, M is positive and the image is upright. If S_1 is infinite, $S_2 = f$ and M is zero. This indicates that a collimated beam is focussed to a single spot at the focal point. (In practice, the size of the image is not zero, since diffraction effects limit the minimum size of the image.)

These formulae may also be used for negative (i.e., diverging) lenses, which can only create virtual images (➋ *Fig. 5.4*). In this case, we take the focal length (f) to be negative.

The plane at which paraxial rays appear to "bend," to converge the focal point, is called the *principal plane*. For a lens in air, this plane is often called the *nodal plane*. Any lens, except a hypothetical "thin" lens, will typically have multiple principal planes (➋ *Fig. 5.29*).

The *f-number* (*f*/#) of a lens is the focal length divided by the diameter of its aperture (*limiting aperture*) of the lens. Hence, a lens with a 50 mm focal length and a 10 mm aperture diameter is referred to as an *f*/5 lens. The *f*/# is useful in determining the relative illuminance of the image formed by the lens; the illuminance varies as the inverse of *f*/# squared. The focal length and *f*-number of a commercial lens are usually printed on the lens barrel.

5.4.3 Analysing a Compound Lens

It is possible to analyse a compound (i.e., multi-element) lens by tracing the path of rays through a series of simple lenses. We can calculate the magnification, field-of-view and aberrations of the compound lens. It is usually most convenient to perform the analysis by considering the simple lenses in the reverse of the order in which light passes through them.

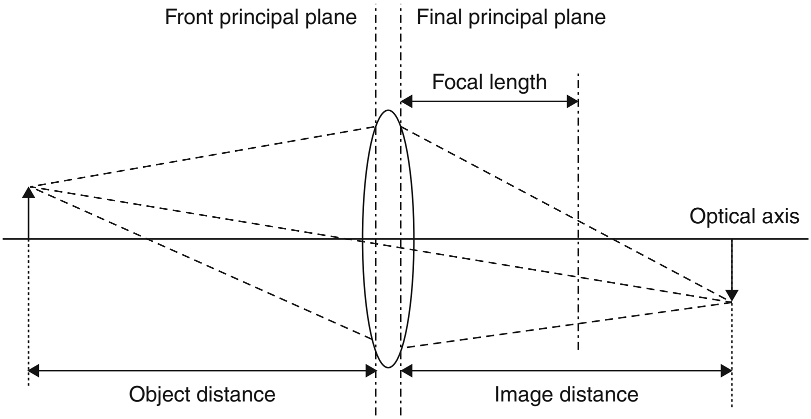


Fig. 5.29
Principal planes for a converging lens

Figure 5.30 explains how the focal length is calculated. Notice that the principal planes of the compound lens are the same as those of the end elements: lenses L_1 and L_3 in Fig. 5.30a; L_4 and L_6 in Fig. 5.30b.

5.4.4 Scheimpflug Condition

Tilting the lens relative to the object-lens axis will cause part of the image to be focus and other parts out of focus (Fig. 5.31a). This will also introduce trapezoidal distortion; a square forms an image in the form of a key-stone.) However, if we satisfy the so-called *Scheimpflug condition*, the whole image can be kept in focus (Fig. 5.31b).

5.4.5 Depth-of-Field and Depth-of-Focus

The *depth-of-field* of a lens is the distance along its optical axis through which the object can be moved while still forming a sharply focussed image (Fig. 5.32). The *depth-of-focus* is the distance along the optical axis that the sensor can be moved while keeping the image in focus. To quantify these, we introduce the concept of the *circle of confusion*, or *blur circle*. Imagine an optical system focussing light from a point source. When the system is slightly misfocussed, the image will be a small but finite-sized spot. If this is very small, the mis-focussing can be neglected. The size of the spot (i.e., the blur circle) is a measure of the level of mis-focussing. In many cases, we can take it that the maximum allowable size for the blur circle is the same as that of the individual photo-detectors in the sensor array. In this case, the depth-of-focus is equal to the f-number (*f*/#) times the size of one individual photo-detector.

Determining the depth-of-field is slightly more complicated. The distance of the far focal point (P in Fig. 5.32) from the lens is

$$D_{\mathrm{far}} = \frac{S_1 f_2}{f^2 - NC(S_1 - f)} \tag{5.19}$$

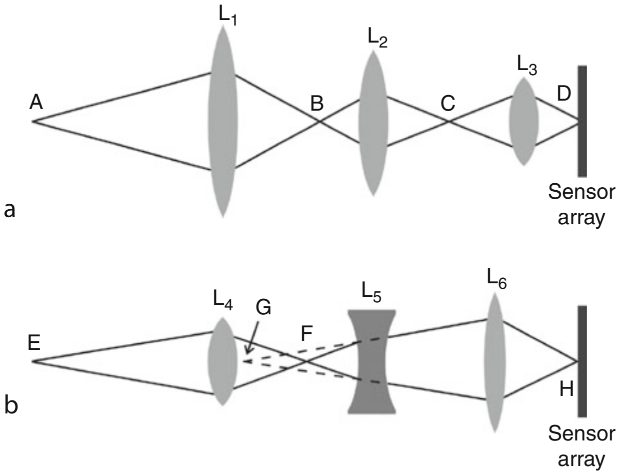


Fig. 5.30

A compound (i.e., multi-element) lens can be analysed by considering it as a chain of simple lenses. (a) All three component lenses are converging. An object at point A is focussed by lens L_1 at point B. Rays passing through point B are focussed at C by lens L_2. Rays passing through point C are focussed at D, on the sensor array by lens L_3. A real image of A appears at the sensor array. (b) A chain containing one diverging lens and two converging lens, forms a converging compound lens. E is focussed at F by lens L_4. Lens L_5 forms a virtual image at G, which is focussed by lens L_6 to form a real image at H, on the sensor array

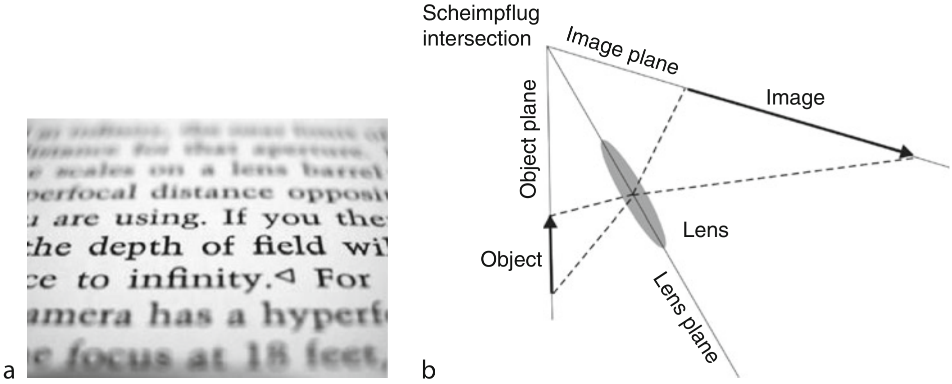


Fig. 5.31

Scheimpflug condition. (a) Effect of tilting a lens. (b) The Scheimpflug condition: the object plane, lens plane and image plane all meet in a single straight line. The normal cross section is shown here

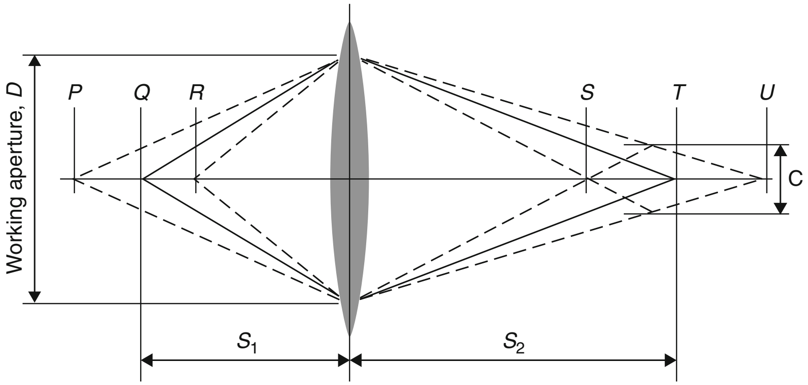


Fig. 5.32
Depth-of-field and depth-of-focus. A point in the plane Q is optimally focussed to a point in plane T. The object can be moved anywhere in the region PR while still maintaining an acceptably sharp focus. This defines the depth-of-field. An object at Q will be seen as being in acceptably sharp focus by a sensor placed anywhere within the region SU. This defines the depth-of-focus

and the distance of the near focal point (R in *Fig. 5.32*) from the lens is

$$D_{\mathrm{near}} = \frac{S_1 f^2}{f^2 + NC(S_1 - f)} \tag{5.20}$$

Where N is the f-number ($f/\#$) of the lens, D is the working diameter of the lens (usually less than its physical diameter), C is the diameter of the blur circle in the image plane, S_1 is the object-lens distance.

The depth-of-field (DoF) is the difference between D_{far} and D_{near}.

$$\mathrm{DoF} = \frac{2NCS_1 f^2\,(S_1 - f)}{f^4 - N^2C^2\,(S_1 - f)^2} \tag{5.21}$$

5.4.6 Aberrations and Resolution of an Optical System

Degradations in the quality of the image can be caused by imperfections in the lens. However, even a good lens, used in an inappropriate way will produce a poor quality image. The most important aberrations for Machine Vision are summarised below. Higher order aberrations are usually of less concern.

5.4.6.1 Spherical Aberration

Lenses with spherical surfaces are not ideal. They are widely used, because they are relatively simple to form, grind and polish. The effect of spherical aberration caused by a biconvex lens is shown in *Fig. 5.33*. Beams parallel to but away from the lens axis are focussed at different points along that axis. This results in a blurring of the image formed in the (optimal) focal plane. A paraxial beam is focussed to a spot of finite size. *Aspheric lenses* have non-spherical surfaces and can reduce spherical aberration very considerably. (Hitherto, they were difficult to make and consequently were expensive. Modern manufacturing methods have greatly reduced the cost.) Spherical aberration can be minimised by careful choice of the lens. For example, a plano-convex lens, focussing a collimated beam, produces a smaller focal spot when the light strikes the convex surface first. The size of the focal spot circle decreases as the cube of *f*/#, So, stopping the lens down greatly reduces the mis-focussing effect. Spherical aberration is important in applications requiring high spatial resolution, for example when trying to detect small "point" or narrow streak defects (e.g., cracks or scratches).

5.4.6.2 Coma

Another common type of aberration is coma. It produces a comet-like spot in the focal plane, when a parallel but non-paraxial beam is focussed (➋ *Fig. 5.34*). Suppose the rays travel at some non-zero angle θ relative to the optical axis. Rays that pass through the centre of the lens are focused to a point at a distance of $f.\tan(\theta)$ from the axis. (Remember this is a thin lens.) On the other hand, rays passing through the lens periphery are focused at different points. These are either further from the axis (positive coma) or closer to the axis (negative coma). A circular beam containing parallel rays, passing through the lens at a fixed distance from its centre, forms a disc-shaped image in the focal plane. This is known as a *comatic circle*. The superimposition

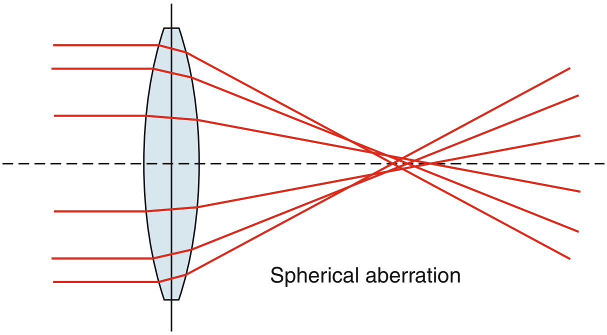


Fig. 5.33
Spherical aberration

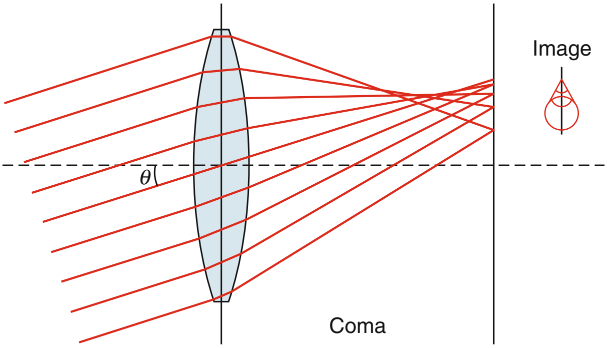


Fig. 5.34
Coma

of all these circles results in a comet-like spot. As with spherical aberration, coma can be reduced by using aspheric lenses. For lenses with spherical surfaces, the magnitude of the comatic aberration increases linearly with distance from the optical axis, but decreases as the square of *f*/#. Reducing comatic aberration is important when it is necessary to provide constant medium-to-high resolution over the whole visual field.

5.4.6.3 Chromatic Aberration

Think of a biconvex lens as approximating two prisms stuck together. With this in mind, we should not be surprised that the lens will disperse white light, into its spectral components. A dark opaque object, viewed in silhouette against a white light source through the lens, will appear to have a multi-coloured periphery. This is the manifestation of the effect known as *chromatic aberration* and is caused by the fact that the refractive index of the lens material varies with the wavelength of light. Hence the focal length (*f*) of the lens depends upon the wavelength. Chromatic aberration can be reduced by using an *achromatic doublet* (or *achromat*) lens (➜ *Fig. 5.35*). Two lenses made of different materials are cemented together and are treated conceptually as a single lens. *Achromatic* lenses are only able to compensate for

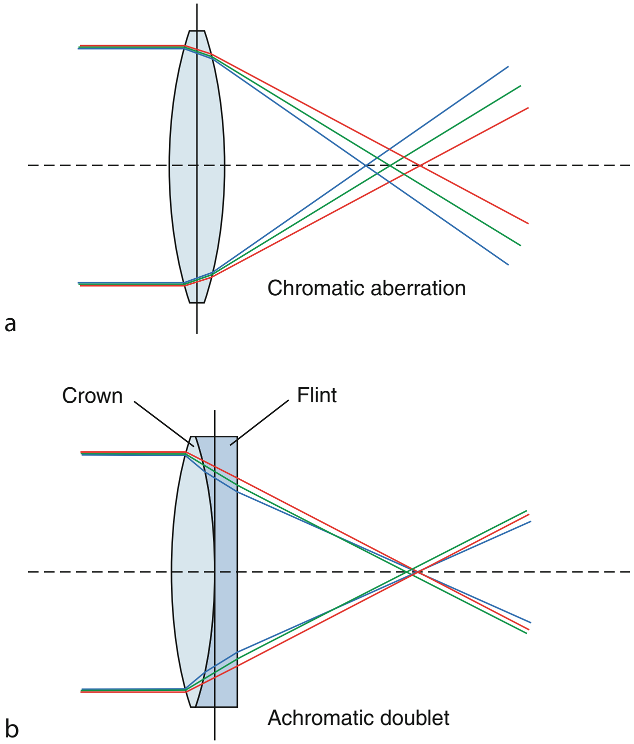


Fig. 5.35
Chromatic aberration. (a) The focal length of a lens depends upon the wavelength. (b) By combining two lenses with different forms, it is possible to reduce chromatic aberration

chromatic aberration over a limited range of wavelengths and normally do not produce perfect correction over the full visible spectral band. An *apochromat* is a lens (or lens system) that is corrected for both chromatic and spherical aberrations. Reducing chromatic aberration to an absolute minimum is clearly essential for accuracy in tasks such as colour inspection. It is also important where high spatial resolution is needed.

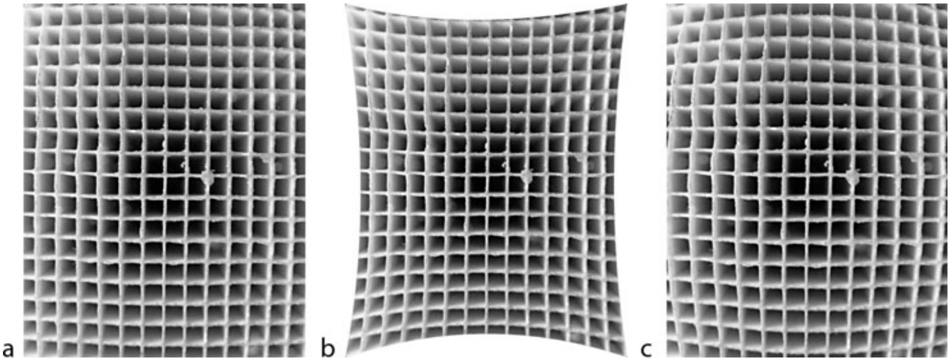
5.4.6.4 Other Aberrations

Field curvature is manifest when evenly spaced points focus at a variable distance from the lens. The effect decreases as the lens is stopped down and is important in any precise measurement or position-locating application.

Astigmatism occurs when horizontal and vertical features focus at different distances from the lens. This results in the resolution being different for horizontal and vertical size measurements.

Distortion Features in the image become noticeably mis-shaped. Important for most vision applications, especially those requiring the measurement of shape/size of discrete parts (➲ *Fig. 5.36*).

Vignetting This occurs when a uniformly illuminated scene is mapped to an image that is uneven in brightness, usually bright in the middle and darker towards the sides corners (➲ *Fig. 5.37*). Vignetting is not strictly an aberration; it occurs when there is a different apparent aperture size for off-axis and on-axis rays. Poorly designed multi-lens assemblies cause vignetting; rear elements are partially shaded by elements in front of them. This reduces the effective lens opening for off-axis rays. This form of vignetting can be cured by stopping down the lens. Lenses are designed for a particular image format. Provided they are used for their intended purpose, most lenses will produce a fairly uniform image brightness. Telecentric lenses (described later) are often designed specifically to provide very even image brightness. Vignetting also occurs when peripheral light rays are blocked by external objects, such as secondary lenses, or inappropriately fitted lens hoods. The corner darkening can be gradual or abrupt, depending on the lens aperture. Vignetting is important because it renders fixed-level image thresholding impractical.



■ Fig. 5.36

Distortion. (a) Original. Catalytic converter. The Barrel distortion present here is due to the lens. (b) Pin-cushion distortion produced by software. (c) Barrel distortion exaggerated by software

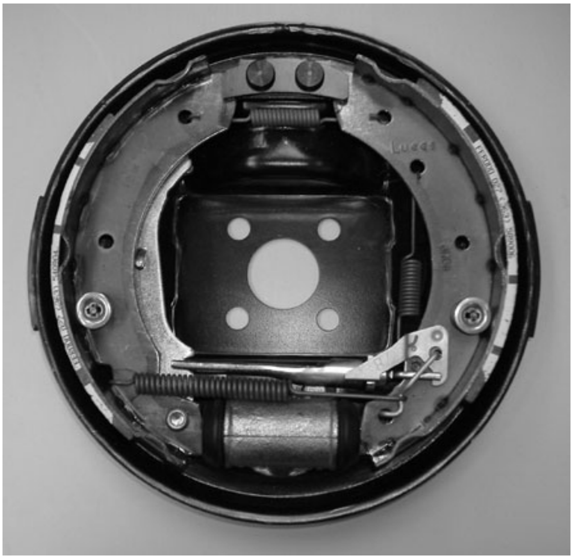


Fig. 5.37

Vignetting. Close to the *right-hand side*, the intensity of the background varies from 110 (*top and bottom corners*) to 157 (*centre-right*) (Gradual changes in shading like this are not immediately obvious to the human eye, although this represents a contrast ratio of 1.42)

5.4.7 Resolution

The resolution of an optical system is defined in terms of its ability to distinguish between two closely spaced points. Following the example of John Strutt (Lord Rayleigh), it is customary to take the limit of resolution to be that occurring when their blur circles are just touching. A (single) beam of light of diameter *A* can be focussed to a spot of diameter not less than

$$\frac{1.22\lambda f}{A} \tag{5.22}$$

where is the wavelength and *f* is the focal length of the lens. Strictly, *A* is the original beam diameter, or the aperture of the lens if that is smaller. Hence, the minimum value for this limit is equal to 1.22 times the *f*/#. This limit is a fundamental limit, imposed by diffraction effects. Hence, it is known as the *diffraction limited resolution* of the lens. The diffraction-limited image is actually a bright spot, surrounded by a series of concentric rings, whose brightness varies inversely with their radius. This is known as the Airy disc. The disc whose diameter given by Eq. 5.22 accounts for about 70% of the light energy in the beam. A disc of twice that diameter contains 83.8% of the light energy. This is used sometimes to define the diffraction-limited spot size. Expression 6.22 is a theoretical limit, which is very difficult to achieve in practice. Lenses that are claimed to provide a resolution even close to this limit should be avoided!

Resolution should not be confused with the ability to detect something. The latter depends on both the optical system and the sensor. The human eye can easily detect a bright star in the night sky, but can only do so because the lens is imperfect! As far as a human being on earth is concerned, the star is an extremely small source of light that is imaged by the lens of the eye to form the blur circle. We incorrectly interpret this as the image of the star. On the other hand, in a video system, the smallest resolvable object is usually determined by the size of an individual photo-detector; most lenses have no problem producing a blur circle that is much smaller than the camera can use.

5.4.8 Modulation Transfer Function (MTF)

The modulation transfer function (MTF)is a sensitive quantitative measure of image quality and describes the ability of a simple lens, or a complete imaging sub-system, to maintain high contrast for small features. Standard bar-chart testing of the resolution of a lens/system is deceptive, because almost 20% of the optical energy is contained within the third harmonic and higher spatial frequencies. A sine-wave test chart, like that shown in *Fig. 5.38a* is used to measure the MTF. This chart may be in the form of a transparency, or a high-quality printed card. The spatial frequency varies along one axis, which is calibrated in terms of cycles/mm. We shall denote the maximum and minimum transmittances of the transparency by $T_{\max}$ and $T_{\min}$. It is viewed against a uniformly illuminated background and the test lens forms a real image. For a given spatial frequency, the resulting image intensities $I_{\max}$ and $I_{\min}$ are measures. The contrast ratios of the chart and image are defined, thus:

$$\mathrm{CR}_{\text{chart}} = \frac{T_{\max} - T_{\min}}{T_{\max} + T_{\min}} \tag{5.23}$$

and

$$\mathrm{CR}_{\text{image}} = \frac{I_{\max} - I_{\min}}{I_{\max} + I_{\min}} \tag{5.24}$$

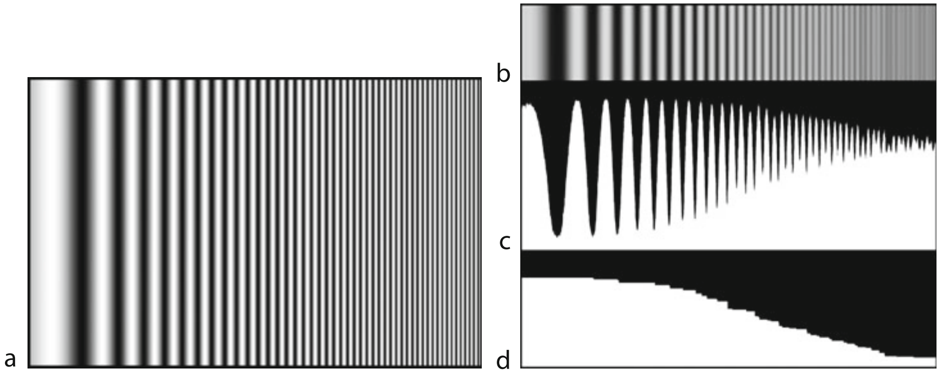


Fig. 5.38
Modulation Transfer Function (MTF) (a) Sine-wave test pattern. The horizontal scale is non-linear and may vary logarithmically, or as a polynomial ($X^{1.5}$ here). (b) The test pattern is photographed through the lens. (c) Intensity profile. (d) Contrast ratio versus spatial frequency. This is the MTF

The modulation transfer function (MTF) of the optical system is then defined to be

$$\mathrm{MTF} = \frac{\mathrm{CR}_{\mathrm{image}}}{\mathrm{CR}_{\mathrm{chart}}} \tag{5.25}$$

It should be understood that MTF is defined only for lenses, or optical systems that can form real images (i.e., have positive focal length). MTF is plotted against spatial frequency (➔ *Fig. 5.38d*). The resulting curve is very useful for assessing the quality of a lens.

It should be understood that the MTF curve is not a unique description of a lens, as it varies with the measurement conditions:

- Where the test chart is placed within the field. (Different MTF curves will be obtained at the centre and periphery of the visual field. Near the periphery, the MTF often varies with the orientation of the test chart.)
- The ratio of object distance to image distance. (This is known as the *conjugate ratio*. The object and its focussed image are called *conjugate pairs*.)

For this reason, MTF measurements are taken under standard conditions. Lenses with astigmatism or comatic aberration produce variable MTF curves, depending on the experimental arrangement. MTF curves are sometimes quoted for different wavelengths and will show the effect of chromatic aberration. While the MTF can be measured for a converging cylindrical lens, it must be remembered that it is only valid along one axis. The MTF can also be defined for concave mirrors (➔ Sect. 5.5).

The MTF is normally favoured over bar-chart test patterns for the reason given above. There are also theoretical advantages in using sine-wave test patterns, since the results can be analysed using Fourier transform methods. In practice, when a lens is used to image a point source, the result is an ill-defined spot of finite size. The intensity distribution of this spot defines the *point-spread function* and its 2-dimensional Fourier transform is called the *optical transfer function (OTF)* of the lens. The MTF is the magnitude of the OTF. The phase component of the OTF is unimportant in a non-coherent optical system, such as that found in a conventional Machine Vision installation. Other ways of measuring the response of a lens may use concentric-ring or radial sine-wave patterns (➔ *Fig. 5.39a–c*). It is simpler and cheaper, of course, to produce test patterns that consist only of binary elements: white and black. Hence, for many undemanding applications, testing the lens with a bar chart is perfectly adequate (➔ *Fig. 5.39d*). Again, this may consist of parallel stripes, radial patterns or concentric rings. The *Contrast Transfer Function (CTF)* is calculated from the image created by a bar chart in a similar way to the MTF. The resolution is normally expressed in terms of line pairs per mm.

5.5 Mirrors

5.5.1 Reflection

The familiar plane mirror is one of the simplest optical components. A ray of light striking the mirror at an angle θ is reflected so that the emergent ray is at an angle $-\theta$ (➔ *Fig. 5.40*). An important point to note is that the reflected light appears to come from a virtual image which is at the same distance behind the mirror as the object is in front of it. No light actually passes through the virtual image.

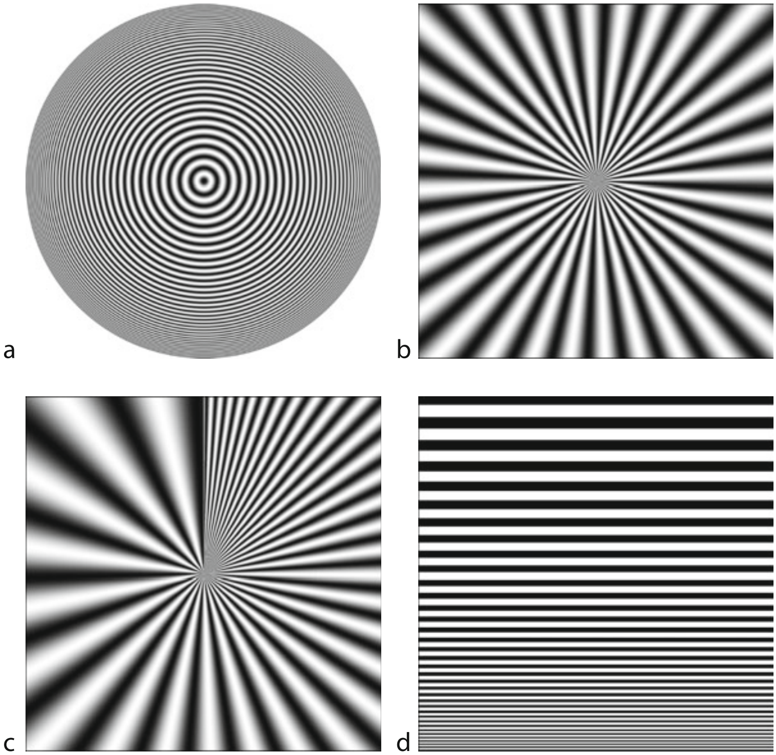


Fig. 5.39

Alternative patterns for testing optical systems. (a) Frequency varies with radius. (b) Angular frequency is constant. (c) Angular frequency varies logarithmically. (d) Bar chart (Binary image; usually calibrated in line pairs per mm)

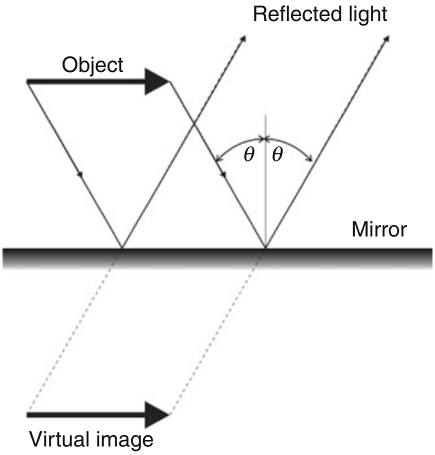


Fig. 5.40

Reflection from a plane mirror. Notice that the light appears to come from the virtual image but that no light actually travels through it

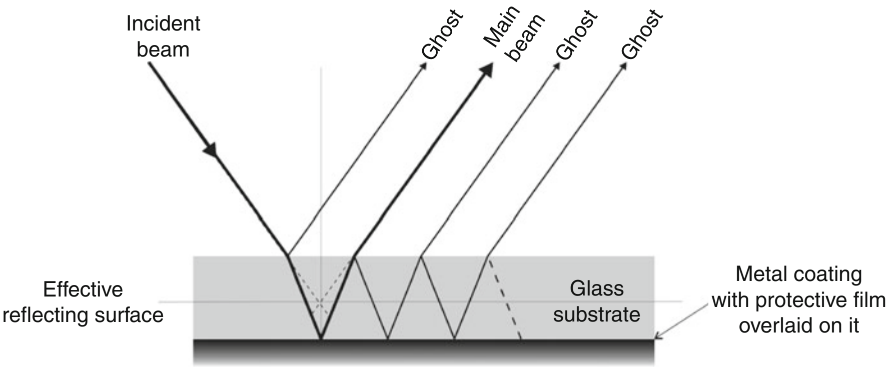

Incident beam
Ghost
Main beam
Ghost
Ghost
Effective reflecting surface
Glass substrate
Metal coating with protective film overlaid on it

Fig. 5.41
Ghosting in a second-surface mirror

Despite its familiarity, choosing a plane mirror is not as straightforward as we might suppose. Even a good quality domestic mirror can introduce significant aberrations or irregularities into an image. Standard, domestic quality mirrors are not accurate enough for most Machine Vision applications, since they rely on reflection from a "silvered" coating at the rear of a glass plate. Rear-silvered mirrors, also called *second-surface* reflectors, are liable to cause ghosting (➲ *Fig. 5.41*). A front-faced, or *first-surface*, reflector avoids this altogether but does expose the reflecting surface to contamination and physical damage. For certain low-resolution applications, where the image contrast is very high, it may be possible to employ domestic mirror tiles. However, high precision, first-surface mirrors should be used, unless there are clear indications that it is safe to do otherwise. For some applications a prism provides a better reflector that a simple coated plate of glass.

5.5.2 Factors to Be Considered When Choosing a Mirror

Five primary factors should be considered when specifying a mirror:

- Coating
- Substrate material
- Flatness or surface accuracy
- Cosmetic surface quality
- Shape and dimensions

5.5.3 Coatings for Mirrors

Metallic coatings are effective over a broad waveband and are relatively insensitive to the angle of incidence. The most commonly used coatings are summarised below. Multilayer dielectric coatings are able to maintain good performance over a specific range of wavelengths and are insensitive to small changes of angle of incidence. The improve durability and damage resistance. Like all optical surfaces, mirrors can easily be damaged by negligent, or over-zealous, cleaning.

Bare aluminium has a very high reflectance value but oxidizes over time. For this reason, protected aluminium is normally preferred.

Protected aluminium coating (➋ *Fig. 5.42*) is normally preferred to bare aluminium because its dielectric overcoat eliminates oxidation, with minimal sacrifice in performance. This is a good general-purpose coating for first-surface reflectors in the visible and near-infrared spectrum; the average reflectance is greater than 87% from 0.40 to 0.80 µm. Another type of dielectric coating prevents oxidation and maintains aluminium's reflectance in the UV waveband but must be cleaned with care.

Enhanced aluminium coating (➋ *Fig. 5.43*) is an aluminium coating overlaid with a durable multilayer dielectric film. This increases reflectance throughout the visible spectrum and provides protection from the environment and handling damage. The peak reflectance is 95% with average reflectance across the visible spectrum of 93%. Enhanced aluminium coatings are as durable and reliable as protected aluminium but have higher reflectance in the mid-visible region.

UV-enhanced aluminium (➋ *Fig. 5.44*) This provides an additional dielectric coating that prevents oxidation and preserves the reflectance of bare aluminium in the ultraviolet. The average reflectance is greater than 86% from 0.25 to 0.40 µm.

Bare silver tarnishes quickly in the atmosphere and hence is used only as a second surface coating, where it can be protected by an overcoat. It is often used on prism faces. Silver provides total internal reflection at high angles of incidence.

Protected silver (➋ *Fig. 5.45*) These coatings combine high reflectance throughout the visible and near infra-red spectral region with good durability. An overcoat and edge seal prevent the silver from oxidizing and deteriorating. Protected silver has higher reflectance than aluminium throughout the visible and near-infrared spectral region.

Bare Gold (➋ *Fig. 5.46*) This combines high reflectance throughout the near, middle, and far infrared with good tarnish resistance. However it is soft and easily scratched. Bare gold provides over 96.5% average reflectance from 0.70 to 2.0 µm.

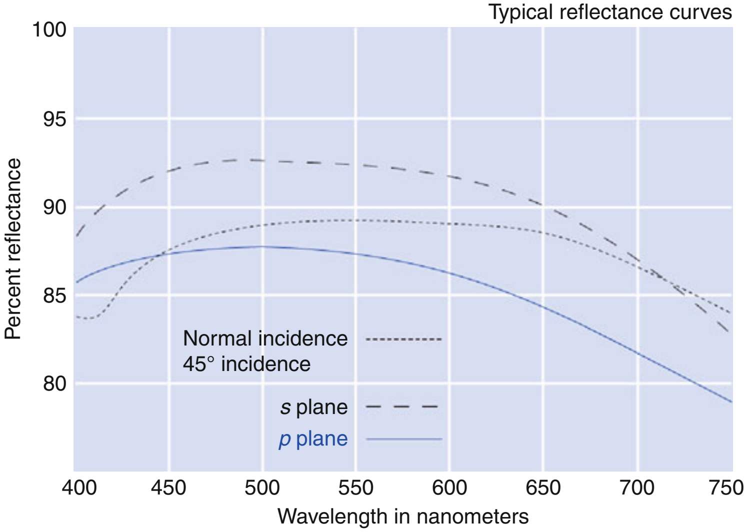


Fig. 5.42
Reflectance characteristic for a first-surface mirror with a protected aluminium coating

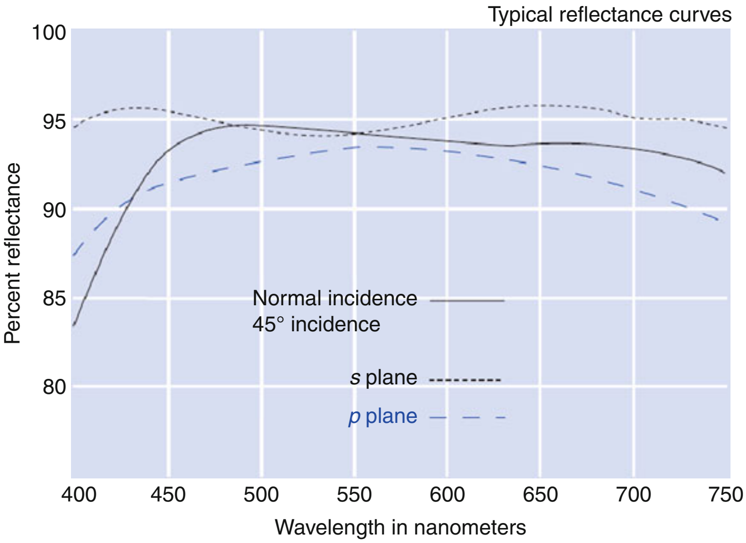


Fig. 5.43

Reflectance characteristic for a first-surface mirror with an enhanced aluminium coating

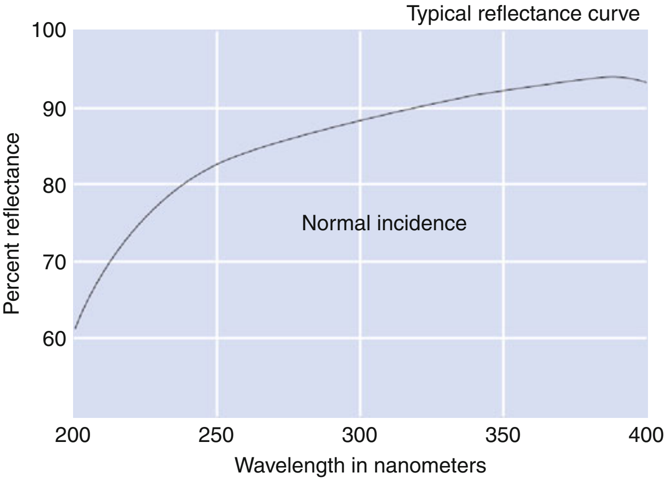


Fig. 5.44

Reflectance characteristic for a first-surface mirror with an enhanced aluminium coating. Notice that this mirror is operating in the ultra-violet region

Protected gold (➋ *Fig. 5.47*) This combines the natural spectral performance of gold with the protection of a durable dielectric overcoat, which improves abrasion resistance. It provides 95.5% average reflectance from 0.65 to 1.70 µm, and over 98% average reflectance from 2 to 16 µm.

MAXBRIte™ (➋ *Fig. 5.48*. MAXBRIte is a product of CVI Melles Griot, Inc.) These coatings provide high reflectance over a broad range of wavelengths. Most of the ultraviolet,

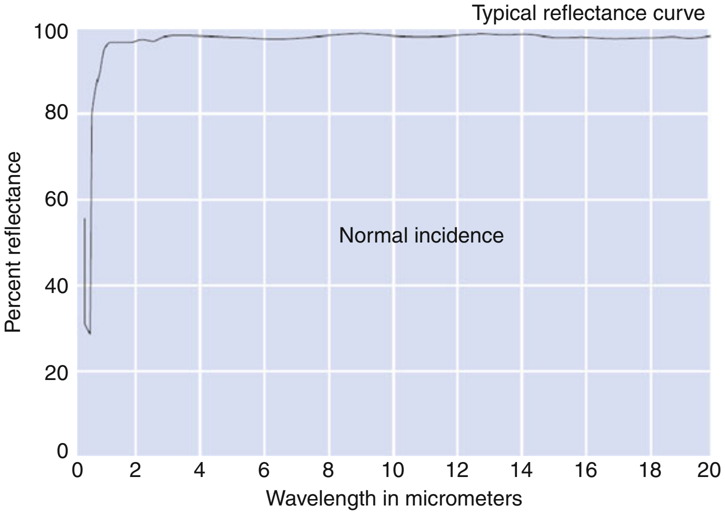


Fig. 5.45
Reflectance characteristic for a first-surface mirror with an protected silver coating. Notice that this mirror is operating in the infra-red region

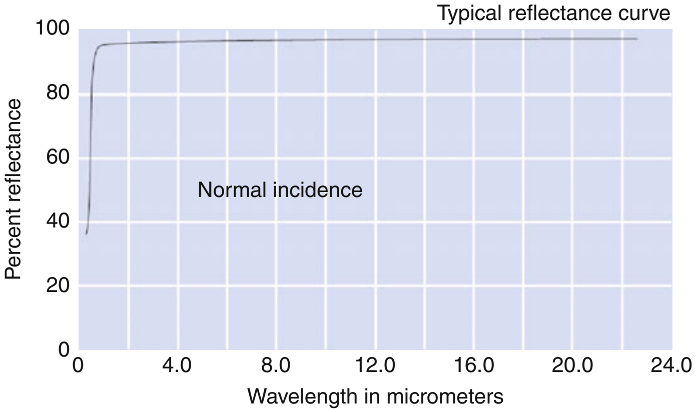


Fig. 5.46
Reflectance characteristic for a first-surface mirror with a bare gold coating. Notice that this mirror is operating in the infra-red region

visible, and near-infrared laser wavelengths are covered by one of the four MAXBRIte coating ranges. MAXBRIte coatings operate for angles of incidence as high as 45°. A minimum of 98% reflectance is achieved over a wide wavelength range at angles of incidence from 0° to 45°.

5.5.4 Substrate Materials

There are several different substrate materials available for mirrors. The main factors affecting choice are thermal expansion coefficient and cost. Mirrors can become temporarily distorted,

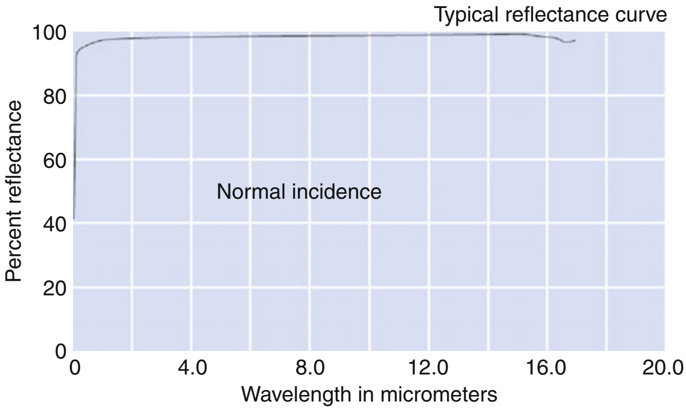


Fig. 5.47

Reflectance characteristic for a first-surface mirror with a protected gold coating. Notice that this mirror is operating in the infra-red region

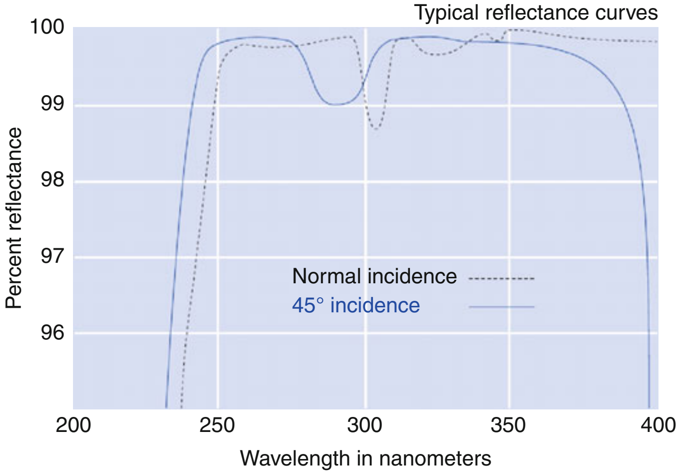


Fig. 5.48

Reflectance characteristic for a first-surface mirror with a MAXBRIte™ coating

or even permanently damaged, when subjected to wide temperature variations, particularly local thermal shock. For applications requiring high precision, low-expansion materials are a key requirement for a successful optical system.

- Optical crown glass is typically used in non-imaging applications, including light gathering or conventional beam-manipulation tasks. It has a relatively high coefficient of thermal expansion and is employed when thermal stability is not a critical factor.
- Low-expansion borosilicate glass is better known by its brand name: Pyrex® (Corning Inc). This material is well suited for high-quality first surface mirrors and optical flats but not for demanding light-transmission applications.

- Synthetic fused silica is a popular material for lenses and makes an excellent mirror substrate, as it has a very low coefficient of thermal expansion.
- Zerodur® (CVI Melles Griot, Inc) is a glass-ceramic material and offers high dimensional stability. It is not used in lenses and other transmissive components.

5.5.5 Quality

5.5.5.1 Flatness and Surface Accuracy

For plane mirrors, flatness is specified as *peak-to-valley* variations of surface height. (λ is the working wavelength.)

- 3λ per 25 mm, acceptable for nonimaging applications or noncritical visual optics.
- λ/4, the most commonly used general-purpose flatness specification. It is acceptable for the majority of research and industrial tasks.
- λ/10, suitable for imaging optics requiring low beam distortion.
- λ/20, needed for the most demanding laser-based setups.

5.5.5.2 Cosmetic Surface Quality

Defects that can be seen on the surface of an optical component reduce its cosmetic surface quality and are important, because they can scatter light. Scratches and digs are indentations on the surface that began much bigger and were not completely removed by polishing. Cosmetic surface quality is specified by two numbers. The first of these (*scratch*) indicates the width of scratches on the surface. This measure can be affected by the sharpness of the edges of the scratch and any unpolished area around it. The second number (*dig*) refers to the diameter of pits in the surface. These two parameters are not dependent on how accurately the surface matches the desired shape: flat, sphere, ellipse, paraboloid, etc. Surface accuracy leads to aberrations, while surface quality factors such as scratch and dig will introduce optical noise in the form of light scattered into the system.

- 80–50 is specified for applications typically not involving lasers, or situations where beam cross section is spread over a wide area.
- 60–40 is used for most research and scientific applications in the visible and infrared.
- 40–20 is appropriate where scattered light can begin to affect system performance, especially with small beam cross sections.
- 10–5 is reserved for high-power laser applications, where the presence of defects can lead to component damage.

5.5.5.3 Other Defects

Within a glass component, such as a lens or mirror, there may be various inclusions, such as air bubbles, striae (local variations in the index of refraction of the material), or physical inclusions, such as dirt and sand (a component of glass). Striae will appear as streaks within the glass that cause local distortion of an image. Minor local irregularities are often tolerated in window glass, domestic mirrors and other mass-produced glass products but can be serious in optical systems.

5.5.5.4 Surface Cleanliness

However, precisely it is made, no optical system will work properly if any one of its working surfaces is dirty. For this reason, it is common practice to enclose the optical components and camera within a sealed protective box (➋ Sect. 5.12. Also see Lighting Viewing Method 93 in ➋ Chap. 40). Light enters this enclosure via an optical window, which must be kept clean, either by regular manual maintenance or by some automated cleaning machinery. This may take the form of an air-jet purge, or washers/wipers like those fitted to automobiles. All optical surfaces on the light source must also be kept clean too. This includes the lamps and their optical diffusers/reflectors.

5.5.5.5 Shape and Dimensions

Optical quality plane mirrors are available in a wide range of sizes (5–200 μm), in both square and round formats. Very much larger, low precision mirrors can be made, using aluminised mylar film, or plastic sheeting (see Lighting Viewing Method 111 in ➋ Chap. 40).

5.5.6 Concave and Convex Spherical Mirrors [2]

Simple ray diagrams can be used to determine the magnification, image location, orientation (upright or inverted) and type of image (real or virtual) formed by concave and convex mirrors (➋ *Figs. 5.49* and ➋ *5.50* respectively). These diagrams provide useful qualitative information about object-image relationships, but do not provide precise quantitative information. To obtain this, the equations given below are used. These are applicable to both concave and convex mirrors, provided we observe the following sign convention:

Focal length, f
 Positive for a concave mirror
 Negative for a convex mirror
Object distance, S_1
 Positive (object is in front of the mirror)
Image distance, S_2
 Positive, real image, in front of the mirror
 Negative, virtual image, located behind the mirror
Object height, h_1
 Positive
Image height, h_2
 Positive if the image is upright (same orientation as the object)
 Negative if the image inverted compared to the object

Mirror Equation

$$1/f = 1/S_1 + 1/S_2 \tag{5.26}$$

Magnification Equation

$$M = S_2/S_1 = h_2/h_1 \tag{5.27}$$

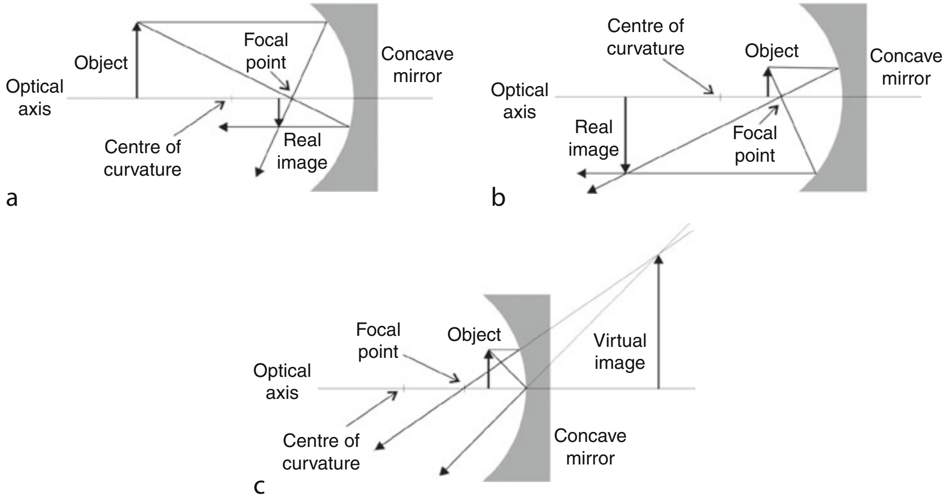


Fig. 5.49

Image formation by a concave mirror. (a) Remote object. When the object is at the centre of curvature, the image is real and inverted. (b) Object between the centre of curvature and the focal point. (c) Object closer than the focal point

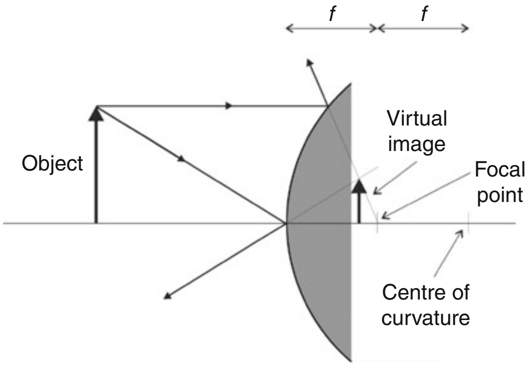


Fig. 5.50

Image formation by a convex mirror. The image is always virtual, upright and smaller than the object

These two equations can be combined to yield information about the image distance and image height if the object distance, object height, and focal length are known.

Using a spherical mirror with a wide aperture always produces some spherical aberration, resulting in blurred images. Incident rays travelling parallel to but a long way from the optical axis of a concave mirror are focussed closer to the reflecting surface than those near the optical axis (➋ *Fig. 5.51*). As with lenses, restricting the aperture reduces the aberration and improves focussing. Spherical aberration can be reduced by using a mirror with a parabolic reflecting surface. This creates sharper, clearer images than those produced by a spherical mirror. First-surface mirrors do not introduce any chromatic aberration.

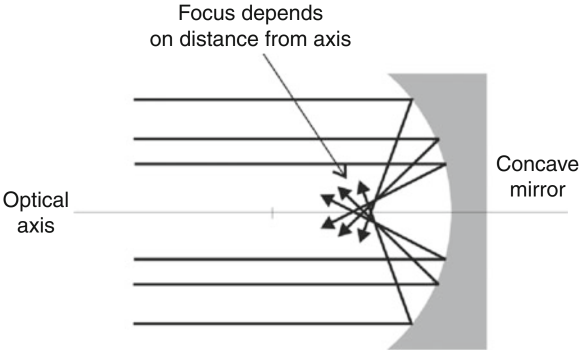


Fig. 5.51
Spherical aberration in a concave mirror. Rays that are parallel to but a long way from the optical axis are focussed closer than those that are near to it. The effect can be reduced by placing a small aperture in front of the mirror

5.5.7 Anamorphic Mirrors

An anamorphic mirror is a device that distorts the image geometry. While the curved mirrors discussed above are often made with spherical reflecting surfaces, only a small proportion of the total area of the sphere is actually used. As a result, the image suffers only a small degree of geometric warping. On the other hand, a full spherical mirror introduces quite severe distortion. Many forms of anamorphic mirror are available and have a variety of specialised uses:

- A cylindrical mirror can be used to adjust the aspect ratio of an image.
- A conical mirror can provide an all-round view of (part of) the internal surface a pipe.
- A complete sphere can provide an all-round view of a room. (The mirrors discussed earlier in this section are formed from small sections of a sphere and do not introduce much geometric distortion. A full sphere reflector does.)
- An elliptical mirror projects all of the light passing through one focus of the ellipse onto its other focus.
- A paraboidal mirror focuses a parallel beam onto a single point. Conversely, it projects all light passing through the focus of the parabola to create a parallel beam
- A concave toroidal mirror can be used to focus a narrow ring of light around (i.e., almost 180°) a cylindrical surface.

It is tempting to dismiss anamorphic mirrors as expensive playthings that serve no useful purpose. This is not the case! Using modern production methods, they are not particularly expensive to manufacture, even on a bespoke basis. First, a high quality master is made, possibly by employing computer-controlled machining methods. This master must then be polished so that its surface is smooth and free of pits, gouges, scratches and protrusions. A mirror surface is then created on it, using electrolytic deposition of a reflecting film, such as nickel. This is, in turn, coated electrolytically, with a highly reflective, hard and durable film. Rhodium reflects well throughout the ultraviolet, visible, and infrared wavebands. It also resists tarnish and oxidation, even at elevated temperatures (i.e., over 200°). Replicated reflectors made in this fashion are comparable in performance to conventional ground and polished

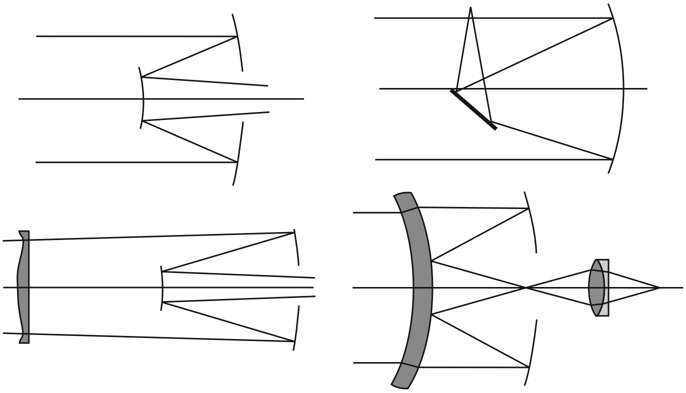


Fig. 5.52

Mirror-based optical systems. (a) Schwarzchild microscope/Cassegrain telescope. (The Gregory telescope is similar but uses a concave secondary mirror.) (b) Newton telescope. (c) Schmidt-Cassegrain telescope. (d) Cassegrain-Matsutov telescope

rhodium-coated mirrors. If the reader still remains unconvinced, remember that the cost of a vision system may actually be reduced by employing one relatively expensive optical component that reduces digital image processing costs/complexity!

Very large anamorphic mirrors (several metres across) can be made by stretching metallised mylar film over a box with a partial vacuum inside and numerous fine holes over its shaped surface. The taut film stretches to form a smooth surface that acts as a tolerably good reflector (Lighting Viewing Method 111 in Chap. 40).

Further examples of anamorphic mirrors are part of our everyday experience; they can be found on silver hollow-ware, brass musical instruments, chrome finish on automobile bumpers (fenders), kitchen equipment (e.g., chrome-plated kettles) and of course, in the fair-ground "Hall of Mirrors."

Photographers have long been familiar with the term *Catadioptric Lens*. This is a combination of one, or more, curved mirrors and a lens assembly. (*Figure 5.52* shows four typical designs for a catadioptric telescope.) It is possible to build a short, light-weight catadioptric device that has a very long focal length. While catadioptric systems use both lenses and mirrors for image formation, *catoptric* systems which use only mirrors. The term *dioptric* is used when referring to optical systems that use only lenses.

5.5.7.1 Spherical Mirror ("Witch Ball")

See *Figs. 5.53* and *5.54*. Prior to the introduction of CCTV surveillance, it was common to find hemi-spherical mirrors in shops, allowing the shop-keeper to monitor suspected thieves. Mirrors of this type are often placed at hazardous junctions on minor country roads to allow drivers to see oncoming traffic. By placing an hemi-spherical mirror above a camera looking upwards, it is possible to obtain an all-round (360°) view of a room. Mobile robots often use this arrangement, enabling them to navigate around obstacles. The geometric

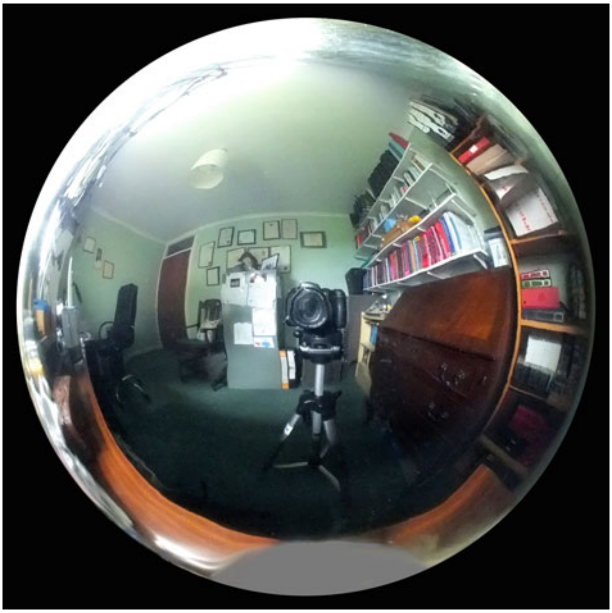


Fig. 5.53
Image warping using a spherical mirror ("Witch Ball")

distortion produced by a spherical mirror is evident in *Fig. 5.53* (Look at the shape of the reflections of the brown table-top and shelving on the right.) Notice that it is possible to locate the light sources: the window and the electric light. This provides the basis for a useful way of characterising the positions and shapes of light source(s) within an Automated Visual Inspection cell [3]. In order to do so, it is necessary to translate the Cartesian address of each point in the image detected by the camera into spherical coordinates, bearing and azimuth. The formulae for doing this are well known [4].

5.5.7.2 Cylindrical Mirrors

Cylindrical mirrors can magnify/minify one axis of an image, while leaving the orthogonal axis unaltered. Both convex and concave versions are available. The cross-section may be circular, elliptical, parabolic, hyperbolic, or polygonal. Within the plane of maximum curvature (YZ in *Fig. 5.55*), a cylindrical mirror with a circular cross-section behaves like a standard curved mirror; Eqs. 5.26 and 5.27 apply close to the X-axis. Within the orthogonal plane (XZ), the curvature is zero and it behaves like a plane mirror.

An example of the warping effect of a convex cylindrical mirror (circular cross-section) is shown in *Fig. 5.56*. Also see [5]. These mirrors were popular in Scotland after the Jacobite Rebellion, when possession of an image of Charles Stewart was illegal. However, his ingenious supporters realised that it was possible to circumvent the law by keeping a warped image of him. When this was viewed via a cylindrical mirror, an undistorted image of the Scottish hero/outlaw appeared [6]. A convex cylindrical mirror would be useful on a mobile robot, to

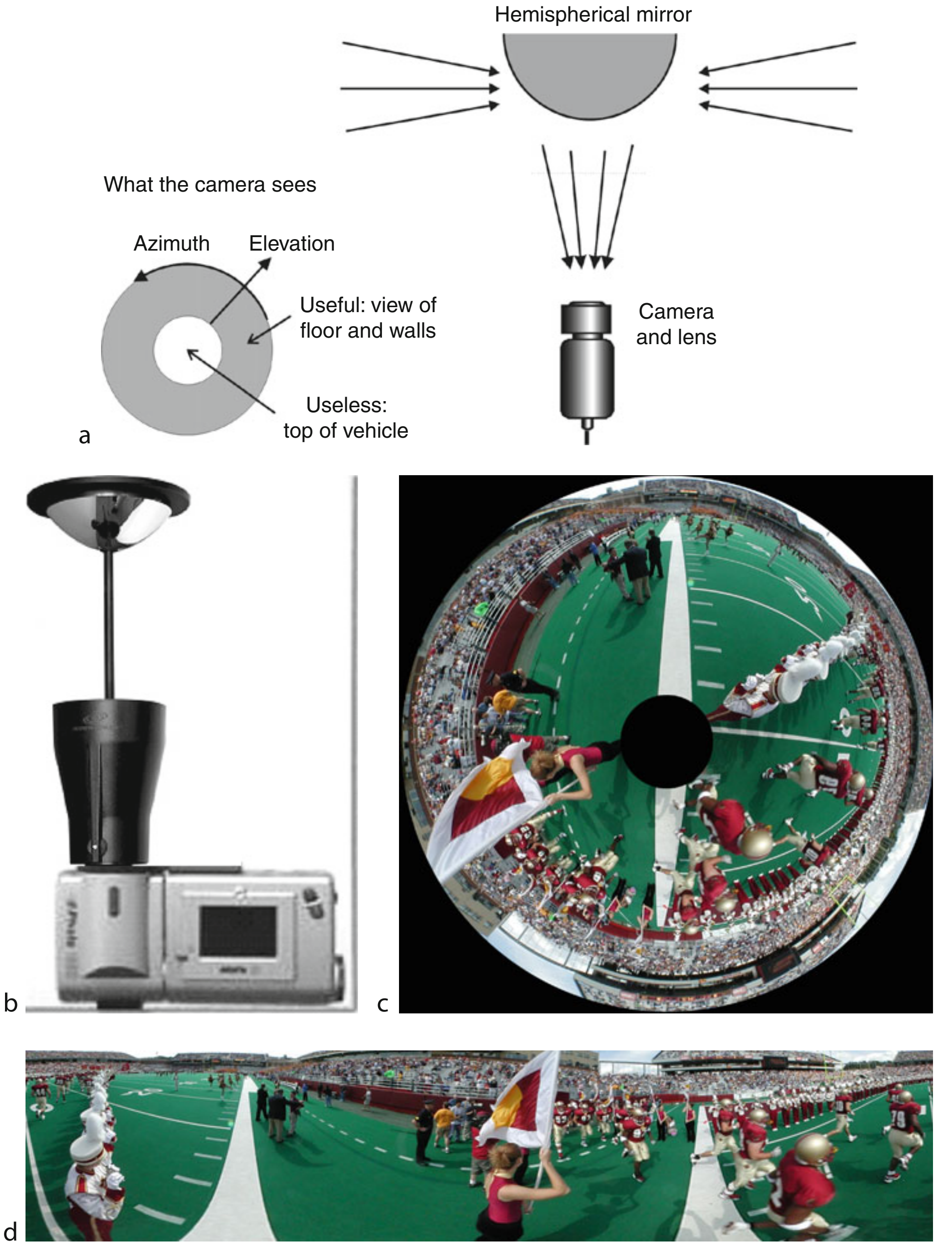


Fig. 5.54

An hemispherical or parabolic mirror can be mounted above a camera to provides a 360° view. Other mirror types (hyperbolic and conical mirrors) can used instead, depending on the range of azimuthal angles to be viewed. (a) Arrangement of the optical system. (b) ''Oneshot'' camera fitted with a parabolic mirror. (c) Image obtained from the ''oneshot'' camera. (d) Software was used to warp the image (Source: http://www1.cs.columbia.edu/CAVE/projects/cat_cam_360/, reproduced with permission. Also see [7] and [9])

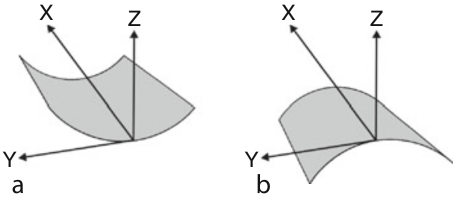


Fig. 5.55
Coordinate axes for cylindrical mirrors. (a) Concave. (b) Convex

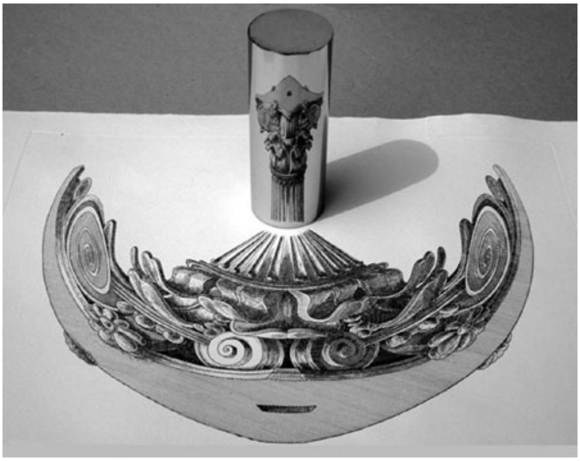


Fig. 5.56
Anamorphic image transformation using a cylindrical mirror. (Image created by Istvan Orosz. See http://en.wikipedia.org/wiki/Anamorphosis_Free_licence)

obtain a wide-angle view of the scene ahead, and could also be used for surveillance of a long thin scene, such as a conveyor belt, walk-way or staircase. Concave cylindrical mirrors with a parabolic cross-section are used in laser scanners. Here, they have two possible uses: ensuring that the projected beam is always paraxial and for gathering light. A concave cylindrical mirror could be used for focussing an elongated linear light source (e.g., a long fluorescent tube) to form a narrow line that could, for example be projected across conveyor belt. The field of view of a line-scan camera can also be broadened using a mirror of this type. Very large cylindrical mirrors are relatively easy to make. It should therefore be borne in mind that a 2-dimensional imaging system can be constructed with two such mirrors. It is therefore possible to build large telecentric optical systems (➋ Chap. 6) that would be prohibitively expensive, if not impossible, using lenses.

5.5.7.3 Parabolic Mirrors

A set of parallel rays can be focussed to a single point by a parabolic mirror (➋ *Fig. 5.57a*). For this reason, a parabolic mirror is often used as the primary (objective) element in an

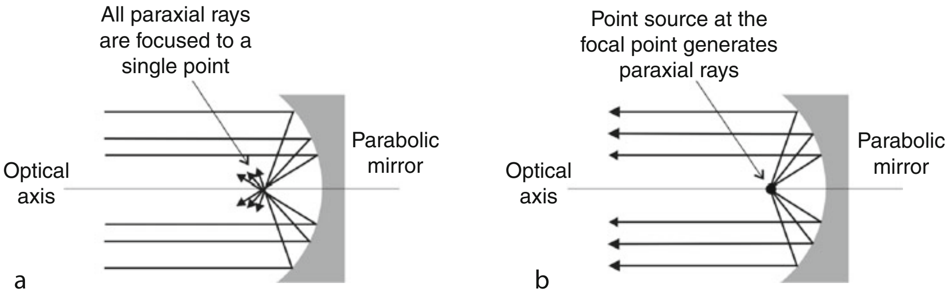

All paraxial rays are focused to a single point
Parabolic mirror
Optical axis
a
Point source at the focal point generates paraxial rays
Parabolic mirror
Optical axis
b

Fig. 5.57
Parabolic mirror. (a) Paraxial rays are focussed to a single point. (b) A point source placed at the focal point generates a parallel beam

astronomical telescope. Conversely, a small light source placed at the focus of a parabolic mirror will form a beam whose rays are nearly parallel (➋ *Fig. 5.57b*). It is often much easier/ cheaper to generate a collimated beam of light with a single paraboidal mirror than with an array of lenses. This arrangement is particularly useful if the scene to be illuminated is far away, or if it is necessary to project a spot of light with a well-defined diameter, without illuminating its neighbourhood. Using a parabolic mirror, it is possible to illuminate the bottom of a deep well, without placing any optical or lighting equipment inside it.

By placing a small camera with a wide field of view at the focal point of a concave parabolic mirror, it is possible to make a telecentric imaging system. Notice however, that the camera "sees" a small image of itself.

Since it is relatively easy to manufacture a parabolic mirror, it is likely to be much cheaper than a lens assembly for any task involving the inspection of objects bigger than about 100 mm. Making a large objective lens can be very expensive.

5.5.7.4 Elliptical Mirrors

Elliptical reflectors have various applications in condensing and illumination systems. A single concave elliptical mirror can replace a complete multi-element condenser system. Elliptical mirrors possess one very valuable property: any light passing through one of the focal points of the ellipse (F_1, ➋ *Fig. 5.58*) will also pass through the other focal point (F_2). Hence, by placing a small light source at focal point F_1, it is possible to project a lot of light onto a small area centred on focal point F_2.

5.5.7.5 Conical Mirrors

A conical mirror was used for inspecting the internal surfaces of pipes as early as 1924 [10]. Also see [11] (➋ *Fig. 5.59*). Conical mirrors have also been used to provide a mobile robot with a panoramic (360°) view of it surroundings. (Cylindrical, spherical, hyperbolic and toroidal mirrors have also been used for this purpose. [12]).

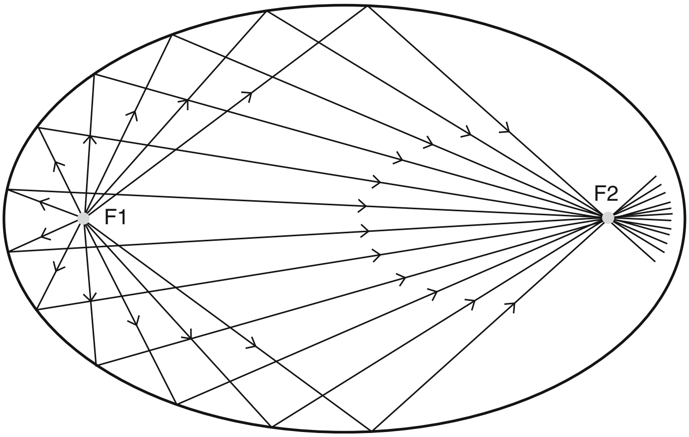


Fig. 5.58
Elliptical mirror. All rays emanating from focus F1 pass through F2

5.5.7.6 Toroidal Mirrors

The flared end of a trumpet indicates some of the possibilities for image formation provided by toroidal mirrors. Since it is rotationally symmetrical, a toroidal mirror is quite easy to form and polish. Like cylindrical mirrors, toroidal mirrors can be made with a variety of cross sections: circular, elliptical, parabolic, hyperbolic or polygonal. Again, both concave and convex versions are possible. A toroidal mirror can be either converging (i.e., directing light towards the central axis of symmetry) or diverging (directing light away from it). Hence it is possible to illuminate/view a rod-like object placed at the centre of the torus, or the inside of a pipe. Convex toroidal mirrors with various cross sections have been used to provide all-round viewing for mobile robots: the camera is placed coaxially with the mirror and "looks" upwards. A concave toroidal mirror can focus a hollow cylinder of light to form a bright ring of light all round a cylinder. The main point to note is that toroidal mirrors offer numerous possibilities for both image acquisition and light projection; the vision system designer simply needs to exercise his/her imagination a little!

5.6 Filters

Optical colour filters can provide significant improvement in contrast. For example, in *Fig. 5.60* the red and blue regions are more easily distinguished from each other if either a long-pass or short-pass filter is inserted in the optical path. In this case, the filter was placed in the illumination path-way, as this avoids distorting the image.

Two types of colour filter are in widespread use. These employ either the bulk properties of coloured materials (e.g., glass or gelatine), or interference effects in a multi-layer dielectric film. Interference filters enable rapid and accurate measurement of the optical energy within a well-defined spectral band. Narrow-band filters provide a means of isolating, or blocking, wave-bands just a few nanometres wide, sometimes even less. This is achieved without the use of

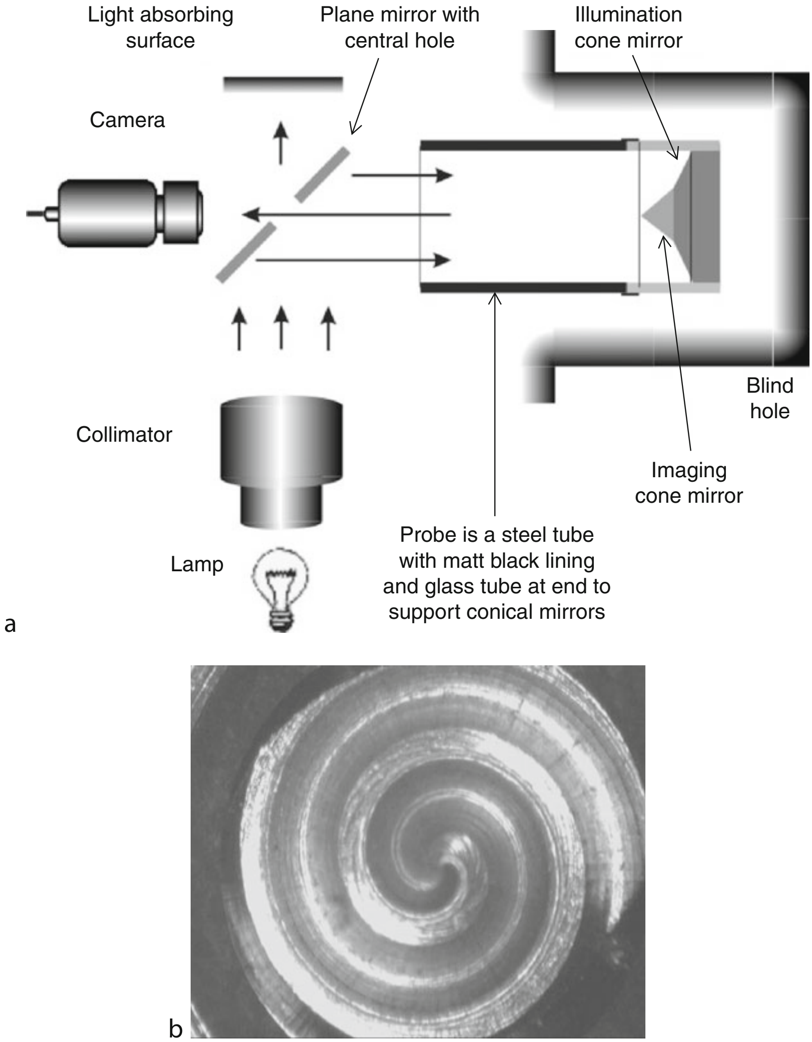

Light absorbing surface
Plane mirror with central hole
Illumination cone mirror
Camera
Blind hole
Collimator
Imaging cone mirror
Probe is a steel tube with matt black lining and glass tube at end to support conical mirrors
Lamp
a
b

Fig. 5.59
Bore inspection using a conical mirror. (a) Possible optical arrangement, using separate illumination and viewing cones. (b) Internal screw thread

prisms or gratings. For example, a single line in the emission spectrum of a flame, a sodium or mercury lamp can be monitored without confusion from other nearby lines. The light from a diode laser line-projector (used for 3D shape measurement) can be detected even when the scene is in bright sunlight. Coloured-glass and gelatine filters are incapable of such fine discrimination as this. Interference filters are based on multi-layer (thin-film) devices. They are often termed “dielectric devices” but may contain metallic layers as well. Indeed, broad-band filters almost always contain a metallic layer in their spacers. Interference filters come in three types:

(a) Band-pass filters transmit over a defined wave-band while rejecting longer and shorter wavelengths.

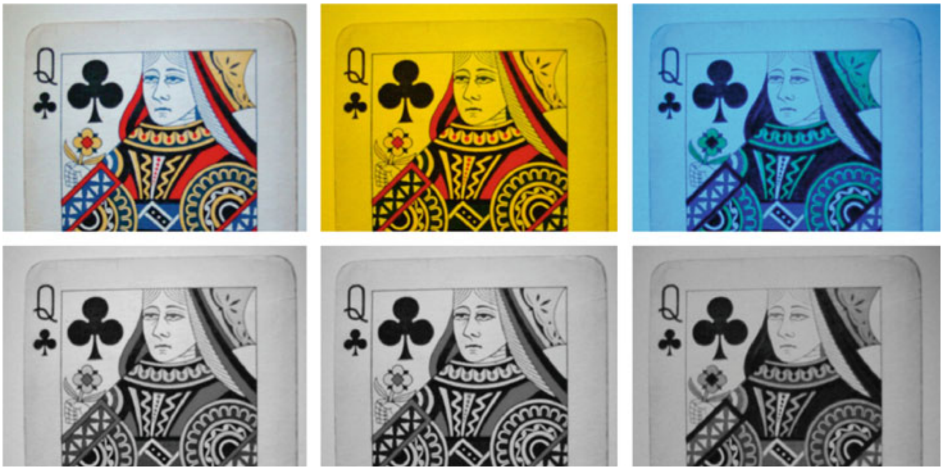


Fig. 5.60

Using optical filters to separate colours. (a) Original image. (b) Long-pass interference filter placed in the illumination path. Cut-off 0.550 µm. (c) Short-pass interference filter placed in the illumination path. Cut-off 0.550 µm. (d) Intensity component of (a). (e) Intensity component of (b). (f) Intensity component of (c)

(b) Band-stop block light within a defined wave-band while passing longer and shorter wavelengths.
(c) Long-pass transmit light with wavelengths *above* a certain critical value and block others.
(d) Short-pass transmit light with wavelengths *below* a certain critical value and block others.

5.6.1 Interference Filters

Narrow-band interference filters operate on the same principle as the Fabry-Perot interferometer; they both rely on the interference of multiple reflected beams (➋ *Fig. 5.61*). Incident light undergoes multiple reflections. When there is no phase difference between emerging wavefronts, constructive interference occurs; each transmitted wave-front has undergone an even number of reflections. The air gap of the Fabry-Perot interferometer is replaced by a thin layer of dielectric material with a half-wave optical thickness corresponding to the wavelength of the desired transmission peak. By analogy with interferometers, the simplest band-pass interference filters are sometimes called cavities. Two, or more, band-pass filters can be deposited, to form a multiple-cavity filter (➋ *Fig. 5.61*). Increasing the number of cavities has a significant effect on the shape of the passband. The resulting overall pass-band transmittance is given approximately by the product of the pass-bands of individual cavities. By using multiple-cavity filters, it is possible to obtain sharp cut-off, improved near-band rejection, and “square” pass-band peaks.

5.6.2 Points to Note

The spectral transmission characteristics of several different types of interference filters are shown in ➋ *Figs. 5.62–5.69*.

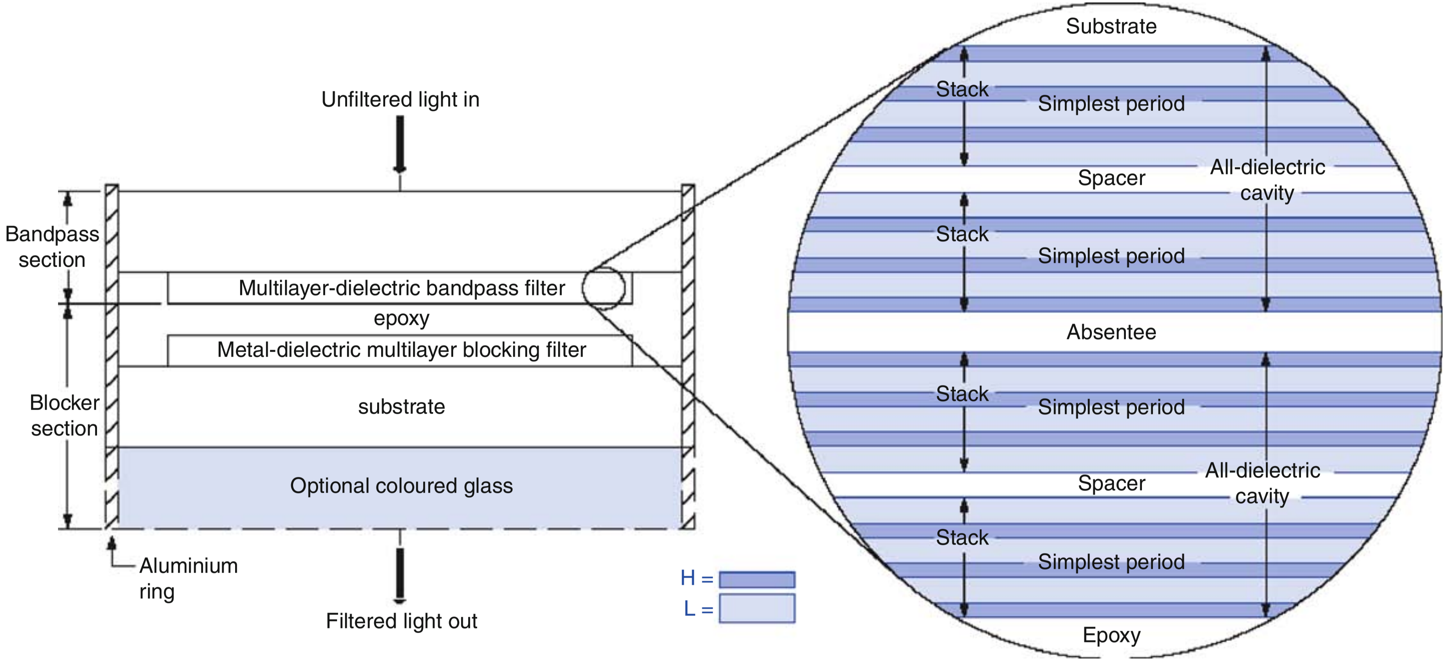


Fig. 5.61
Construction of a band-pass interference filter

5.6.2.1 Dichroic Filters/Mirrors

A dichroic filter, or dichroic mirror, passes light within one part of the spectrum, while reflecting others. Dichroic mirrors are usually characterized by the colour(s) that they reflect, rather than those that they transmit. These devices operate using interference in a multi-layer thin film and hence can have sharp cut-off characteristics. Many quartz-halogen lamps have an integral dichroic reflector. They were incorporated into slide projectors to avoid melting the slides. They are now widely used in domestic "mood" lighting and to illuminate product and poster displays. So-called *hot mirrors* reflect wavelengths in the infra-red wave-band and transmit visible light. (*Cold mirrors* reflect VIS and transmit IR.) Since they absorb very little energy, a hot mirror does not become as hot as a conventional IR-absorbing filter. Dichroic filters are also widely used in multi-sensor colour cameras to separate the RGB channels optically. Dichroic and Trichroic filters in the form of solid-glass prisms are discussed in ❯ Sect. 5.8.

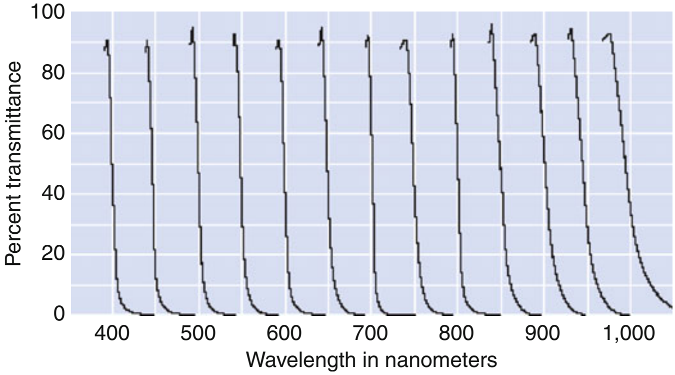


Fig. 5.62
Typical short-pass filter characteristics

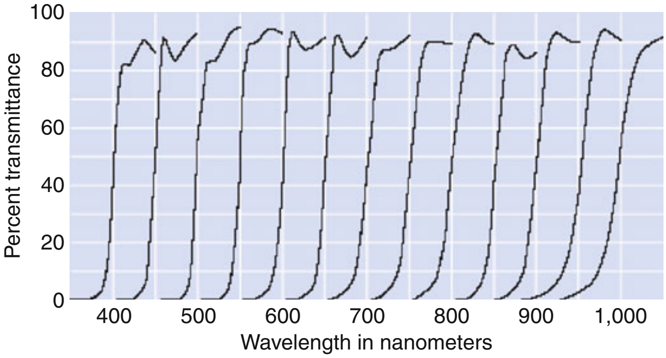


Fig. 5.63
Typical long-pass filter characteristics

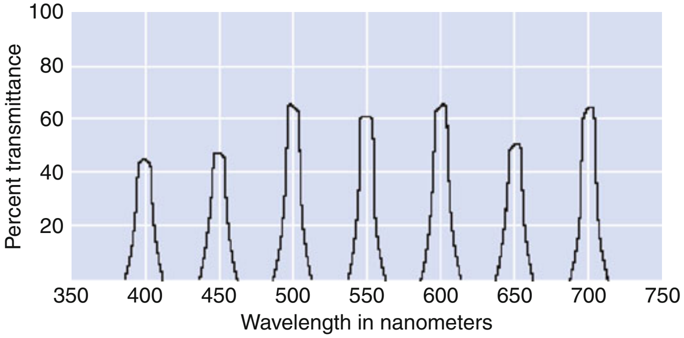


Fig. 5.64
Typical narrow band-pass filter characteristics (10 nm wide pass band)

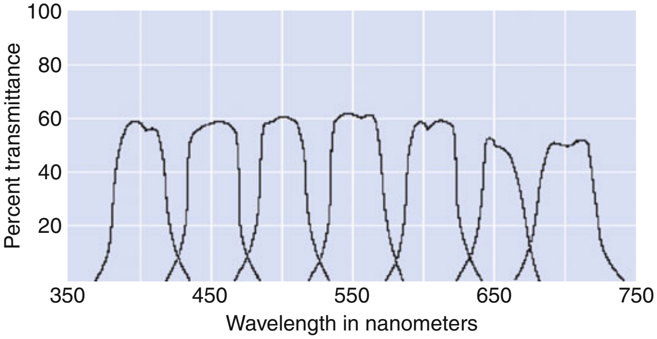


Fig. 5.65
Typical narrow band-pass filter characteristics (40 nm wide pass band)

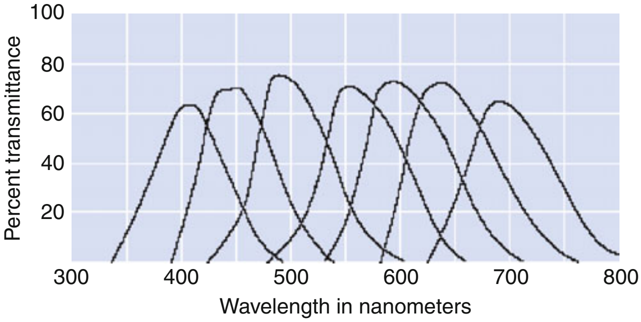


Fig. 5.66
Typical narrow band-pass filter characteristics (80 nm wide pass band)

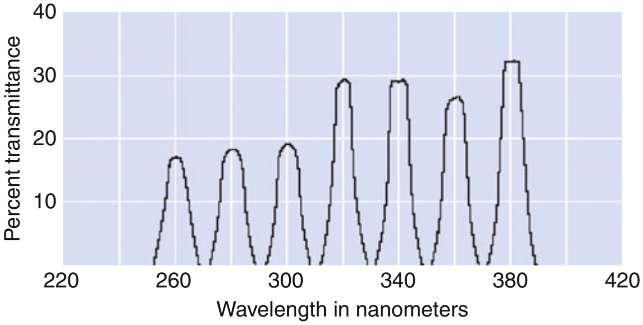


Fig. 5.67
Typical UV band-pass filter characteristics

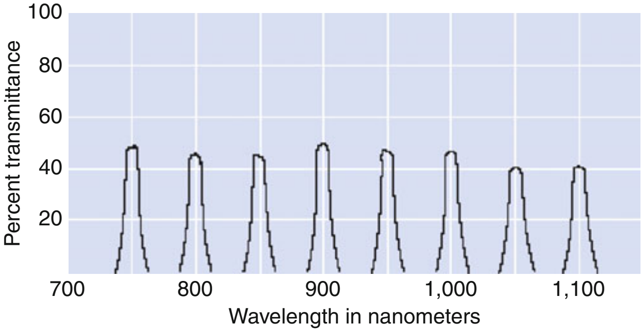


Fig. 5.68
Typical IR band-pass filter characteristics

5.6.2.2 Tilting a Filter Alters Its Cut-Off Wave-Length

A useful characteristic of interference filters is that transmittance spectrum shifts to shorter wavelengths as they are tilted from normal to oblique incidence. This applies to long-pass, short-pass and band-pass filters. (As the angle of tilt is increased in filters constructed with metallic layers, the transmittance peak splits into two orthogonally polarised peaks which shift to shorter wavelengths at different rates. However, filters made with all-dielectric multilayers do not suffer from this disadvantage.) This shift in the cut-off wave-length is very useful in fine tuning band-pass filters to suit the application requirements.

5.6.2.3 Filter Orientation

A good rule of thumb, especially important if there is a risk of overheating, is that interference filters should be oriented with the shiniest (metallic) and nearly colourless side toward the light source. This will minimize thermal load on the component. Notice, however, that reversing the orientation of a filter will have no effect on its transmittance near, or within, the pass-band.

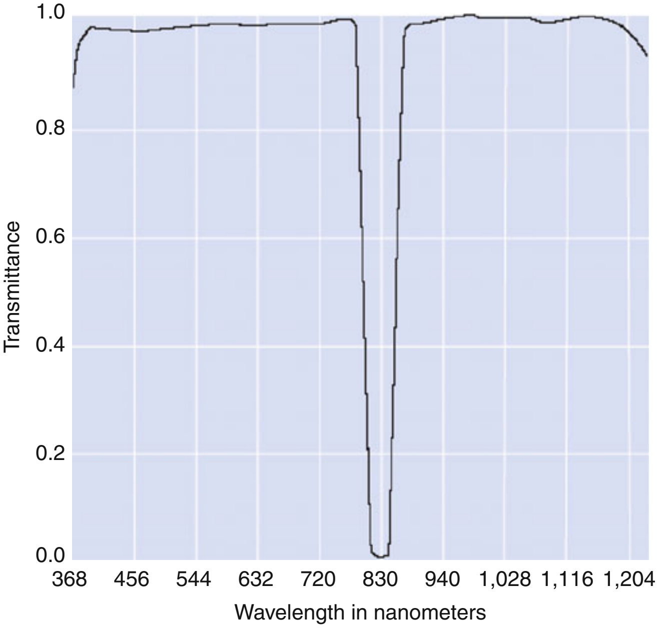


Fig. 5.69
Typical band-stop filter characteristics ("notch filter"). Notch filters are useful for blocking a single wavelength, such as a laser line, while allowing a wide spectral band on both sides of the blocked wavelength to pass with little attenuation. High rejection at the blocked wavelength allows these filters to be used in applications such as laser-induced fluorescence. The wavelength that is blocked can be shifted to a shorter wavelength by tilting the filter so that the angle of incidence is not 0°. Typically, a shift of up to 1% of the nominal wavelength can be achieved with a tilt of 10–15°

5.6.2.4 Temperature Effects and Thermal Shock

The transmittance spectrum of an interference filter is slightly temperature dependent. As temperature increases, all layer thicknesses increase. At the same time, all of their refractive indices change. These effects combine in such a way that the transmittance spectrum shifts slightly to longer wavelengths with increasing temperature. The thermal coefficient is a function of wavelength, as shown in the following table. Interference filters are typically designed for use at or near 20°C. Unless band-pass filters with extremely narrow pass-bands are used at very different temperatures, the transmittance shifts are negligible; standard interference filters can be used at temperatures down to −50°C. Thermal contraction will result in permanent filter damage below this temperature. High-temperature limits depend on filter design: 70°C is a safe and conservative limit for filters of this type, although some can tolerate temperatures up to 125°C. As a general rule, it is unwise to subject interference filters to thermal shock, especially close to the lower limit of −50°C. Temperature change rates should not exceed 5°C/min.

5.6.3 Coloured-Glass Filters

To improve stability and reliability, use high-grade glass (e.g., Schott glass) filters in preference to gelatine filters. The spectral properties of coloured-glass filters are uniform over their entire aperture and are invariant with temperature below 250°C (482°F). Unlike interference filters, their spectral properties are insensitive to the angle of incidence (➲ *Figs. 5.70* and ➲ *5.71*).

At any specific wavelength the external transmittance, *T*, of a coloured-glass filter is given by the approximate formula

$$T = t_1 t_2 T_\mathrm{i} \tag{5.28}$$

where t_1 is the transmittance of the first air-glass interface, t_2 is the transmittance of the second airglass interface, and T_i is the internal transmittance. T_i is proportional to the width of the filter material. This formula neglects possible multiple reflections that may occur between interfaces. The interface transmittances t_1 and t_2 are equal, and their product $t_1 t_2$ is called the correction factor. This is related to the refractive index n of the glass by the following formula

$$t_1 t_2 = 1 - 2\left(\frac{n-1}{n+1}\right)^2 + \left(\frac{n-1}{n+1}\right)^4 \tag{5.29}$$

5.7 Neutral Density Filters (ND Filters)

The purpose of a neutral density filter (ND filter) is to reduce the intensity of the light passing through it. Ideally, it attenuates all wavelengths within the working spectrum equally. Neutral density filters allow greater flexibility of aperture settings when working in extremely bright conditions. (Recall that the depth of field of a lens is dependent on the size of its aperture.) An ND filter allows the aperture to be opened in a bright environment, while keeping the depth of field small. In some situations, having a small depth of field is an advantage; back-ground detail does not produce sharp edges. Neutral density filters are particularly useful when catadioptric devices are to be employed.

5.7.1 Graduated ND Filters

The attenuation varies across the face of a graduated ND filter. This type of device is useful when one region of the scene being viewed is very bright, compared to the remainder.

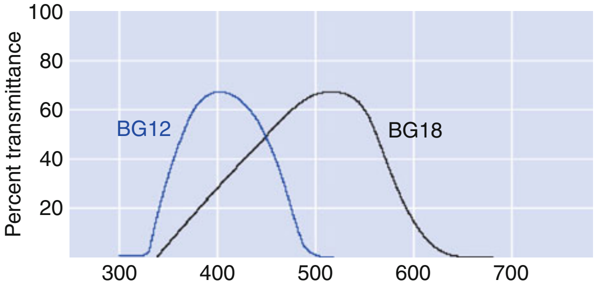


Fig. 5.70
Typical coloured glass filter characteristics (band-pass) for two different types of material

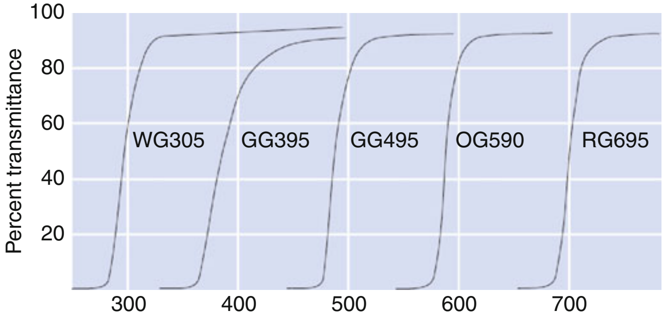


Fig. 5.71
Typical coloured glass filter characteristics (long-pass) for five different types of material

Photographers have long used graduated ND filters when trying to capture sunset images. The attenuation profile may be abrupt (*hard edge ND filter*), or smooth. in which case it may vary linearly, logarithmically, or otherwise non-linearly with position. Using a graduated ND filter effectively allows the camera's dynamic range to be increased. This is useful when the position, or size, of a very bright object is to found relative to a much darker one, using a single camera. Graduated neutral density filters might have two or more step-like transitions between regions of constant attenuation. They are also made with the intensity varying with angular position, or as a function of distance from the central point. Again, the transitions may be smooth or abrupt. Stepped ND filters are useful for checking the linearity of the intensity scale of a camera, since they have well-defined attenuation values.

5.7.2 Metallic ND Filters

Metallic ND filters are made with annealed glass, or optical-quality synthetic fused silica substrates. Vacuum deposition is used to apply a thin film of several special metallic alloys to the substrate. These alloys are chosen to create a spectral-density curve that is flatter over a wider range than the curves of most pure metals. Substrate materials are chosen for homogeneity, transmittance uniformity, finishing characteristics, and (in the case of synthetic fused silica) ultraviolet transmittance. Substrates are polished to minimize light scattering. Metallic ND filters can be used at any wavelength between 0.20 and 2.50 μm (fused silica), or between 0.350 and 2.50 μm (BK7 glass). Their operation depends on absorption in, and reflection from the thin metallic film. When used in high-intensity (laser) beams, ND filters should be oriented with the metallic film facing toward the source to minimize substrate absorption and heating. Alloy films are corrosion resistant and do not age at normal temperatures. Adhesion of alloy films to their substrates is tenacious and unaffected by moisture and most solvents from −73°C to 150°C. Exposure to higher temperatures should be avoided because it causes film oxidation and increased transmittance. These filters are not suitable for use with high-power lasers.

5.7.3 Absorptive ND Filters

Absorptive ND filters provide an alternative to metallic ND filters. The light attenuating characteristics of this type of filter is a function of its material properties and the filter thickness. Since there can be large variations between glass melts, actual thickness and glass material may vary in order to guarantee optical density. These filters are recommended for low-power applications only, because of their absorbing properties.

Despite their inherent simplicity, neutral density filters are not perfect. Like any other thin glass plate, they introduce some aberrations, they scatter light and do not have a perfect spectral response. However, these are minor effects and do not cause serious problems for most applications. More importantly, they do not respond well outside the visible waveband and it may be necessary to add UV- and/or IR-blocking filters.

5.8 Prisms

Prisms are blocks of glass, or some other transparent optical material, with flat polished sides, arranged at precisely defined angles. They are used in four principal ways:

(a) To decompose light into its constituent wavelengths.
(b) To deflect/invert/rotate a light beam.
(c) To split a light beam into two secondary beams, while preserving its image structure.
(d) To separate light rays according to their polarisation.

5.8.1 Aberrations

Prisms introduce aberrations when used with convergent or divergent beams of light. Using prisms with collimated, or nearly collimated light, will reduce aberrations to a minimum. Since light of different wavelengths is dispersed by a prism it may be necessary, in some applications, to use monochromatic or narrow-band lighting.

5.8.2 Total Internal Reflection

Rays that are internally incident upon an air/glass boundary at angles greater than the critical angle are reflected with 100% efficiency, regardless of polarisation. The critical angle is given by θ_{critical} where

$$\theta_{\text{critical}} = \sin^{-1}\left(\mu_{\text{air}}(\lambda) \Big/ \mu_{\text{glass}}(\lambda)\right) \tag{5.30}$$

Note that the refractive index of glass, μ_{glass}, is a function of wavelength. (μ_{air} does not vary significantly, over the visible waveband.) Reflectance decreases rapidly at angles of incidence less than the critical angle. This effect is used in prismatic beam-splitters to select rays with different polarisations. (See ➤ Sect. 5.8.5)

5.8.3 Equilateral Dispersing Prism

Dispersive prisms are able to decompose light into its constituent wavelengths. These are seen by human beings as having different colours. The refractive index of glass differs slightly for different

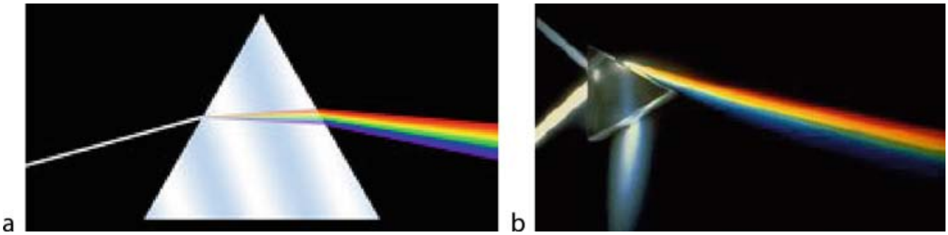


Fig. 5.72
Standard equilateral triangle (i.e., 60°/60°/60°) prism. (a) Idealised schematic. (b) In practice, the incoming light beam is partially reflected at the first and second surfaces, as well as producing a dispersed beam

wavelengths of light (strictly, optical frequencies). Rays with different wavelengths are diverted through different angles; blue light is slowed down more than red light and is therefore diverted more than red light. In his famous experiment, Isaac Newton used a prism to disperse a beam of sun-light, creating a rainbow-like spectrum (➲ *Fig. 5.72*). He then used a second prism to reconstitute white light from a rainbow-like spectrum, thereby allowing him to study the physics of colour. Reflection losses are minimized for unpolarised rays travelling parallel to the base of the prism. This is called the *minimum deviation condition* and occurs when the angles of the entrant and emergent rays, relative to their respective surface normals, are equal.

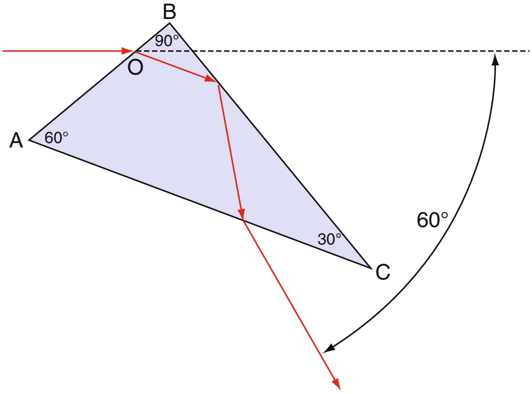
Dispersive prisms are used exclusively for spectral analysis and do not have any direct role in image formation. While a prism can be used for performing spectral analysis, the same function can often be achieved better by a fine grating.

5.8.4 Abbe and Pellin-Broca Prisms

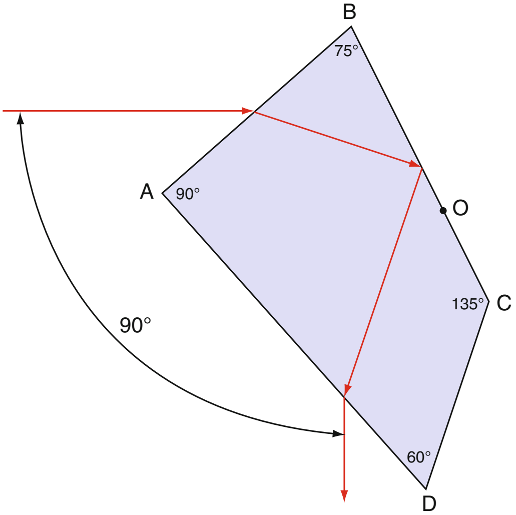
An Abbe prism has angles of 30°, 60° and 90° (➲ *Fig. 5.73*). The light beam undergoes total internal reflection from the hypotenuse face. One particular wavelength of light is deviated by exactly 60°. (This is the minimum possible angular deviation; all other wavelengths are deviated by greater angles.) By rotating the prism about the point O, the wavelength of the emergent ray that is deviated by 60° can be altered. The Pellin-Broca prism is shown in ➲ *Fig. 5.74*.

5.8.5 Dichroic and Trichroic Prisms

A *dichroic* prism splits light into two beams of differing wavelength (colour) and consists of one or more glass prisms with dichroic optical coatings on certain faces. These act as wavelength-sensitive beam-splitters and are often used in camcorders and good quality digital cameras. A *trichroic* prism assembly is a combination of two dichroic prisms and is typically used to decompose an image into three components: red, green and blue. A trichroic prism can be used in reverse to combine three monochromatic beams (e.g., from LEDs) to form a single beam and has potential application in a programmable projector that allows different mixtures of red green and blue light to be created (➲ *Fig. 5.75*).



■ **Fig. 5.73**
Abbe prism



■ **Fig. 5.74**
Pellin-Broca prism. Rotating the prism about point O (BO = 2OC) causes a ray with a different wavelength to emerge with a deviation of 90°

5.8.6 Reflecting Prisms

A right-angle prism can be used to deflect a beam of light by 90°, or to retroreflect it (➲ *Fig. 5.76*). Little light is lost, since total internal reflection occurs at the hypotenuse face. To improve the performance, antireflection coatings should be used on the entrance and exit faces. Two prisms can be used to rotate an image by 180°. If the hypotenuse face is given a metallic (“silvered”) coating, the prism may be used as an external front-surface mirror.

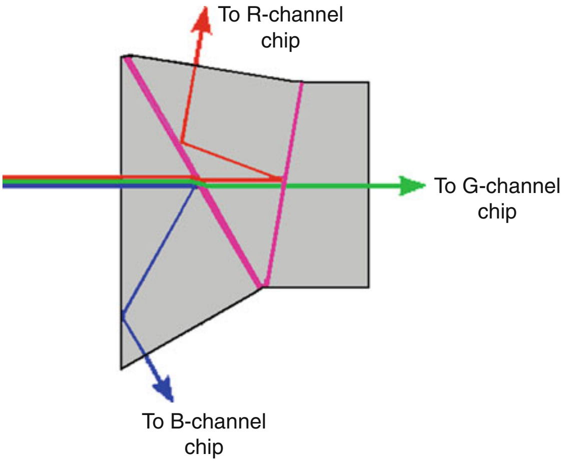


Fig. 5.75
Trichroic prism

5.8.7 Amici Roof Prism

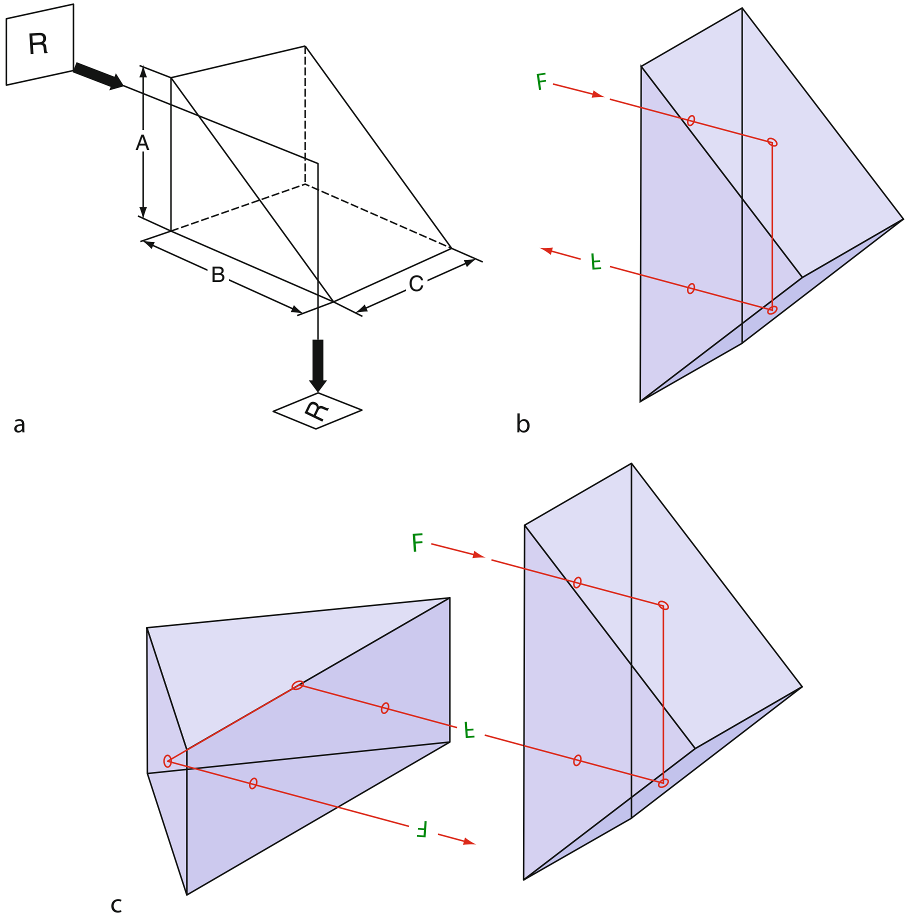
An Amici roof prism is a right-angle prism whose hypotenuse has been replaced by a 90° totally internally reflecting "roof," two faces inclined at 90° to each other (➲ *Fig. 5.77*). This prism deflects the light beam through an angle of 90° and performs left-to-right axis inversion. Amici prisms are available with antireflection coated entrance and exit faces. Any imperfections, such as chips, along the ridge of the roof, spoil the centre-line of the image. This may cause problems in some applications.

5.8.8 Pentaprism and Roof Pentaprism

A pentaprism is a five-sided reflecting prism and is able to deflect a light beam by 90° without inverting it (➲ *Fig. 5.78*). This type of prism does not rely on total internal reflection. Hence, the two reflecting faces must be coated, to provide mirror surfaces. The two transmitting faces are often coated to reduce spurious reflections. The fifth face is not used optically. In the roof pentaprism one of the reflective faces of a normal pentaprism is replaced by "roof" section (two surfaces angled at a 90° to each other). This inverts the image. However, chips in the "ridge top" of the roof spoil the centre of the image, which may make the roof pentaprism unsuitable for certain applications.

5.8.9 Dove Prism

A Dove prism is used to rotate an image; if a Dove prism is turned through an angle θ, the image is rotated by 2θ. It is a truncated 90°/45°/45° prism. The beam undergoes total internal reflection at the hypotenuse face, which need not be aluminized (➲ *Fig. 5.79*).



◘ **Fig. 5.76**

Right angle prism. (a) A single prism used as an internal mirror. (b) A single prism used as a retroreflector. (c) Two prisms used to rotate an image by 180°

However, the entrance and exit faces may be coated to reduce reflections Dove prisms are normally used with parallel, collimated beams. Since it disperses white light into its spectral components, a Dove prism should be used with monochromatic (or narrow-band) light.

5.8.10 Anamorphic (Beam Expanding) Prisms

Anamorphic prisms are used in pairs to magnify beam size along one axis, while leaving the beam unchanged along the orthogonal axis (➲ *Fig. 5.80*). They can typically achieve magnification ranging from 2× to 6× (or if reversed 0.17–0.5). These prisms

- Minimize reflection losses
- Maintain the polarisation state of the radiated beam

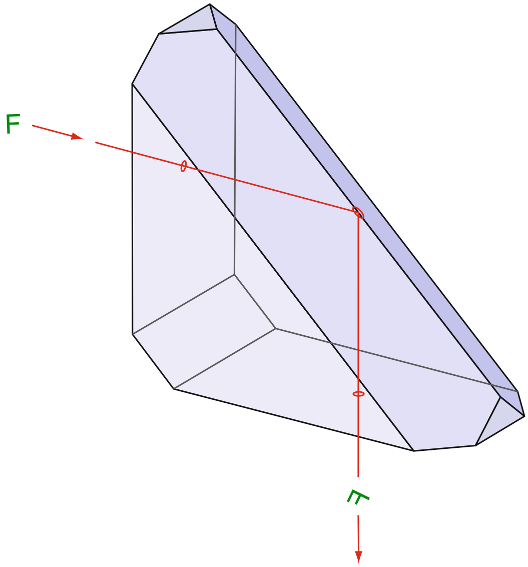


Fig. 5.77
Amici roof prism

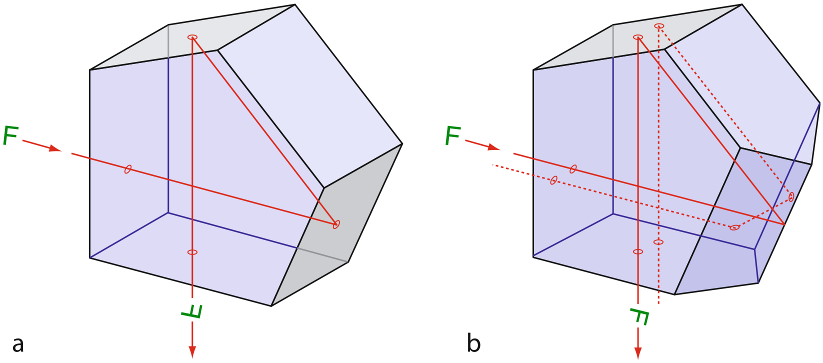


Fig. 5.78
Pentaprism. (a) Standard pentaprism. (b) Roof pentaprism

- Displace the transmitted beam by only a small lateral amount
- Typically deviates the beam by a very small angle (less than 1°)

5.9 Polarisation and Polarisers

Polarising sunglasses provide a clue as to the benefits of using polarisers for Machine Vision. Polarisers include the ability to

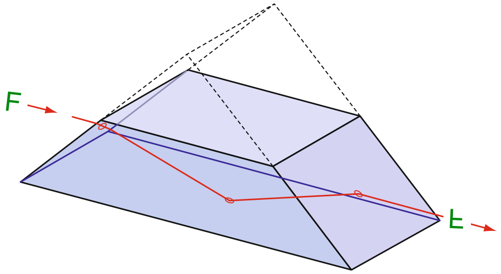


Fig. 5.79
Dove prism

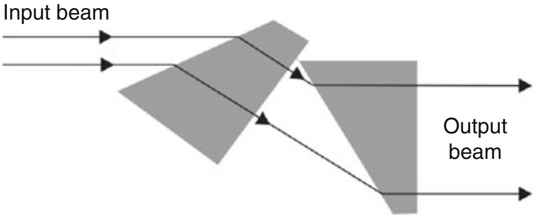


Fig. 5.80
Beam expanding prisms (This can be used in reverse, to reduce the input beam)

(a) Suppress glinting from smooth dielectric materials
(b) Visualise stress patterns in glass, plastics and other clear materials (Recall the stripes that appear on automobile wind-screens when wearing polarising sunglasses, *Fig. 5.81*.)

Several different types of polarisers are in common use and are reviewed below. Their relative merits should be considered carefully before deciding which one to use in an industrial vision system.

5.9.1 Physics of Polarisation

Consider light as a wave: the electric field (**E**) and magnetic field (**H**) are orthogonal to one another and to its direction of travel (*Fig. 5.82*). We can describe polarisation by considering the E-vector alone. If all rays travelling along a given path have their E-vectors aligned in the same direction, the beam is said to be polarised. This can be achieved by passing it through an array of fine parallel conductors. The rays whose E-vectors are parallel to the conductors are blocked, while those whose E-vectors are normal to the array can traverse unhindered. To understand how rays with E-vectors at other orientations behave, we must first decompose the ray into two orthogonal components (*Fig. 5.82c*), parallel and normal and to the conductor array. The former is blocked, while the latter is able to pass through the array unattenuated. As a result, the E-vector of the emergent ray is normal to the conductors but its amplitude is lower than that of the incident ray. What we have described thus far is called *linear polarisation*.

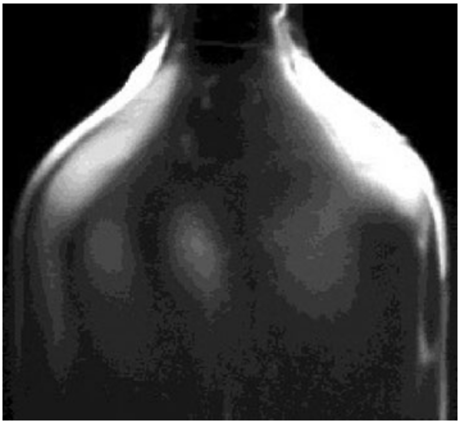


Fig. 5.81

Stress patterns in glass are made visible using two crossed linear polarisers

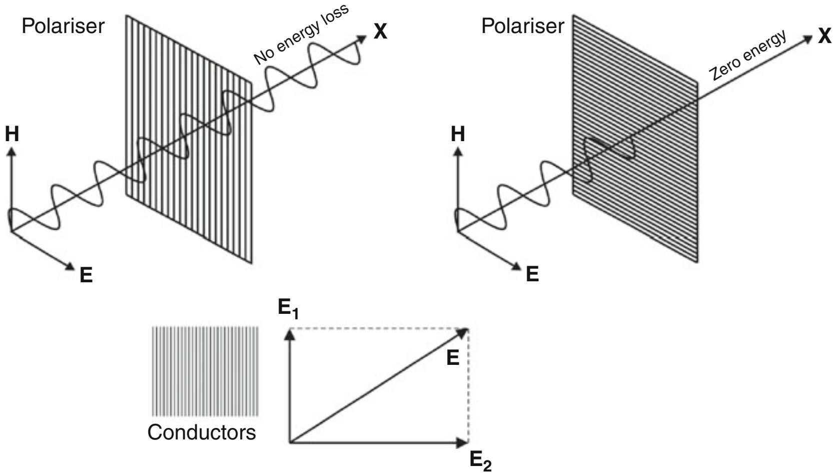


Fig. 5.82

Linearly polarised light. Polarising light using an array of very fine conductors. (a) A wave whose E-vector is normal to the conductors is transmitted unattenuated through the array. (b) A wave whose E-vector is parallel to the conductors is blocked by the array. (c) A wave with E-vector E resolved into two components: E_1 and E_2 where $E = E_1 + E_2$. The polariser blocks E_1 and passes E_2

Let us now consider a slightly more complicated situation (➲ *Fig. 5.83*). Two waves travelling along the same path have orthogonal E-vectors, have the same wavelength but have different phases. Their amplitudes may not be the same. Since their E-vectors are additive, the result is a ray whose E-vector rotates in both time and along the direction of travel. This is called *elliptical polarisation*. If the amplitudes of the two waves are the same, the ray is *circularly polarised*. A circular polariser may be constructed by combining a linear polariser with

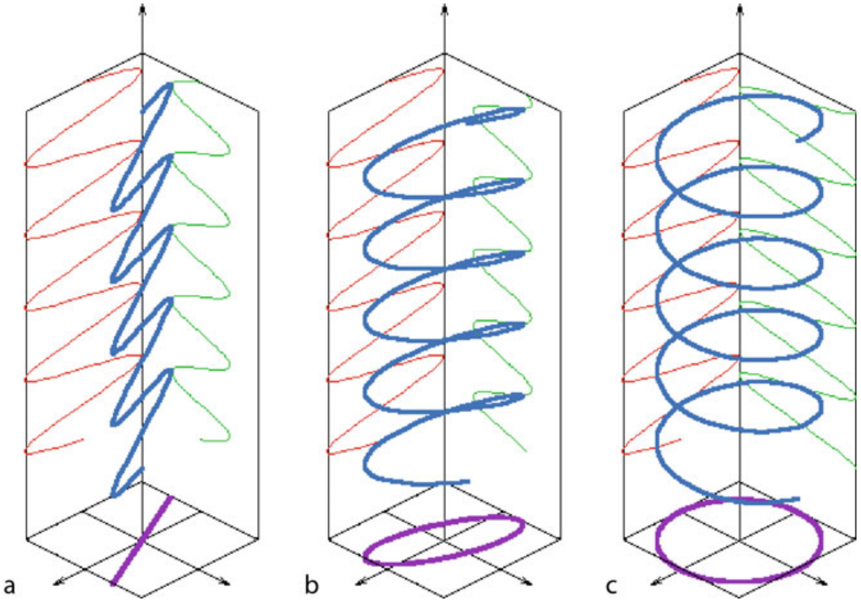


Fig. 5.83
Polarisation states. (a) Linear. Sinusoidal components, projected onto orthogonal axes, are in phase. (b) Elliptical. Sinusoidal components are not in phase and their amplitudes are unequal. (c) Circular. Sinusoidal components are not in phase and their amplitudes are equal

a quarter-wave plate. Circularly polarised light can be used to suppress glinting, while reducing the effect of changing the angle of illumination. At normal incidence, we cannot distinguish between vertical and horizontal polarisation in which case using polarisers will not help us.

Note the distinction between *randomly polarised light* and *unpolarised light*. Light from a standard source, such as an incandescent filament, or fluorescent tube, is unpolarised. A linear polariser placed in the beam will pass the same amount of light, whatever its orientation. On the other hand, light from a laser is usually polarised, although its polarisation at any instant may be unknown and may vary with time. Reflections from non-metal surfaces are often polarisation sensitive. Hence, if a polarised source is to be used for illumination, it is best to know what the polarisation is and hold it constant.

When a perfect polariser is placed in a linearly polarised beam of light of intensity I_0, the intensity of the light passing through it is given by *Malus's law:*

$$I = I_0 \cos^2(\theta) \tag{5.31}$$

where θ is the angle between the initial plane of polarisation and the axis of the polariser. A beam of unpolarised light contains a uniform mixture of linear polarisations at all possible angles. The average value of $\cos^2\theta$ is 0.5, so the transmission coefficient becomes

$$I = I_0/2 \tag{5.32}$$

In practice, some light is lost in the polariser. A Polaroid-type sheet polariser transmits about 38% of the incident light, while some birefringent prism types achieve transmission coefficients of over 49.9%.

If two polarisers are placed in tandem, and the angle between their polarising axes is θ in Malus's law, the transmission coefficient is

$$I = I_0 \cos^2(\theta)/2 \tag{5.33}$$

If $\theta = 90^\circ$ (the two axes are orthogonal), the polarisers are said to be *crossed*. In theory no light is transmitted. In practice, however, the transmission is not exactly zero. For example, when a room light is viewed through two crossed sheets of Polaroid material, a dark blue spot is seen. The ratio of the transmission of the unwanted component to the wanted component is called the *extinction ratio*. It varies from about 1:500 for Polaroid sheet polariser, to about 1:10^6 for a Glan-Taylor prism polarisers.

Any polarisation effects produced by a transparent material placed between two crossed polarisers increases the transmission. This principle is used to visualise the shear stress in glassware and clear plastics. Applying Malus's Law three times explains this. Imagine three perfect polarisers with axes at 0°, θ and 90°, placed in tandem. An unpolarised light source of intensity I_0 is used as a back-light. The light transmitted has intensity

$$I = I_0\cos^2(\theta)\sin^2(\theta)/2 \tag{5.34}$$

This has a maximum value of 0.125 when $\theta = 45^\circ$. When θ is either 0° or 90°, I is zero; no light is transmitted.

5.9.2 Brewster's Angle

Consider a light ray travelling across a boundary where there is a step change of refractive index. Some light will be reflected and some will enter the new medium. At one particular angle of incidence, called Brewster's angle the reflected ray will be strongly polarised (➜ *Figs. 5.84* and

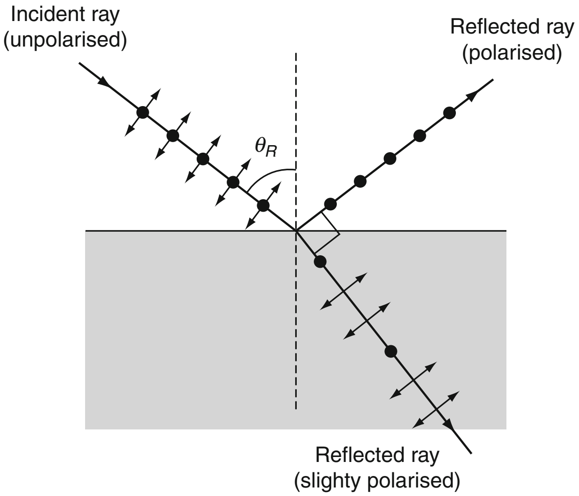


Fig. 5.84
Brewster angle. Notice that the angle between the refracted and reflected rays is 90°

➋ 5.5). The polarisation of the reflected ray is such that its **E**-vector lies in the same plane as the incident ray and the surface normal. Light with this polarisation is said to be *p-polarised*. Light with the orthogonal polarisation is said to be *s-polarised*. When unpolarised light strikes a surface at Brewster's angle, the reflected ray is always s-polarised. The value of Brewster's angle, θ_B, is given by

$$\theta_B = \tan^{-1}(n_1/n_2) \tag{5.35}$$

where n_1 and n_2 are the refractive indices of the two media. Brewster's angle for an air-glass interface is approximately 56°. (For glass, $n_2 \approx 1.5$; for air, $n_1 \approx 1$. For water, $n_2 \approx 1.33$.) Brewster's angle is approximately 53°. Since the refractive index depends on wavelength, Brewster's angle varies with wavelength. For angles of incidence close to Brewster's angle, the reflected ray is strongly but not completely polarised. By placing a suitably orientated polariser in the path of the reflected (or incident) ray; we can greatly reduce the intensity of the reflected beam. This is the principle upon which polarising sunglasses operate.

5.9.3 Birefringence

In a birefringent material, such as calcite, the velocity of light, and hence its refractive index, changes slightly, depending on the orientation of the E-vector. Again, decompose the ray of light into two separate rays whose E-vectors are orthogonal to one another. Passing light through a birefringent medium alters the phase relationship between these components. Calcite and other birefringent materials exhibit this behaviour because the distribution of free charge carriers (electrons) in the crystal structure is anisotropic. The orientation of the E-vector for rays travelling along the fastest path defines the *optic axis*, or *fast axis*, of the material. Such a ray is called the *o-ray*, or *ordinary wave*. The orthogonal wave is called the *e-ray*, or *extraordinary wave*. The o-ray sees the lowest refractive index of the birefringent material and the e-ray the highest. In calcite ($CaCO_3$), a birefringent material commonly used in optical components, the refractive indices are 1.658 (o-ray) and 1.486 (e-ray) at 0.59 µm. Birefringent materials can be used to separate the *o*- and *e*-rays; a prism made of calcite will separate a (monochromatic) unpolarised beam into two beams that are both polarised. Many glasses and plastics, especially when stressed, exhibit birefringence, as do ice, quartz, mica and several gem-stones.

5.9.4 Dichroism

The term *dichroic* is used in two ways in optics, to refer to the property of a material that either

(a) Separates light rays with different wavelengths, by transmitting some and reflecting others. (This is the way that the word was used in ➋ Sect. 5.6.)
(b) Absorbs light rays with certain polarisation states more strongly than others. In some materials, such as tourmaline, the polarising effect is wavelength dependent, in which case the material appears to have different colours depending on the polarisation of the light used to view it.

It is the second meaning that is relevant in this section.

5.9.5 Polarising Components

Polarisers can take various forms: plastic sheets, prisms, stacked plates, or optical coatings. The choice depends upon the polarisation purity needed, cost, robustness, physical size, weight and the working waveband.

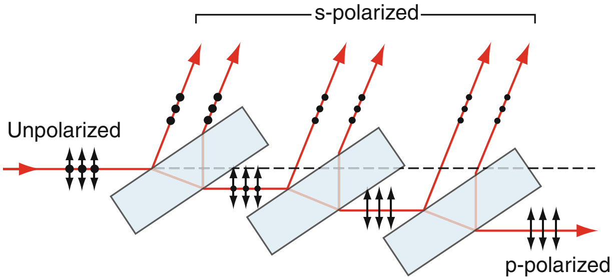
5.9.6 Polaroid Film

Polaroid is the generic name given to a range of synthetic plastic sheets containing needle-like crystals. These are aligned during manufacture, by stretching the film, or by applying an electric or magnetic field. The film absorbs light that is polarised so that the **E**-vector is parallel to the direction of the crystal alignment, but transmits light which is polarised perpendicular to it. The needle crystals act as the conductor array mentioned above. Both linear and circular polarising material is available in large sheets. The polarisation purity is not as good as the prismatic devices based on birefringence but Polaroid sheeting allows a shorter optical system to be built, that is lighter and considerably cheaper. The film can be damaged by scratching and high temperatures. Moreover, it does not polarise infra-red radiation, so this should be removed, by filtering.

5.9.7 Brewster Polariser

A polariser can be built using a stack of glass plates inclined at Brewster's angle to the incident beam. (For visible light in air and for a typical optical glass, Brewster's angle is about 57°, ➔ *Fig 5.85*.) Some of the *s*-polarised light is reflected from both surfaces of each plate. In a typical case, about 16% of the *s*-polarised light is reflected for each air-to-glass or glass-to-air interface. If there are *N* glass plates, the polarisation purity of the emergent beam is approximately

$$(1 - 0.16)2N \tag{5.36}$$



■ **Fig. 5.85**

Polariser consisting of a set of glass plates inclined at Brewster's angle (about 57°) to the optical axis. At each air–glass or glass–air interface, about 16% of the s-polarised light is reflected, leaving a greater proportion of the p-polarised light in the transmitted beam. Both the reflected and transmitted beams can be used for illumination but only the latter is useful for imaging applications

Hence, many plates are required to achieve even modest levels of polarisation purity. The reflected beam is strongly polarised but represents $2N$ spatially separated images of the light source and hence is unlikely to be useful in practice.

5.9.8 Thin Film Polarisers

A simple film polariser consists of a glass plate with a multilayer optical coating. In another method of construction, the coating is applied to the hypotenuse of a 90°/45°/45° glass prism. A similar prism is then cemented to it. (Do not confuse this with the Glan-type polarisers which also have the form of a cubic but rely on birefringence, not interference.) In both cases, optical interference within the coating film separates the polarised components. Since interference is sensitive to the wavelength of the light, this type of polariser is only able to operate over a narrow waveband. The cube-type polarisers generally perform better than the plate polarisers. Thin-film polarisers generally do not perform as well as Glan-type polarisers but are cheaper and provide two beams that are about equally well polarised.

5.9.9 Wollaston Prism

A Wollaston prism consists of two 90° calcite prisms, with perpendicular optic axes, cemented together on their hypotenuse faces to form a cuboid (➲ *Fig. 5.86*). Outgoing light beams diverge from the prism, giving two polarised rays, with the angle of divergence determined by the prisms' wedge angle and the wavelength of the light. Commercially available prisms are available with divergence angles from about 15° to 45°. The work well over a wide spectral range: 0.35–2.3 µm.

5.9.10 Nicol Prism

The Nicol prism polariser consists of a crystal of birefringent calcite that has been cut diagonally into two prisms and then cemented together again using Canada balsam (➲ *Fig. 5.87*). The crystal is cut such that the *o*- and *e*-rays are in orthogonal linear polarisation states. Total internal reflection of the *o*-ray occurs at the balsam interface which is therefore

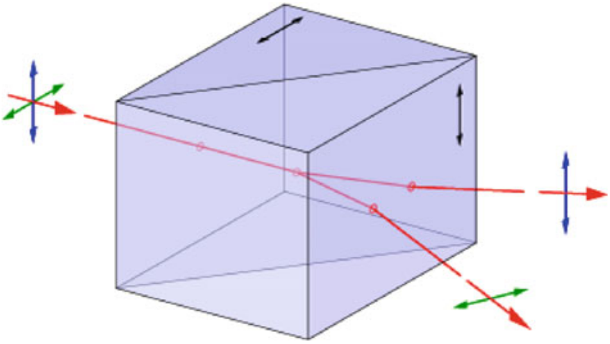


Fig. 5.86
Wollaston prism

deflected to the side of the cube. (The refractive index is larger in calcite than in balsam.) The *e*-ray, which experiences a smaller refractive index in the calcite, is transmitted directly across the balsam interface into the second prism. Nicol prisms produce very pure polarised light and are not wavelength sensitive.

5.9.11 Glan-Thompson Prism

A Glan-Thompson prism consists of two right-angled calcite prisms cemented together at their hypotenuse faces. (The optical axes of the crystals are both aligned perpendicular to the plane of reflection, ➋ *Fig. 5.88*.) The *o*-ray undergoes total internal reflection at the calcite–Balsam cement interface; only the *e*-ray is able to cross that interface. Notice that some of the e-ray energy is lost due to reflection, so the light deflected by the prism is not purely polarised. The Glan-Thompson polariser has a wider acceptance angle than the Glan-Foucault prism and works over a wide range of wavelengths: 0.35–2.30 µm. The linearly polarised e-ray is not deviated from its initial path.

5.9.12 Glan-Foucault Prism

The Glan-Foucault prism (or Glan-air prism) differs from the Glan-Thompson prism, in that two 90° calcite prisms are separated by a narrow air-gap (➋ *Fig. 5.89*). Total internal reflection

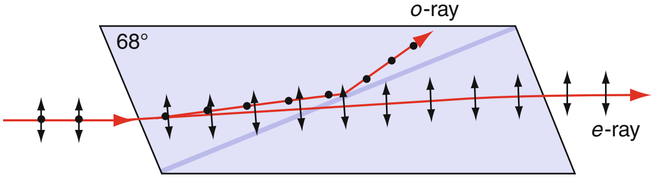


Fig. 5.87
Nicol prism polariser

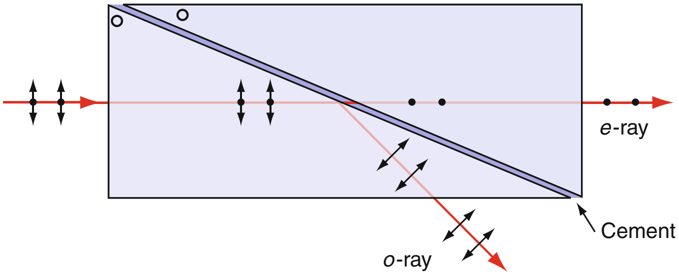
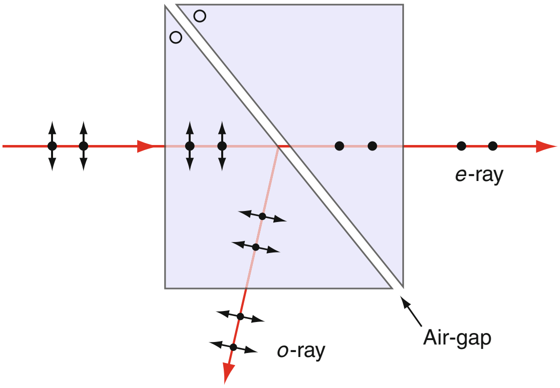


Fig. 5.88
A Glan-Thompson prism deflects the o-ray whilst transmitting e-ray. The two halves of the prism are joined with optical cement, and the crystal axes are perpendicular to the plane of the diagram

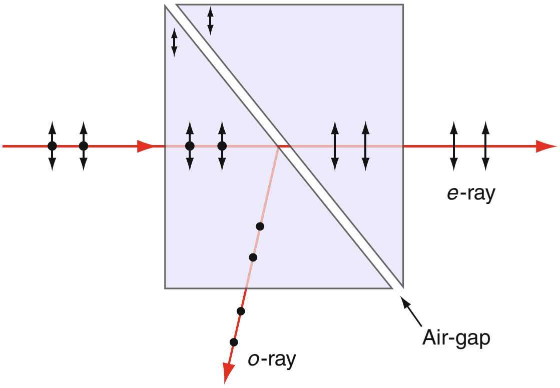
of the *o-ray* occurs at the calcite-air interface, so only *the e-ray* is transmitted straight through the prism. The reflected beam is not purely polarised. Compared to the Glan-Thompson prism, the Glan-Foucault device has a narrower acceptance angle. For this and other reasons, Glan-Foucault prisms are rarely used nowadays.

5.9.13 Glan-Taylor Prism

The Glan-Taylor prism is a polarising beam splitter and consists of two 90° prisms of calcite. Their hypotenuse faces are separated by a narrow air gap (➋ *Fig. 5.90*). The optical axes of the



◘ Fig. 5.89
Glan-Foucault prism deflects the o-ray and light, transmits the e-ray. Notice that the calcite prisms are separated by a narrow air gap. The optical axis of the calcite is perpendicular to the plane of the diagram



◘ Fig. 5.90
Glan-Taylor prism reflects the o-ray at the air-gap, transmitting only the e-ray. The reflected light is not purely polarised. The length-to-aperture ratio is 0.85

calcite crystals are aligned parallel to the plane of reflection. Total internal reflection of the *o-ray* occurs at the air-gap, so only the e-ray is able to cross it. The device is efficient, losing only a small proportion of the e-ray energy due to reflection at the second air-calcite interface. Although the transmitted beam is purely polarised, the reflected beam is not. Glan-Taylor polarising prisms are recommended for applications requiring a high degree of polarisation purity and high total transmission. They transmit well over a wide spectral range (0.35–2.3 µm) and the linearly polarised e-ray is not deviated significantly from its initial path.

5.10 Beamsplitters

A beam splitter is an optical device that divides the energy within a beam of light between two secondary beams, whilst maintaining the spatial distribution in each one. Beamsplitters can also be used to merge beams from two light sources, or superimpose two images. For example, it is possible to superimpose a calibration grid onto the view of a scene. Beamsplitters can also be used to allow two cameras to view the same scene in different ways, perhaps via different colour filters or polarisers.

5.10.1 Cube Beamsplitter

One of the most common forms of beamsplitter is a cube, made from two 90° triangular glass prisms that are cemented together at their hypotenuse faces (➋ *Fig. 5.91*). The hypotenuse face of one of the prisms is coated to provide a partially reflecting mirror. Light should enter this prism first. Normally, the coating thickness is adjusted so that the light intensity emerging from the two exit faces of the cube is approximately the same. Both entrant and exit faces should have antireflection coatings. Cube beamsplitters should be used with normally incident, collimated, or nearly collimated light, to avoid producing ghosting, spherical aberration and dispersion with varying wavelength. A cube beamsplitter is easy to mount, is physically robust and does not suffer from problems due to vibration. The coating is sealed inside the cube, so it is less susceptible to degradation due to chemical attack than plate or pellicle beamsplitters.

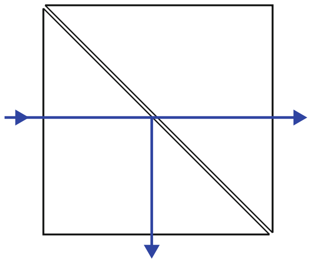


Fig. 5.91
Cube beamsplitter

5.10.2 Plate Beamsplitter

Another popular beamsplitter uses a half-silvered mirror, inclined at 45° to the path of the light beam. The beam should be collimated (i.e., all rays are parallel). This type of beamsplitter commonly consists of a plate of glass with a thin uniform coating of metal (e.g., aluminium or Inconel) deposited on one face. The thickness of the metal layer is normally adjusted so that 50% is of the light is transmitted and 50% is reflected. Other ratios can be produced. A plate beamsplitter is efficient but can produce ghost images as a result of reflections from the non-metallised surface (➔ *Fig. 5.92*). Simple geometric optical analysis shows that the spacing between ghost images is as follows:

$$\text{Adjacent to the transmitted beam}: \quad 2W\tan(\theta')/\sin(\theta) \tag{5.37}$$

$$\text{Adjacent to the reflected beam}: \quad 2W\tan(\theta')/\cos(\theta) \tag{5.38}$$

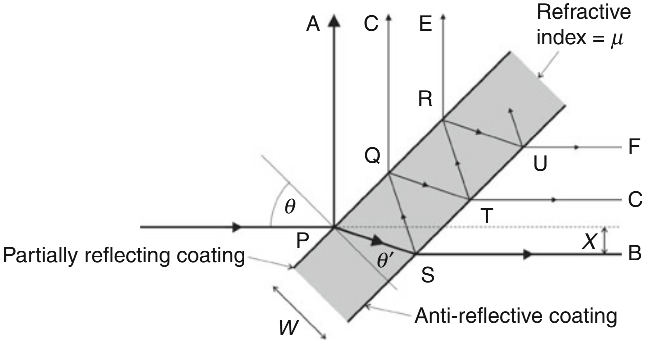
When $\theta = 45°$, both of these expressions reduce to

$$2\sqrt{2}W\tan(\theta') \tag{5.39}$$

For crown glass, the refractive index is 1.6. So, for a plate inclined at 45° to the incident beam, $\theta' = 26.2°$ and ➔ Eq. 5.39 reduces to

$$5.43W \tag{5.40}$$

Notice that the transmitted beam is offset slightly from the incident ray. Using the notation of ➔ *Fig. 5.92*, the displacement is equal to



Displacement of the transmitted beam:
$X = W\sin(\theta)(1\text{-}\cos(\theta)/\mu\cos(\theta'))$

Fig. 5.92

Beam-splitter based on a partially reflective coating. The input beam is split into two main components: A and B. In addition, low intensity beams emerge after multiple reflections within the substrate: C, E, F and G. Notice that beam B is offset slightly from the input beam by an amount X, W. If W is small, beams A, C and E are close together, as are B, F and G. Hence to avoid producing ghost images, it is important keep W small. In a pellicle beamsplitter, the substrate width (W) is very small, so ghosting and spherical and chromatic aberration are all negligible

$$\mathrm{W}\sin(\theta)(1 - \cos(\theta)/(\mu\cos(\theta')) \tag{5.41}$$

Again, letting $\theta = 45°$, $\mu = 1.6$, we find that $\theta' = 26.2°$ and the displacement is equal to

$$0.3616\mathrm{W} \tag{5.42}$$

5.10.3 Coatings

Two types of partially reflecting metal coating can be produced:

(a) *Spatially uniform coating*, deposited on the substrate by condensing metal from a vapour. The thickness of the metal film determines the transmission:reflection ratio.

(b) *Polka dot coating*, consists of a large number of small dots of metal deposited on the substrate. Each of these acts as a tiny mirror that reflects all of the light it receives. Of course, the clear spaces between these small mirrors allow light to travel through the substrate. Provided the reflective dots are small enough, the overall effect is to produce a beamsplitter that has a well defined transmission:reflection ratio. If the dots are positioned on a regular grid pattern, strobing may occur with the photo-sensor array. If they are too small, the dots will cause diffraction effects. For these reasons, it may be difficult to use a Polka dot beamsplitter in an imaging application. It is possible to manufacture a graded Polka Dot beamsplitter, so that the transmission:reflection ratio varies in a controlled way across the device.

A dielectric coating is normally applied to the non-metallised side of the plate, to reduce the intensity of the ghost images. In addition, a transparent coating may be used to protect the metallised surface from physical damage. It should be noted that plate beamsplitters are sensitive to the polarisation of the entrant beam. This can be a nuisance, or an advantage, depending on the application requirements.

For operation at a fixed wavelength, a mutli-layer dielectric coating can be used instead of metal for the reflecting film. This also offers scope for building a dichroic device that reflects wavelengths above/below a certain critical value and transmits wavelengths below/above that value. These were discussed in ❯ Sect. 5.6.

5.10.4 Pellicle Beamsplitter

A third design, called a pellicle beamsplitter, uses a thin, high tensile-strength elastic membrane (e.g., nitrocellulose), stretched tightly, like the skin of a drum, and coated on one side with a partially reflecting film. Pellicles are manufactured with two different coatings. Again, dielectric coatings are useful when working at one specific wavelength. However, a metal coating, containing a mixture of nickel, chromium, cobalt, and iron, can be made to produce a device that is almost independent of wavelength. Aluminium also allows operation over a wide spectrum.

A pellicle does not produce a ghost image of any significance, because the thin membrane provides a separation between the primary and ghost images of only a few microns.

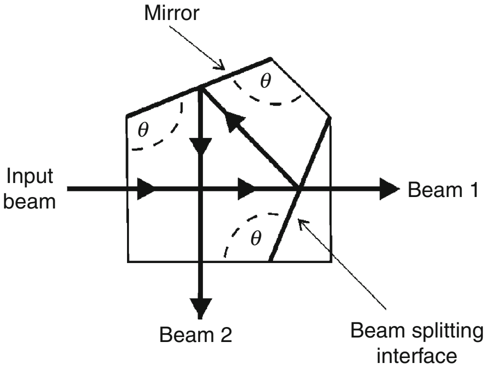


Fig. 5.93
Pentaprism beamsplitter. All angles marked θ are 112.5°

(The variable W in Eqs. 5.37 and 5.38 is very small.) In addition, chromatic and spherical aberrations are negligible in converging beams. The membrane normally provides low absorption across the visible spectrum but may attenuate other wavelengths. The thickness of the metal film can be adjusted during manufacture, to provide 50:50%, or other splitting ratios.

Pellicles suffer from two major disadvantages

(a) The membrane is very delicate; it must never be touched by fingers. A pellicle should be cleaned only with a gentle flow of clean, dry air.
(b) Acoustic noise and vibration can be troublesome, since the film resonates at certain frequencies.

5.10.5 Prism Beamsplitters

It is possible to build a robust, broadband beamsplitter using a pentaprism with one face coated with a partially reflecting metal film (*Fig. 5.93*). Such a device is heavier and more bulky than a plate or pellicle beamsplitter but it defines the angular separation of the reflected and transmitted beams precisely at 90°. Ghost images are eliminated and there is no offset for the direct beam (Beam 1 in *Fig. 5.93*) Neither beam is flipped horizontally nor vertically. There is no separation of rays with different wavelengths.

The use of prisms for separating rays according to their wavelength and polarisation has been discussed in Sect. 5.8.

5.11 Optical Windows

Optical windows are used to isolate one operating environment from another while allowing light to pass. A closed box is sometimes used to protect the camera, in which case an optical window allows the camera to "see" the outside world. (See *Fig. 5.94*.) An important point to note is that the optical window should be regarded as a sacrificial element; it is the one optical component that must face the hostile factory environment.

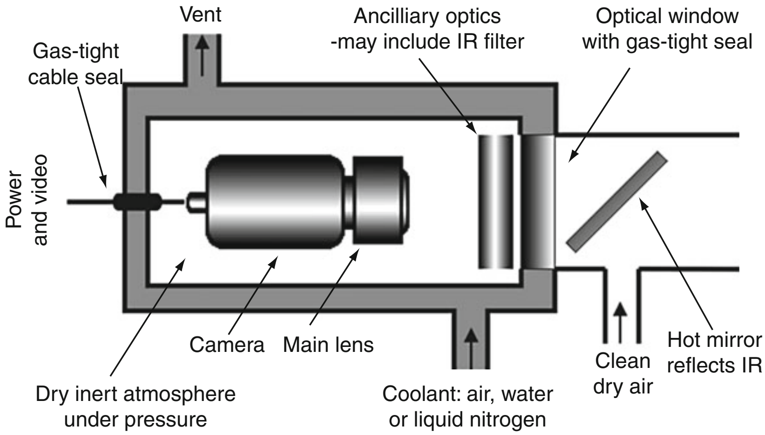


Fig. 5.94
Protecting a camera and optics in a hostile environment

When choosing a suitable window material, the following points should be should be borne in mind:

- There should be very little attenuation of light over the working spectrum
- There should be a minimal change to the spectrum
- There should be very little scattering of light
- Wavefront distortion should be minimal
- The window should be resistant to damage by
  - Water
  - Oil
  - Acids
  - Other chemicals used in the manufacturing process
  - Nuclear radiation
  - Scratching (The window may be rubbed vigorously during over-zealous cleaning.)
  - Flying debris
  - Extremes of temperature
  - Thermal shock due to rapid temperature changes
  - Large temperature gradients

In Sect. 5.2, we pointed out that, for many purposes, sapphire is an ideal material for this function. A synthetic sapphire window can be provided with a thin metal coating, which allows it to be heated electrically so that condensation is eliminated. The spectral transmission characteristic of sapphire is vey flat over the visible waveband. (See *Fig. 5.17*.) However, it should not be forgotten that other window materials may be more suitable for some applications.

5.12 Practical Issues

We conclude this chapter with some advice about building and maintaining optical systems in good working order. Adhering to a few simple rules will improve overall system performance and reliability. Failure to do so will result in disappointment, which may, in turn, lead to frustration and exasperation with the whole of Machine Vision technology. Do not forget that cleaning optical surfaces properly is just as important as making sure that computers remain virus free, or that cameras are not overheated.

5.12.1 Cleaning Optical Components

Optical components must be kept clean, to ensure that they perform properly. Failure to do so will result in a general and/or local reduction of light levels, a change in the spectral composition and polarisation, as well as an unpredictable scattering effect. Optical surfaces can be contaminated in many ways, for example by water (condensation, splashes drops or streaks), saliva (from sneezing or coughing), oil, dust, fumes from the manufacturing process – and fingerprints. Contamination can be kept to a minimum by returning unused optical components to their cases immediately after use, fitting lens caps and using a protective enclosure (➋ *Fig. 5.94*). A jet of clean dry air can also help to keep dust away. However, even with these precautions, the optics will almost inevitably accumulate dust, stains or other form of contamination.

5.12.1.1 General Advice

- Keep optical components in their protective cases when they are not in use.
- Fit lens caps whenever possible. Be careful to avoid touching the lens when doing so.
- If possible, place optical components, including lights within protective enclosure.
- Do not touch optical surfaces with bare fingers; always wear powder-free gloves, or finger cots.

5.12.1.2 Cleaning Methods

The six cleaning methods described below are suitable for different types of optical components and surfaces (➋ *Table 5.3*).

1. *Blowing* The first step in cleaning any kind of optical device is to remove dust, grit, lint or other loosely held particles using a dry, dust-free air blast. This reduces the chance of scratching the optical surfaces. Blowing is specifically recommended for bare metal, soft coatings, and membrane beamsplitters (pellicles). These types of surfaces should never be touched with bare fingers.
2. *Drop-and-drag* It may be necessary to remove the optical component from its mount. When wiping optical surfaces, be careful not to drag any dirt across it, as this could result in scratching. (Use blowing first.) To eliminate streaks, the following procedure should be followed two to three times, wiping in different directions each time.
   1. Place a drop of solvent, such as methanol, on a lens tissue paper.
   2. Place the wet area of the tissue onto the optical surface to be cleaned.
   3. Slowly drag the tissue across the surface until the lens tissue is dry.
3. *Wiping* If the optical component cannot be taken out of its mount, or if more stringent cleaning is required, use the following procedure.
   1. Fold a piece of lens tissue paper, creating a folded edge.
   2. Wet the folded edge with acetone.

   3. Wipe the optic with the lens tissue paper with one continuous motion. If necessary, carefully apply light pressure. When repeating this process, always use a new lens tissue, to avoid redepositing dirt.
   4. Apply final wipe with methanol, as this does not leave streaks on the surface. Acetone and isopropyl alcohol are also effective but can leave streaks.

      If the device is very small, cotton buds can be used instead of lens tissue.

4. *Bathing* Small and lightly contaminated optical components can be cleaned by immersing them in solvent. Do not use this method with cemented components, or mounted optics, both of which are likely to be damaged.
   1. Line a shallow dish with lens tissue and fill it with methanol.
   2. Place the optical element in the dish, clean its surfaces gently with a piece of soaked cotton wool.
   3. Agitate the solvent for a few minutes.

Table 5.3

Methods for cleaning optics

	Cleaning method					
Component/surface/coating	1	2	3	4	5	6
Aluminium, bare metal	✓					
Bare substrate	✓	✓	✓	✓	✓	✓
Coloured filters and heat absorbing filters	✓	✓	✓	✓	✓	
Compound camera lens (Assembly cannot be dismantled. Usually no need to clean the rear surface as this is well protected)	✓		✓			
Cube beamsplitters, polarising prisms, other compound prisms	✓	✓	✓			
Fibre optic bundles (illumination and imaging)	✓	✓	✓		✓	
Gold, bare metal	✓					
Half-silvered plate beamsplitters	As reflective coating					
Interference filters (short-, long-, band-pass)	✓	✓	✓	✓	✓	
Mirrors	As reflective coating					
Multi-layer dielectric anti-reflective coating (E.g., HEBBAR™, Melles Griot, Inc.)	✓	✓	✓	✓	✓	
Multi-layer dielectric reflective coating (E.g., MAXBRite™, Melles Griot, Inc.)	✓	✓	✓	✓		
Neutral density filters	✓	✓	✓			
Pellicle beamsplitters	✓					
Prisms (some prisms also have reflective coating)	As anti-reflective coating					
Protected and enhanced aluminium coatings	✓	✓	✓	✓		
Protected silver and gold	✓	✓	✓	✓		
Silver, bare metal	✓					
Simple lens	As anti-reflective coating					
Single-layer MgF_2 coating	✓	✓	✓	✓	✓	

   4. Repeat steps 1–3 with acetone.
   5. Remove the component from and blow until it is dry.
   6. If stains remain, apply the drop-and-drag method.
5. *Soap Solution Bathing* Surfaces contaminated by fingerprints, oil or grease, cannot be cleaned properly using methanol, or acetone, which simply re-distribute the pollution. In this case, a soap solution can be used instead. Mild, non-abrasive soap without additives (green soap), should be used.
   1. Using a soap solution, apply either the Methods 3 (Wiping) or 4 (Bathing).
   2. Rinse thoroughly with de-ionized water.
   3. Repeat method 3/4, using methanol or acetone.
   4. Blow dry to remove water.
6. *Ultrasonic Cleaning* The component is placed in an ultrasonic bath for about 1 min. Notice that this method may damage dielectric coatings.

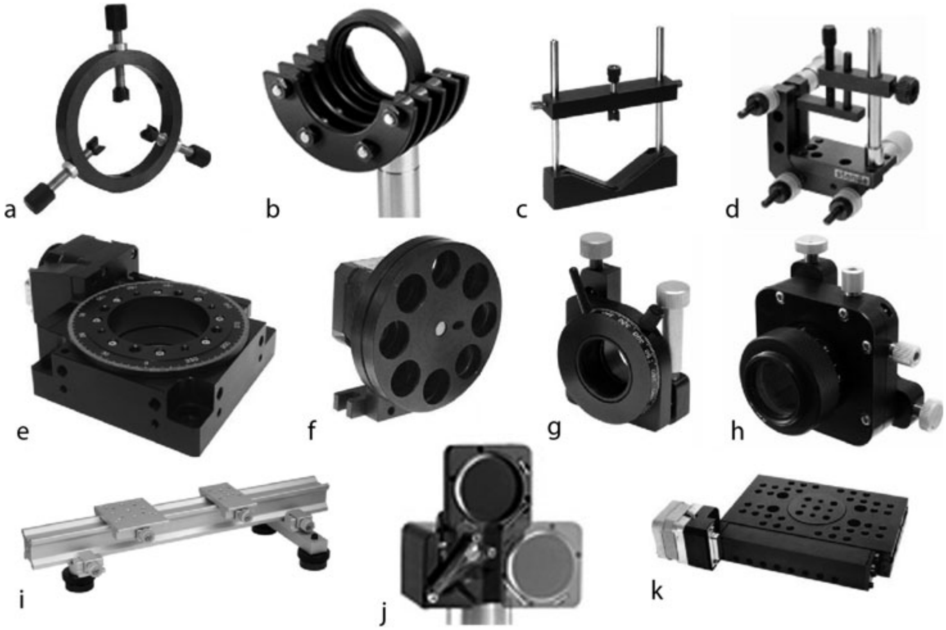
5.12.2 Protecting Optical Systems and Sensors

Following the maxim that protection is better than cure, it is worthwhile discussing methods by which fragile and delicate optical components can be made more secure. Small lighting devices, optical components and cameras can be enclosed in an air-tight box, with only a small tough viewing window. Sapphire is an ideal material for this window, which has to face the full hazards of the work-place. To maintain good visibility, the window must be cleaned regularly, either automatically or as part of a disciplined maintenance schedule. It must also be accepted that this window is a sacrificial element in a vision system. If anything in the optical sub-system breaks as a result of an accident, the window will! A well designed enclosure will protect the optics from dust, flying grit, steam, water and oil splashes and hot air. If necessary, the enclosure can be water-cooled and in extreme circumstances, liquid nitrogen could be used as a coolant instead. IR blocking filters and/or hot mirrors can be used can be used to protect the camera. IR absorbing filters get hot and can be cooled with an air purge. An air jet can also be used to protect the window. A tube placed around the window will keep much of the flying debris away from it. Again, an air jet injected into the tube will help to keep the window clean. These ideas are summarised in ➜ *Fig. 5.94*.

5.12.3 Mounting Optical Components

Providing proper mountings for optical components is likely to be regarded as a mere detail to be left to the very last moment during the design of a vision system. The system may well fail to fulfil its intended function if the lenses and/or mirrors are not held rigidly in their appropriate places. The mechanical fixtures that hold them are just as important as the optical elements themselves, or the camera and the image processing algorithms. A casual attitude to the mounting of optical components will almost invariably lead to disappointment! Mounting the components in an optical system requires just as much care as choosing the elements in it! Failure to give proper attention to optical mountings will result in blurred images, low image contrast and inconsistent overall performance by the vision system. The task of building a robust optical system has been made easier, because manufacturers have devised specialist holding devices for holding lenses, mirrors, filters, prisms, polarisers, beam-splitters, lamps and cameras. The prime requirements for any optical mounting system are that it must be stable with varying temperature and free from vibration. Each optical component, including the lamp(s) and camera(s), must be held in such a way that it does not move relative to the other components. In some harsh environments, this may require the use of a heavy optical bench, with sophisticated pneumatic damping on each of its supporting pillars. In less demanding applications, a much lighter and far less expensive mounting rail will suffice. The reader is warned that optical mounting devices are not cheap and may even be more expensive than the bits of glass that they hold. Buying good mechanical fixtures is well worth the investment, since failure to mount the optical components solidly in place will result in serious deterioration of the image quality. Making a safe, stable mounting for optical components requires careful design and precise machining; it is unlikely that a non-specialist designer and general machine-tool operator can produce better, cheaper fixtures than the commercial equipment.

It is not our intention here to describe in detail the range of fixtures that have been devised for even standard optical devices. A few are illustrated in ➲ *Fig. 5.95*. A variety of lens mounting



◘ Fig. 5.95

Optical mounting and precision translation devices. (a) Lens holder, manual centring. (b) Multiple filter holder. (c) Single lens/filter holder. (d) Cylindrical lens holder. (e) Universal prism holder. (e) Motorised rotation stage. (f) Motorised filter wheels. (Two wheels can be rotated independently) (g) Precision polariser holder, with calibrated rotation scale. (h) Multi-axis position device with manual controls. (i) Mounting rail. Notice the anti-vibration feet and that the saddles by repositioned and locked in place easily (j) Filter holder with motorised flipping in/out of optical path. (k) Motorised linear translation stage (Reproduced with permission from Standa, Vilnius, Lithuania, http://www.standa.lt/. Accessed 23rd April 2008)

rings, not shown, have been designed specifically for certain types/sizes of lenses. Bespoke designs for this function would probably do little more than replicate what is already available in off-the-shelf products and would very likely be less accurate and more expensive. A variety of precision motorised devices exists for adjusting optical systems. These include rotary, linear and multi-axis translation stages, filter wheels, filter flippers. The last two can insert/remove colour filters, neutral density filters or polarisers from the optical path.

5.12.4 Centration

The mechanical axis and optical axis exactly coincide in a perfectly centred lens. For a simple lens, the optical axis is defined as a straight line that joins the centres of lens curvature. For a plano-convex or plano-concave lens, the optical axis is the line through the centre of curvature and perpendicular to the plane surface. The mechanical axis is determined by the way in which the lens will be mounted during use. There are typically two types of mounting configurations: edge mounting and surface mounting. With edge mounting, the mechanical axis is the centreline of the lens mechanical edge. Surface mounting uses one surface of the lens as the primary stability reference for the lens tip and then encompasses the lens diameter for centring. The mechanical axis for this type of mounting is a line perpendicular to the mounting surface and centred on the entrapment diameter. The tolerance on centration is the allowable amount of radial separation of these two axes, measured at the focal point of the lens. Centration error is measured by rotating the lens on its mechanical axis and observing the orbit of the focal point.

5.12.5 Alignment and Calibration

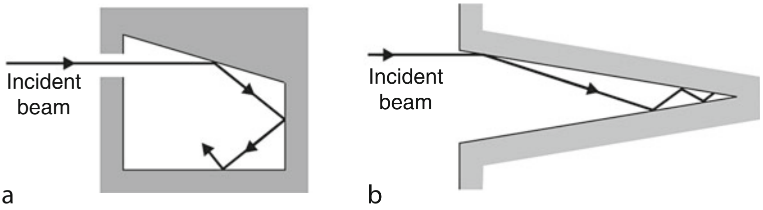
It is important that all of the components in an optical system are aligned accurately. To do this properly requires a clear understanding of the way that each component works. Much can be achieved using simple tools and ingenuity based on a good general appreciation of optical principles:

(a) A small, low power (diode) laser
(b) A single low power LED mounted on a wand
(c) Two opaque diaphragms with tiny pinhole apertures
(d) Micrometer mounts for the pinholes/laser can be useful when aiming for precise alignment
(e) A calibration target consisting of a board with an array of high-contrast horizontal and vertical lines. A higher contrast target can be made with a back-lit board with an array of holes. The same effect can be produced using an array of LEDS
(f) As in (e) but with the holes/LEDs arranged around a series of concentric circles

A useful ploy, in some situations, is to replace the camera temporarily with a "point" light source: a laser or a single LED. A narrow beam of light is then projected backwards through the optical system during alignment. Beam steering components (mirrors, prisms, beamsplitters and lenses) should be aligned first; filters and polarisers, can be removed from the system during the initial phases of alignment. A laser beam can be used to define the optical axes. Systems with mirrors and prisms have non-linear optical axes, while those containing beam splitters have more than one. After alignment, the behaviour of the system should be investigated with the "point" light source placed both on and off the optical axis. To do this properly requires precise position control, hence the inclusion of micrometer mounts (d) in the list above. This allows the origin and destination of stray light to be determined.

5.12.6 Light Absorbing Materials and Structures

When a 50:50 beamsplitter is used in the illumination path of a vision system, half of the light it receives from the lamps is immediately discarded and must be disposed of properly. To absorb the large quantities of unwanted light that this type of system produces requires that special techniques be used. The naïve approach to producing a light absorbing surface is to use commercially available matt black paint. However, this still reflects a significant amount of light. Black velvet provides an efficient light absorbing surface but this is hardly practical for use in a dirty factory environment. Although amorphous carbon (soot) is a good absorber, it is friable. A standard approach to improving matters in photographic lens hoods, is to use rifling on the inside surface. A closed box with a small aperture in its side wall and painted matt black inside is a good absorber. A deep hollow cone is too (➲ *Fig. 5.96*). Any light reflected by the inside surface of the cone is reflected deeper into it, where more of the unwanted light will be aborbed. This leads to the idea of building a surface consisting of a series of small conical depressions. A similar idea is to use a stack of razor blades; the side of the stack is a good light absorber. Creating a sponge-like structure from a material that itself has a low reflection coefficient seems to offer the best approach to building a really efficient light absorber. One recently announced coating material aborbs 10–20 times more less light than conventional techniques [8]. (It is claimed by its inventors to be 25 times blacker than conventional black paint.) This coating consists of an alloy of nickel and phosphorus. The surface is pitted, with tiny craters. Another type of very black surface ha been constructed using an array of carbon nano-tubes aligned with the path of the incoming light. These work on the same principle as illustrated in ➲ *Fig. 5.96b*. Absorption is claimed to be better that 99.99%. At the time of writing (April 2008) these materials are still expensive.



▣ Fig. 5.96

Light absorbing structures. In both cases, the internal surface is coated with absorbent material, such as matt black paint. (a) Box (b) Deep conical pit

5.13 Links

5.13.1 Web Links

Key:

Wk, Wikipedia;
MG, CVI Melles Griot;
NC, Newport Corporation;
RMI, Rocky Mountain Instrument Co;
EO, Edmund Optics.

Topic	URL
Abbe prism, Wk	http://en.wikipedia.org/wiki/Abbe_prism
Abbe-Koenig prism, Wk	http://en.wikipedia.org/wiki/Abbe-Koenig_prism
Achromatic Doublet Lenses, NC	http://www.newport.com/Achromatic-Doublet-Lens-Tutorial/141058/1033/catalog.aspx
Achromatic Lenses, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=312
Amici prism, Wk	http://en.wikipedia.org/wiki/Amici_prism
Amici roof prism, Wk	http://en.wikipedia.org/wiki/Amici_roof_prism
Anti-Reflection Coatings, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=247
Application Examples, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=267
Aspheric Lenses, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=315
Beamsplitter, E Weisstein	http://www.ericweisstein.com/research/thesis/node98.html
Beamsplitters, Guide for Designers, MG	http://www.mellesgriot.com/pdf/P_Handbook_Beamsplitters.pdf
Beamsplitters, NC	http://www.newport.com/Beamsplitters/141111/1033/catalog.aspx
Borescopes and Fiberscopes, EO	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=1732
Calibration Standards, EO	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=1729
Cleaning Optics, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=265
Cleaning Optics, NC	http://www.newport.com/Care-and-Cleaning-of-Optics/141176/1033/catalog.aspx
Components Sets, NC	http://www.newport.com/Optics-Sets/311771/1033/catalog.aspx
Cylindrical Lenses, NC	http://www.newport.com/Beam-Shaping-with-Cylindrical-Lenses/144888/1033/catalog.aspx

Topic	URL
Design and Tolerancing, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=260
Dichroic prism, Wk	http://en.wikipedia.org/wiki/Dichroic_prism
Diffraction-Gratings, NC	http://www.newport.com/Diffraction-Gratings-Selection-Guide/421120/1033/catalog.aspx
Dove prism, Wk	http://en.wikipedia.org/wiki/Dove_prism
Ellipsoidal-Mirror, NC	http://www.newport.com/Custom-Ellipsoidal-Replicated-Mirror/372498/1033/catalog.aspx
Endoscopes, Lighthouse Imaging Corp.	http://www.lighthouseoptics.com/info.php?info_id=17
European Optical Society (EOS),	http://www.myeos.org/
FAQs 1, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=284
FAQs 2, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=285
FAQs 4, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=276
Filters, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=273
Focusing and Collimating, NC	http://www.newport.com/Focusing-and-Collimating/141191/1033/catalog.aspx
Formulas, NC	http://www.newport.com/Optics-Formulas/144956/1033/catalog.aspx
Fundamental Optics, MG	http://optics.mellesgriot.com/optics_guide_1.asp
Geometric Optics, Colwell CH	http://online.cctt.org/physicslab/content/phyapb/review/summaries/geometricoptics.asp
Geometric Optics, Tatum JB	http://www.astro.uvic.ca/~tatum/goptics.html
Geometric Optics, SparkNotes	http://www.sparknotes.com/physics/optics/geom
Geometric Optics, University of Colorado	http://phet.colorado.edu/simulations/sims.php?sim=Geometric_Optics
Geometric Optics, University of Queensland	http://www.physics.uq.edu.au/people/mcintyre/vergences/optics/geomalt.html
Geometric Optics, Christian W	http://dev.physicslab.org/asp/applets/opticsbench/default.asp
Glasses, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=259
Glossary, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=297
Hyperlinks, EO	http://www.edmundoptics.com/techSupport/DisplayCategory.cfm?categoryid=32
Illumination Devices, EO	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=34
Illumination Devices, EO	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=246

Topic	URL
Illumination, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=264
Lambert's cosine law, Wk	http://en.wikipedia.org/wiki/Lambert%27s_cosine_law
Lens Edge Blackening, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=316
Lens Selection Guide, NC	http://www.newport.com/Lens-Selection-Guide/140908/1033/catalog.aspx
Library Index, MG	http://optics.mellesgriot.com/optics_guide.asp
Light Meters, EO	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=1726
Light Shaping Diffusers, NC	http://www.newport.com/Light-Shaping-Diffusers/141131/1033/catalog.aspx
Material Properties, MG	http://optics.mellesgriot.com/optics_guide_4.asp
Materials, NC	http://www.newport.com/Optical-Materials/144943/1033/catalog.aspx
Materials, RMI.	http://rmico.com/capabilities/materials
Metallic Mirror Coatings, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=269
Mirror Selection Guide, NC	http://www.newport.com/Mirror-Selection-Guide/141086/1033/catalog.aspx
Modulation Transfer Function (MTF), EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=317
Off-The-Shelf Integration, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=313
Optical Coatings, MG	http://optics.mellesgriot.com/optics_guide_5.asp
Optical Flats, EO EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=254
Optical Materials,, RMI.	http://rmico.com/technical-notes/transmission-curves
Optical Society of America (OSA)	http://www.osa.org/
Optical Specifications, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=252
Optical System Integration, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=268
Optics (on-line textbook), Benjamin Crowell	http://www.lightandmatter.com/area1book5.html
Optics 2001	http://www.optics2001.com/
Optics Calculator, NC	http://www.newport.com/OpticalAssistant/
Optics Fundamentals, NC	http://www.newport.com/Optics-Fundamentals/604533/1033/catalog.aspx
Optics Glossary, MG	http://www.mellesgriot.com/glossary/wordlist/glossarylist.asp
Optics, Wk	http://en.wikipedia.org/wiki/Optics

Topic	URL
Parallel Windows, NC	http://www.newport.com/Parallel-Windows/141051/1033/catalog.aspx
Pellicle Beamsplitters, GlobalSpec	http://optical-components.globalspec.com/LearnMore/Optics_Optical_Components/Optical_Components/Pellicle_Beamsplitters
Pellin-Broca prism, Wk	http://en.wikipedia.org/wiki/Pellin-Broca_prism
Pentaprism, Wk	http://en.wikipedia.org/wiki/Pentaprism
Photographic optics, P. van Walree	http://www.vanwalree.com/optics.html
Photonics-Dictionary, NC	http://www.newport.com/Photonics-Dictionary/604499/1033/catalog.aspx
Polarisation, NC	http://www.newport.com/Polarization/144921/1033/catalog.aspx
Polarisers, NC	http://www.newport.com/Polarization-Optics-Selection-Guide/141146/1033/catalog.aspx
Polarization Techniques, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=257
Porro prism, Wk	http://en.wikipedia.org/wiki/Porro_prism
Prisms, Wk	http://en.wikipedia.org/wiki/Prism_%28optics%29
Projector Design, EO	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=251
Reflecting Optics, RMI.	http://rmico.com/technical-notes/102-reflecting-optics
Replicated Mirrors, NC	http://www.newport.com/Replication-Process/374840/1033/catalog.aspx
Replicated Optics, NC	http://www.newport.com/Replication-Justification/372497/1033/catalog.aspx
Sapphire Windows, NC	http://www.newport.com/Sapphire-Windows/378626/1033/catalog.aspx
Scotchlite, Wk	http://en.wikipedia.org/wiki/Scotchlite
Spatial-Filters, NC	http://www.newport.com/Spatial-Filters/144910/1033/catalog.aspx
Speciality Lenses, NC	http://www.newport.com/Cylindrical-and-Specialty-Lenses/545083/1033/catalog.aspx
Spectrometers, EO	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=1725
Specular and Diffuse Reflection, Henderson T	http://www.glenbrook.k12.il.us/gbssci/phys/class/refln/u13l1d.html
Spherical Mirrors, Calvert JB	http://www.du.edu/~jcalvert/optics/mirror.htm
SPIE (Soc. Photo-Optical Instrument. Eng.)	http://spie.org/
Substrates, RMI	http://rmico.com/technical-notes
Technical Library, MG	http://optics.mellesgriot.com/optics_guide.asp
Technical Literature, MG	http://www.mellesgriot.com/products/technicalliterature.asp#optics

Topic	URL
Telecentric Lenses, MG	http://www.mellesgriot.com/resourcelibrary/specsheets/machinevision/default.asp
Temperature Sensitive Liquid Crystal, EO	http://www.edmundoptics.com/onlinecatalog/displayproduct.cfm?productID=1642
Test Targets, EO	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=289
Transmission Curves 1, RMI	http://rmico.com/technical-notes/103-transmission-curves
Transmission Curves 2, RMI	http://rmico.com/technical-notes/zns-cleartran-sapphire-spinel#zns
Transmitting Optics, RMI.	http://rmico.com/technical-notes/transmitting-optics
Transmitting, RMI.	http://rmico.com/technical-notes/101-transmitting-optics
Triangular prism, Wk	http://en.wikipedia.org/wiki/Triangular_prism_%28optics%29
White Reflectance Coating, EO	http://www.edmundoptics.com/onlinecatalog/displayproduct.cfm?productID=1325

Acknowledgements

➤ *Figures 3–7*, ➤ *9*, ➤ *14–19*, ➤ *20*, ➤ *22–29*, ➤ *33–35*, ➤ *42–48*, ➤ *61*, ➤ *69*, ➤ *83*, ➤ *90* and much of the other source material for this chapter were kindly supplied by CVI Melles Griot, Albuquerque, New Mexico, USA.

References

1. BBC News (2008) http://news.bbc.co.uk/1/hi/sci/tech/2732487.stm. Accessed 24 Apr 2008
2. Jerrett J (2011) Image formation by a concave mirror. http://www.physics.mun.ca/~jjerrett/mirror/concavem.html. Accessed 27 Feb 2011
3. Batchelor BG (1999) Characterizing the illumination in an automated visual inspection work cell. In: Miller JWV, Solomon SS, Batchelor BG (eds) Proceedings of the SPIE. Machine vision systems for inspection and metrology VIII, Vol 3836, pp. 134–143, 1999
4. Spherical mirrors (2011) Wikipedia http://en.wikipedia.org/wiki/Spherical_coordinates. Accessed 1 Mar 2011
5. Conklin EL (2011) Ingenious devices & wonderful toys. http://ericconklin.com/ingeniousdevices.html. Accessed 1 Mar 2011
6. Thomas DE (1980) Mirror images. Scientific American Dec 1980, pp 158–171
7. Geyer C, Danilidis K (2011) Catadioptric projective geometry. http://www.cis.upenn.edu/~kostas/mypub.dir/geyer01ijcv.pdf. Accessed 1 Mar 2011
8. BBC News (2008) Blacker is the new black. http://news.bbc.co.uk/1/hi/sci/tech/2732487.stm. Accessed 1 Mar 2008
9. Catadioptric Cameras for 360 Degree Imaging, University of Columbia, http://www1.cs.columbia.edu/CAVE/projects/cat_cam_360/. Accessed 7 Apr 2008
10. Kirtane MM (1924) Optical system for viewing tubes. US Patent 1,653,575
11. http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel3/3678/10818/00509208.pdf, http://www.patentstorm.us/patents/5717455-claims.html
12. http://www.cis.upenn.edu/~kostas/omni.html, http://cmp.felk.cvut.cz/demos/OmnidirectionalVision.html

6 Telecentric, Fresnel and Micro Lenses

Spencer D. Luster[1] · *Bruce G. Batchelor*[2]
[1]Light Works, LLC, Toledo, OH, USA
[2]Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_6,

Abstract: Telecentric lenses provide constant perspective and large magnification depth of field. For this reason, they are often well suited for gauging and a variety of other demanding Machine Vision applications. This is especially true when viewing objects that have variable depth and/or lateral position. Lighting issues relating to the use of telecentric optics for viewing and lighting are discussed, together with their ability to provide sensitive detection of defects in clear or flat and shiny materials. Telecentric lenses can view objects, with constant perspective, anywhere within a cylindrical volume. On the other hand, hypercentric lenses can view objects within a conical region, whose apex is in front of the lens. (A fish-eye lens views objects within a conical region, whose apex is behind the lens.) As a result, hypercentric lenses are able to view the top and sides of an a cylinder simultaneously.

By adding a spacer between certain types of hypercentric lens and the camera, it is possible to view inside a bore, even when the viewing aperture, at the camera end of the bore, is very small. Both telecentric and hypercentric lenses are only able to work effectively if the front refracting element is larger than the object being viewed. Manufacturing large conventional (i.e. curved-surface) lenses is very expensive. Fresnel lenses provide a convenient alternative. There are three common types of optical component made by forming numerous very small grooves and/or mounds on a plane surface. *Fresnel* lenses employ very fine prismatic surface features on a thin plane plate. *Micro-lens arrays* consist of numerous small lenses (lenslets) placed close together on a plane substrate. In both of these cases, the optical formulae explained in ➲ Chap. 5 are applicable, because the surface features, while small, are still much larger than the wavelength of light. On the other hand, *Diffraction (or Binary)* optical components have such small surface features that they cause interference due to diffraction; refraction usually plays no significant part in the operation of these devices. Some hybrid refractive-diffractive lenses are made but these are uncommon.

6.1 Telecentric Lenses

The term *telecentric lens* is used to refer to one of the following:

1. A lens that receives mostly parallel rays of light from the object being inspected. This is called an *Object Space Telecentric (OST)* lens.
2. A lens that projects mostly parallel rays of light onto the camera's photo-detector array. This is called an *Image Space Telecentric (IST)* lens.
3. A lens that does both of the above. This is called a *Double Telecentric (DI)* or *Bi-telecentric* lens.

➲ *Figure 6.1* shows all three types. There are very few cases in Machine Vision applications where IST lenses are useful, so we won't discuss them here. Most commercial telecentric lenses are of the OST variety, with a few vendors providing DT lenses. There are advantages to the latter, which will be discussed below.

6.1.1 Advantages of Telecentric Lenses

Telecentric lenses offer several important advantages for Machine Vision and frequently transform what would otherwise be difficult applications into much simpler ones. They often

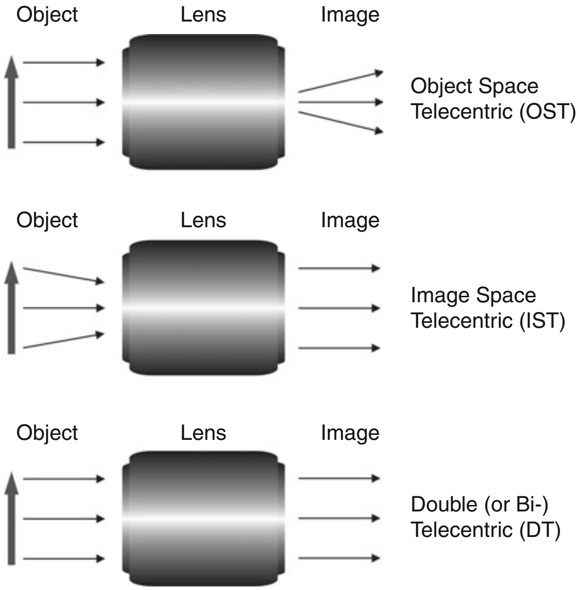

Object
Lens
Image
Object Space Telecentric (OST)
Object
Lens
Image
Image Space Telecentric (IST)
Object
Lens
Image
Double (or Bi-) Telecentric (DT)

Fig. 6.1
Three types of telecentric lenses

make the processing of images in software much simpler, faster and less expensive. Indeed, it might reasonably be claimed that telecentric lenses form one of the major pillars that has made Machine Vision successful in practice!

6.1.1.1 Constant Magnification

A telecentric lens is able to provide a nearly constant magnification, irrespective of object-camera distance. Everyday experience teaches that a conventional lens, including that in the human eye, produces an image that becomes larger as the subject approaches the observer/ camera (➋ *Fig. 6.2*). The size of an image, and information about the type of object that produced it, are combined in the brain, so that the object-eye distance can be estimated. Human beings are relatively insensitive to image size. To most people, a dog seen at a range of 10 m appears very little different from the same animal at 5 m. Size constancy, as this is called, is a useful feature of human vision. It can, however, easily mislead us when building a vision system, as it gives false expectations. Varying image size is a nuisance! Automatic compensation for image size variation is possible using a vision system. If the absolute size of the object is known, it is possible to estimate its range from the size of its image and vice versa. However, measuring image size does not always lead to estimates of distance that are acceptable for precise gauging applications. In many applications, it is necessary to measure the size, irrespective of object-camera distance. Varying image size has the disadvantage that it complicates the image processing and can introduce many "wasted" pixels. The near-constant magnification of a telecentric lens is therefore of great practical value.

While it is always desirable to constrain the presentation of objects for inspection, this is not always possible. Variation of object-camera distance occurs naturally in the industrial context

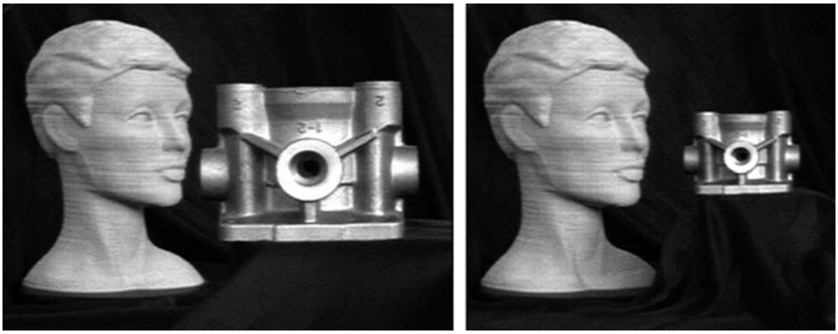


Fig. 6.2
Size variation in a conventional lens occurs when objects are moved along the camera's optical axis. Which object is larger? Size variation, like this, that are due to movement along the optical axis are greatly reduced using a telecentric lens

and we are fortunate if it can be limited by the clever use of mechanical handling. Two situations where this is not possible are described below:

- Bottles on a conveyor may arrive for inspection at a wide range of positions across a conveyor belt. Hence, a side-view camera views them at different distances.
- Components on circuit boards might be mounted at different heights. A low-altitude overhead camera fitted with a conventional lens will therefore see large size variations.

In absolute terms, these variations in the object-camera distance might seem insignificant, but when trying to obtain high-precision measurements of object size or distance, the resulting errors might be totally unacceptable.

When using structured lighting (laser-line triangulation) for height measurement, it is important to use a lens that provides constant magnification. Failure to do so will require significantly more complicated algorithms to estimate surface height.

Many objects must be viewed with the camera close to one face and a long way from others (➋ *Fig. 6.3*). When viewed using a conventional lens, these different faces will each be viewed with a different magnification.

Good telecentric lenses, especially of the Double Telecentric variety, overcome these problems; the image size does not vary significantly over a large distance along the optical axis. How far an object can change its distance and still appear to be the same size is called *Magnification Depth of Field*. This term is completely different from image-sharpness depth of field, which is what most people refer to when using the simple term "*Depth of field*."

6.1.1.2 Constant Perspective

Unlike conventional lenses, OST and DT lenses view the object from nearly the same angle across the whole field of view (➋ *Fig. 6.3*). The effect is the same as viewing the object from an infinite distance using a conventional lens. This can be very important for gauging applications, especially for non-flat objects, or for applications in which it is necessary to look down the bore of a tube, or pipe, to check for obstructions (➋ *Figs. 6.4* and ➋ *6.5*). Like a pair of calipers, these lenses are capable of providing accurate diameter, or width, measurements.

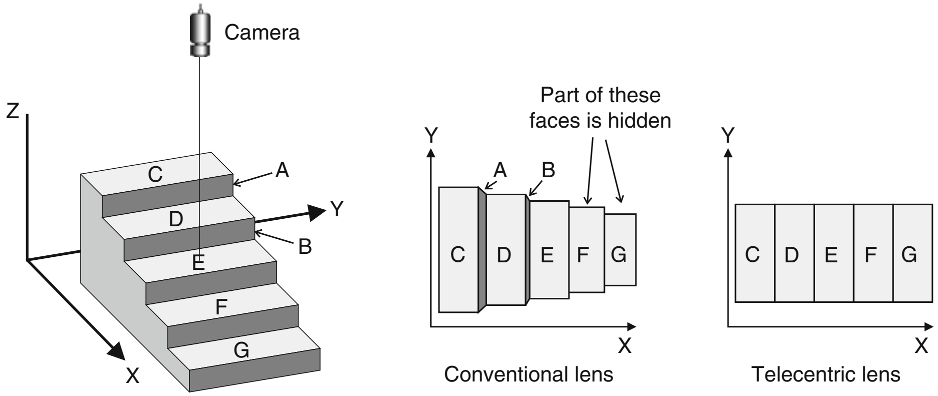


Fig. 6.3
Size constancy in telecentric lenses is not shared by conventional lenses. (a) Scene (b) Image created by a conventional lens (Geometric distortion introduced by the lens has been ignored) (c) Image created by a telecentric lens. Notice that faces A and B are not visible but that C, D, E, F and G are fully visible

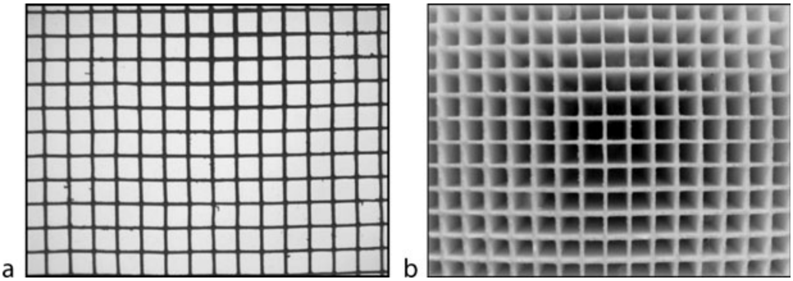


Fig. 6.4
Constant perspective across the whole visual field. (a) Catalytic converter element imaged with a telecentric lens. This object consists of a "honeycomb" structure, containing square tubes, 6 × 6 × 100 mm^3 (b) A conventional lens does not provide constant perspective. Notice the apparent barrel distortion, which is just perspective error

6.1.1.3 Small Angular Acceptance Range

Any lens can be "stopped down," to reduce its entrance pupil diameter. As the *f-number* (*f*/#) increases, the aperture *inside* the lens is made smaller. The cone enclosing the light rays from a given object point passing through the lens becomes narrower. The center axis of every little cone of light emanating, from every point on the object, converges toward the entrance pupil of the lens. There can be large angles between the central axes of these cones. For OST and DT lenses, the axes of all these cones are parallel, since the lens will not accept non-parallel rays. Such a lens can be useful for detecting certain types of subtle defects in transparent, or reflective, flat material. This feature of telecentric lenses is often overlooked, but we will provide a good example later (➋ *Fig. 6.6*).

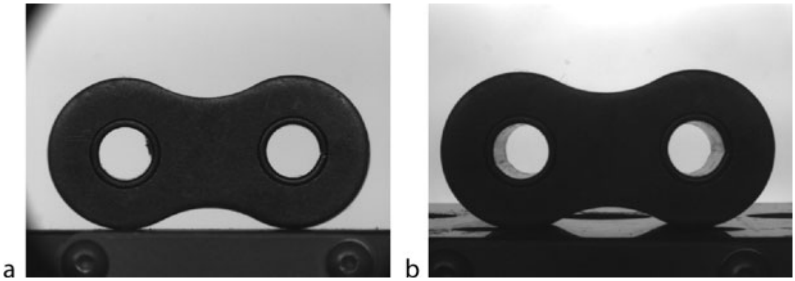


Fig. 6.5
Link of a motorcycle chain, with a blocking defect (a) Telecentric lens (b) Conventional lens

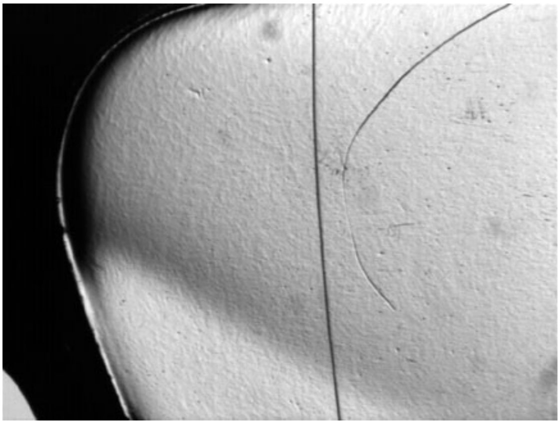


Fig. 6.6
Lens for a pair of plastic safety spectacles, imaged with a telecentric lens and collimated light

Telecentric lenses, of whatever kind, do not have larger image-sharpness depth-of-field than conventional lenses; this kind of depth-of-field is only a function of magnification and f-number. (This assumes that the lens is has been designed properly and is of a high quality of manufacture.)

6.1.2 Disadvantages of Telecentric Lenses

One of the most serious problems with telecentric lenses is that the first (i.e., objective) element must be at least as large as the object being inspected. Refer again to *Figs. 6.2* and *6.3*, which show that this must be the case, since the lens only accepts rays of light that are nearly parallel. This is not the case, of course, with conventional lenses, which are very often smaller than the objects they view. The cost of a large telecentric lens can be quite high. However, there are other solutions:

- A Fresnel lens may be used as the front element. This will reduce the cost and weight. Telecentric lenses with Fresnel front elements of 1.5 m diameter, are perfectly feasible.

However, the image quality is reduced (often considerably) compared to a conventional lens. (Fresnel lenses are discussed later in this chapter, ➋ Sect 6.3)

- A catadioptric (mirror based) telecentric system may be used instead. Again, this can reduce the cost and weight, compared to a telecentric lens using a large refractive front element. (The reader who is unfamiliar with Fresnel lenses may be reassured that they are commonplace devices. A Fresnel lens usually forms the main optical element in an overhead projector. They are also used in visual displays and rear-view mirrors in automobiles).
- If a line-scan sensor is being used, the cost of a telecentric lens incorporating a very large Fresnel or catadioptric front element, may be reduced very significantly.

For many commercial telecentric lenses, especially of the OST variety, it is difficult to provide good telecentric imaging with long working distance. This is less of a problem for DT lenses.

A DT lens may be constructed around a pin-hole aperture, located at the focal points of two simple lenses. (➋ *Fig. 6.7*) The aperture must be very small to make sure that only parallel rays are accepted. However, this greatly reduces the brightness of images produced by the lens. In many Machine Vision applications, this may not be a serious disadvantage, since lighting a small area is normally sufficient. (The area that needs illumination is only a little large than that of the front aperture of the telecentric lens).

6.1.3 Applications of Telecentric Lenses

6.1.3.1 Gauging

The large magnification depth-of-field of a telecentric lens makes it ideal for a great variety of dimensional gauging applications. It is possible to think of a telecentric lens as forming an optical caliper with very long jaws. Typical gauging applications include measuring threaded fasteners, machined parts, the geometry of bottles, and component placement on PCBs.

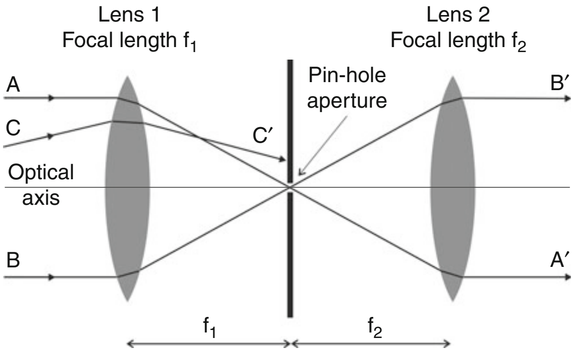


Fig. 6.7

Constructing a telecentric lens with a pin-hole aperture. Rays AA′ and BB′ travel through the aperture, while ray CC′ does not. In practice, several lens elements may be used on each side of the aperture, to ensure a bright image without serious aberrations (The pin-hole inevitably blocks much of the light, thereby reducing the image brightness)

Any application that requires the measurement of an object with little or no perspective error, or with varying distance, is suitable for a telecentric lens.

6.1.3.2 Constant Perspective

Non-gauging applications that make use of the constant perspective property of telecentric lenses include inspection of tubes, pipe bores, gun barrels and the verification of hole clearance. An extreme case of this is shown in *Fig. 6.4*. This image of a catalytic converter element was captured using a telecentric lens with a very large field-of-view. It would not be possible to view more than a few of the cells (nine at the most) with a conventional lens.

Figure 6.5 compares the diverging field of view of a conventional lens with that of a telecentric lens. The defect (surplus material in the left-most hole) can only be seen with a telecentric view.

6.1.3.3 Defect Detection in Transparent or Flat Materials

A telecentric lens collects rays of light that are very nearly parallel. Hence, it is possible to detect defects that divert light away from a parallel path. Nominally flat transparent and reflective sheet materials, such as glass, plastic, polycarbonate (optical discs), or simple plastic packaging can be inspected very effectively with a telecentric lens. Scratches, cracks, pits, blisters, pimples, embossing, inclusions, bubbles, water/oil droplets, etc., and local variations in refractive index all divert light and hence are able to produce high contrast images with a telecentric lens. These can be difficult to see using conventional lenses and lighting, In addition, telecentric lenses can often make subtle defects clear on mirror-like surfaces (e.g., CDs, DVDs, silicon wafers, optical components) *Fig. 6.6* shows an example of imaging part of a pair of plastic safety glasses. With a conventional lens, the defects are barely visible, even to the human eye. With a telecentric lens and collimated back lighting (parallel beam), the defects are sharply defined.

6.1.4 Lighting

As often stated in this book, illumination can be critically important. Poor lighting can seriously affect the performance of a vision system, even in seemingly straightforward applications such as back-lit gauging. For example, difficulties may arise when measuring bright shiny objects. *Figure 6.8* illustrates this point by showing a shiny sphere with a presumed large, diffuse back light.

One solution to this problem is to mask off the diffuse light source so that it is only just larger than the object being inspected. An equivalent approach is to move the light source far behind the object. Either case reduces the angular range of the light rays that approach the object. Sometimes, however, this technique is impractical. The object may move around too much to allow effective masking, or there may be no room for a distant diffuse light source in the inspection station.

A more expensive, but technically superior solution is to use a collimated back light such as shown in *Fig. 6.9*. A small light source placed at the back focus of an appropriate lens will produce nearly parallel or collimated rays of light as an output. Such a light provides a close

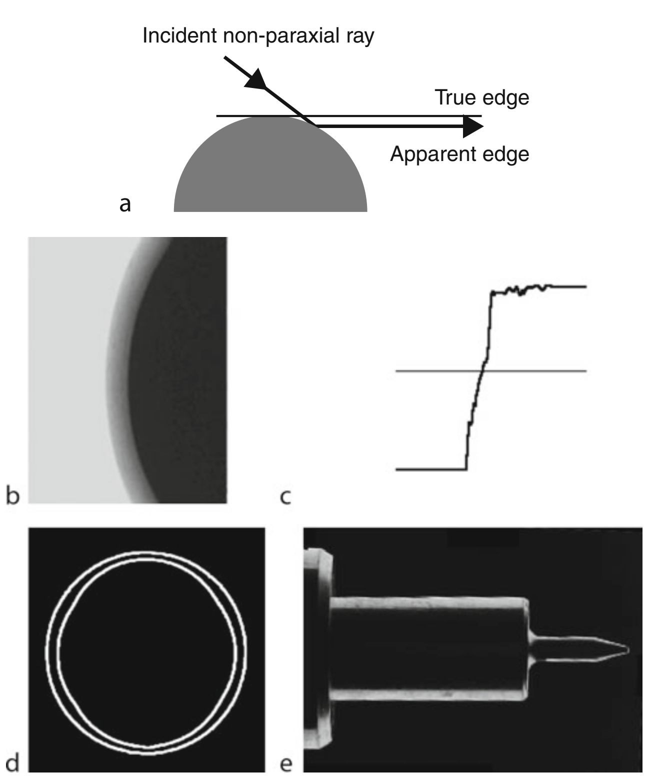


Fig. 6.8
Large-area diffuse back lighting of a shiny object allows non-paraxial rays causes reflections that could be avoided by using telecentric lens (a) Ray optics explains why this happens. (b) Sample image, derived from a brightly polished metal sphere in front of a large "light box." Ordinary back-illumination like this produces a light grey annulus around a very much darker disc. The grey band is caused by reflections of non-parallel rays. (c)Intensity profile shows the gradual change of intensity within this grey band. (d) True contour (*outer curve*) and the apparent edge contour (inner curve). Clearly, it is easy to under-estimate the diameter of the sphere. (e) The effects of non-paraxial illumination angles are emphasised in dark-field illumination. The non-paraxial rays which illuminated the object (a carburretor needle) to produce the bright edge would be eliminated by a telecentric collimator

source that allows the object to move about during inspection. Furthermore, this is a much more efficient use of light. When well-aligned, all the rays of light from the collimator enter the telecentric lens and contribute to the image. Where a diffuse light source may require many watts of power to provide enough light for high-speed applications, a collimated source may operate at a tenth to a hundredth of the power.

6.1.5 Comparison of Telecentric and Conventional Lenses

Telecentric lenses have better magnification depth-of-field than do conventional lenses. Furthermore, Double Telecentric lenses are better than Object Space Telecentric lenses in this

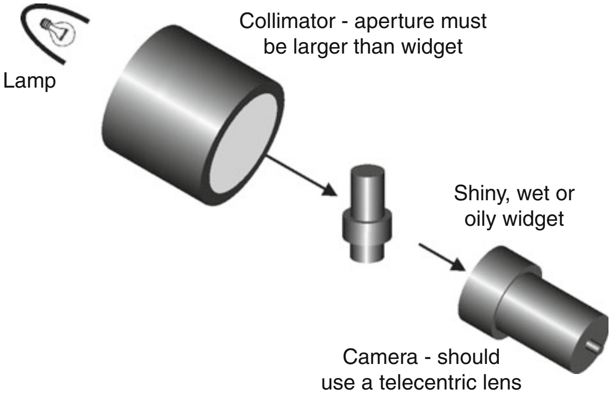

Lamp
Collimator - aperture must be larger than widget
Shiny, wet or oily widget
Camera - should use a telecentric lens

Fig. 6.9

Collimated light source for improved telecentric imaging of shiny objects. In practice a very small, intense light source would be used. The beam generated by the collimator contains rays that are almost all, very nearly paraxial

Table 6.1

Comparing magnification depth-of-field for conventional, OST and DT lenses

Object distance change (mm) centred on 100mm stand-off	Magnification (Theoretical)		
	Conventional (20mm fl.)	OST (100mm fl.)	DT (afocal)
0 mm	0.25	0.250	0.250
4 mm	0.238	0.248	0.250
8 mm	0.227	0.245	0.250
12 mm	0.217	0.243	0.250
Max. change	13.2%	2.8%	0.0%

respect. ❯ *Table 6.1* quantifies this, by comparing the changes in magnification versus object distance for these three types of lenses. In each case, it is assumed that the nominal optical magnification is 0.25X, and that the nominal stand-off distance (lens to object) is 100mm. If we ignore the limiting factor of image sharpness depth-of-field, a double telecentric lens could theoretically have perfectly constant magnification over an infinite range of object distances. In practice, real lenses with finite manufacturing tolerances cannot meet this ideal. A DT lens will have a greater magnification depth-of-field than will an OST lens, and much greater than the equivalent stand-off distance for a conventional lens.

❯ *Table 6.2* compares the merits of telecentric and conventional lenses. In many cases, with a little thought and care, telecentric lenses used at the “front end” of a Machine Vision system can make the “back end” engineering much easier and more cost-effective. Do not be put off by the increased price compared to conventional lenses; investing in a telecentric lens can often reduce the *system cost* significantly.

Table 6.2

Comparing telecentric and conventional lenses

Feature	Type of lens	
	Conventional (CL)	Telecentric (TL)
Advantages	Low cost Greater availability, compared to TL Greater flexibility, compared to TL	Constant magnification (little change with varying object-lens distance) No perspective error Narrow acceptance angle
Disadvantages	Magnification varies with object-lens distance Perspective Error	Darker images than CL Heavy Higher cost than CL Requires large first element (refractive lens, Fresnel lens or mirror)
Applications	Colour checking Counting objects/features Detecting foreign bodies Existential inspection Grading & sorting Inspecting large objects Inspecting textured surfaces	Inspecting deep holes, pipes, tubes, etc. Inspecting plain surfaces Inspecting transparent film Metrology Object/feature alignment

6.2 Hypercentric Lenses

Inspection tasks often require seeing a lot more of an object than a single view can achieve. The usual solution is to rotate the object, which is often time-consuming, or use multiple cameras. Sometimes, employing an optical view-splitter works very well. Other times, however, a preferred solution would be to capture a continuous view all around the object. Hypercentric lenses offer one such solution. A hypercentric, or pericentric, lens provides a converging view of an object, so that its top and sides can be viewed simultaneously. It has the reverse effect of the more familiar fish-eye lens. The operation of a hypercentric lens is explained in *Fig. 6.10*.

In normal mode, hypercentric lenses provide a converging view, as if aimed at a single point, called the Convergence Point (CP). The volume that can be well-imaged is contained within an imaginary truncated cone, called the Near View Cone (*NVC*). This is the blue hatched region in *Fig. 6.10a*. The dimensions of this region are *T*, *B* and *Max WD*. For standard Hyper-Eye lenses, *T* varies from 15.2 mm to 25.6 mm, *B* from 35.3 mm to 51.2 mm, and *MaxWD* from 28 mm to 42 mm. Other options are available.

Some hypercentric lenses can also serve as a long stand-off borescope, by simply adding a short spacer (typically 0.5–1.5 mm) between the lens and camera. Although counter-intuitive, this increases the focus distance beyond the CPD. The lens can therefore view the inside of a bore, even those with very small openings (*Fig. 6.11*). Unlike standard borescopes, this arrangement provides a large working distance between the lens and the object being viewed. This reduces the risk of parts crashing into the lens, makes it easier to arrange for suitable illumination within the bore.

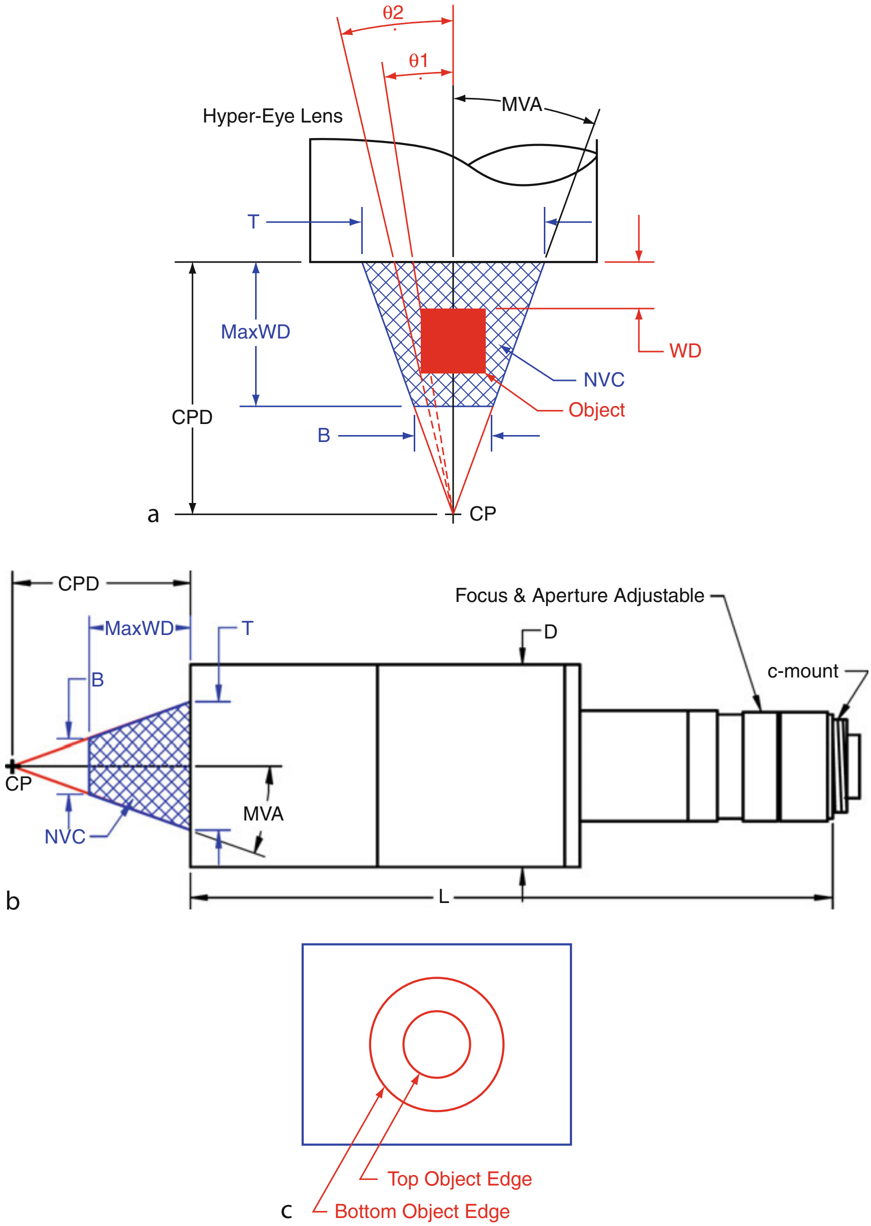


Fig. 6.10

Hypercentric lens. (a) The lens receives rays of light as if they all originate from a single Convergence Point (CP). The distance from the front of the lens to CP is the Convergence Point Distance (CPD). Any object that is within the Near Viewing Cone (NVC) can be focused onto the camera detector. Rays of light that are collected at smaller angles (1) are imaged closer to the center of the detector. Those collected at higher angles (2) are imaged farther out. (b) Hypercentric lens with a C-mount fitting, plus focus and aperture controls. (c) A hypercentric lens used to view a cylindrical object end on can see its top and sides

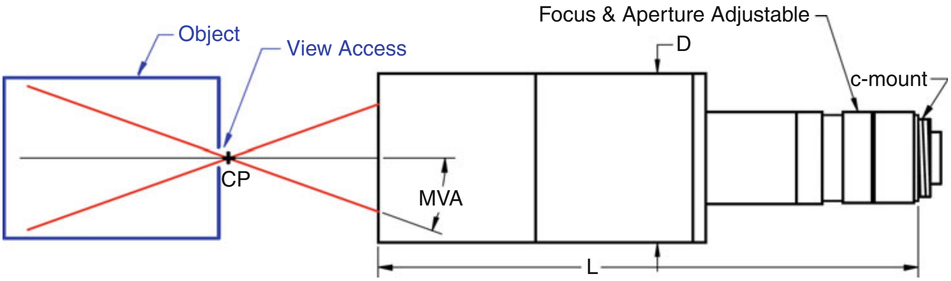


Fig. 6.11

Using a hypercentric lens as a borescope. A small spacer (not shown) is inserted on the C-mount between the lens and camera on the C-mount

Table 6.3

Commercially available hypercentric lenses. [Hyper-Eye™ lenses, Light Works, http://www.lw4u.com/, Accessed 27th October 2010]

Model	HE-5008A2	HE-5012A2	HE-7508A3	HE-7512A3
L (Length)	179 mm	178 mm	267 mm	268 mm
D (Max. Diameter)	55.9 mm	55.9 mm	81.3 mm	81.3 mm
CPD	49 mm	49 mm	84.5 mm	84.5 mm
T	35.3 mm	38.4 mm	51.2 mm	51.2 mm
B	15.2 mm	16.5 mm	25.6 mm	25.6 mm
MaxWD	28 mm	28 mm	42 mm	42 mm
MVA	21°	21.5°	17°	17°

➋ *Tables 6.3* and ➋ *6.4* summarise the properties of a range of commercially available hypercentric lenses. ➋ *Figure 6.12* shows a sample of images produced using a this type of device.

6.3 Fresnel and Micro Optics

The term "*Micro-optics*" may refer to the very small conventional optical components employed in fibre-optic communications systems, and medical imaging instruments such as needle endoscopes. There is however, another meaning for this term and is the one that we use here. There are three common types of optical component made by forming numerous very small valleys and/or mounds on a plane surface. *Fresnel* lenses employ very fine prismatic surface features on a thin plane plate. *Micro-lens arrays* consist of numerous small lenses (lenslets) placed close together on a plane substrate. In both of these cases, the optical formulae explained in ➋ Chap. 5 are applicable, because the surface features, while small, are still much larger than the wavelength of light. On the other hand, *Diffraction (or Binary)* optical components have such small surface features that they cause interference due to diffraction; refraction plays no significant part in the operation of these devices.

Table 6.4
Showing how the ray angles from the CP correspond to the detector at the image plane. This relates to the Hyper-Eye™ range of lenses produced by Light Works LLC, http://www.lw4u.com/] The table indicates that the HE-5012A2 lens is best used with a 1/2" format or larger detector. The image radius at 20.5° is 2.35mm (diameter 4.7mm) The standard 1/2" format detector is 4.8mm x 6.4 mm, therefore the image circle will just fit inside it. A 1/3" format detector (3.6 mm × 4.8 mm) would not be a good match.The HE-5008A3 and HE-7508A3lenses are typically for use with 1/3" format or larger cameras, while the HE-5012A3 and HE-7512A3 are suitable for use with 1/2" format or larger cameras

HE-5008A2		HE-5012A2		HE-7508A3		HE-7512A3	
Angle (°)	Image radius	Angle (°)	Image radius	Angle (°)	Image radius	Angle (°)	Image radius
9	0.66 mm	9.5	1.07 mm	9	0.72 mm	9	1.08 mm
10	0.74 mm	10.5	1.19 mm	10	0.81 mm	10	1.21 mm
11	0.82 mm	11.5	1.31 mm	11	0.89 mm	11	1.33 mm
12	0.90 mm	12.5	1.42 mm	12	0.98 mm	12	1.47 mm
13	0.98 mm	13.5	1.54 mm	13	1.07 mm	13	1.60 mm
14	1.06 mm	14.5	1.66 mm	14	1.16 mm	14	1.74 mm
15	1.14 mm	15.5	1.78 mm	15	1.26 mm	15	1.88 mm
16	1.21 mm	16.5	1.90 mm	16	1.36 mm	16	2.03 mm
17	1.28 mm	17.5	2.01 mm	17	1.46 mm	17	2.18 mm
18	1.36 mm	18.5	2.13 mm				
19	1.43 mm	19.5	2.24 mm				
20	1.49 mm	20.5	2.35 mm				
21	1.57 mm	21.5	2.45 mm				

6.3.1 Fresnel Lens

A Fresnel lens consists of a thin sheet of transparent material, such as Polymethyl Methacrylate (acrylic resin), with a series of fine grooves impressed on one, or both, of its sides. (One surface is usually plane with no grooves.) The grooves act like tiny prisms see (➋ *Figs 6.13* and ➋ *6.14*). The angles of their inclined surfaces vary across the face of the lens and are chosen so that a parallel coaxial beam is focussed to a very small spot. Little light is lost by absorption, since the plastic sheet is thin. This also reduces the weight of the lens. Fresnel lenses with a high groove density can produce good quality images, whereas a low groove density improves efficiency, which is important in light-gathering applications. The prismatic grooves disperse light, making Fresnel lenses prone to chromatic aberration. (Of course, in many Machine Vision applications, monochromatic illumination can be used, which will eliminate this aberration.) Fresnel equivalents of aspheric, convex, concave, fish-eye and cylindrical refracting surfaces are all available. Using two "crossed" Fresnel cylindrical lenses, it is a simple task to perform anamorphic warping in which the two image axes are magnified by different amounts. These axes are not necessarily orthogonal. Fresnel lenses that operate in the infrared waveband are also manufactured.

Fresnel lenses are usually made by moulding, stamping or rolling and can be manufactured in bulk, cheaply and easily. Very large Fresnel lenses can be made for a modest cost that would not be possible with conventional lenses. For example a Fresnel lens with a diameter of 1.65m costs under €220. (US $300, January 2010) A Fresnel lens for an overhead projector costs about €40, while a wide-angle automobile rear-window lenses costs about €5.

It is, of course, possible to make reflecting surfaces on the same principle. (*Figs 6.15* and *6.16*) Fresnel reflectors are sometimes used in overhead projectors and illumination optics.

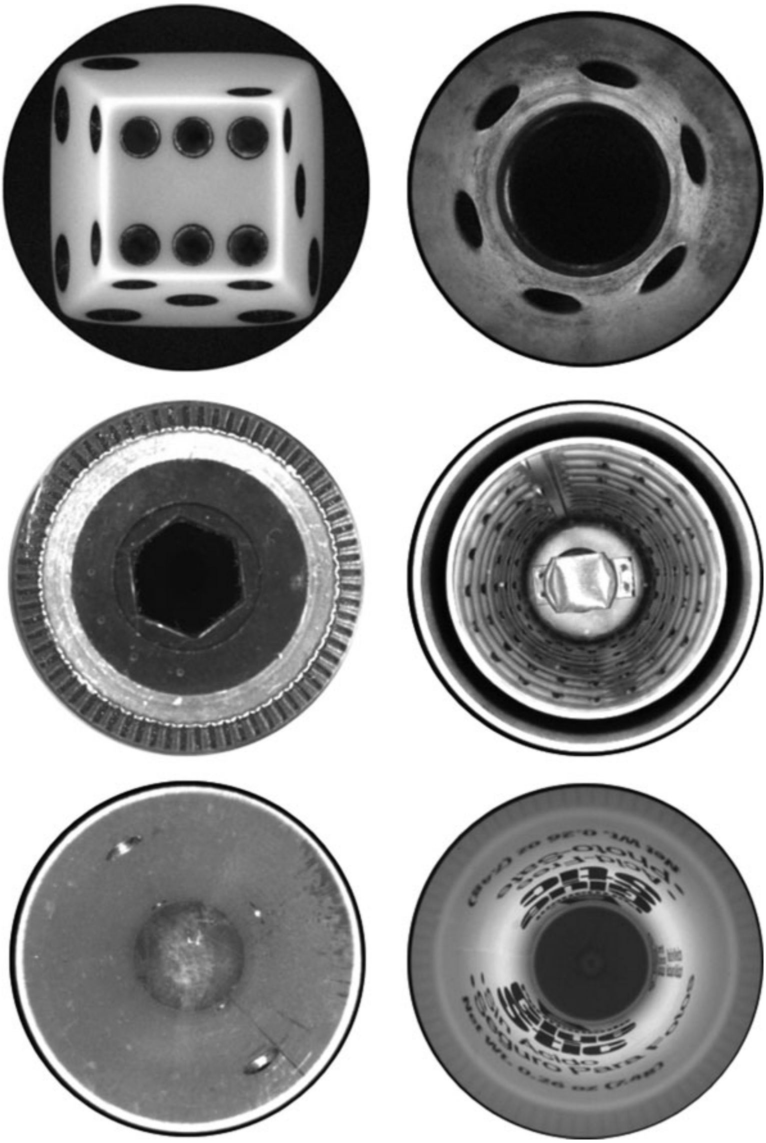


Fig. 6.12
(Continued)

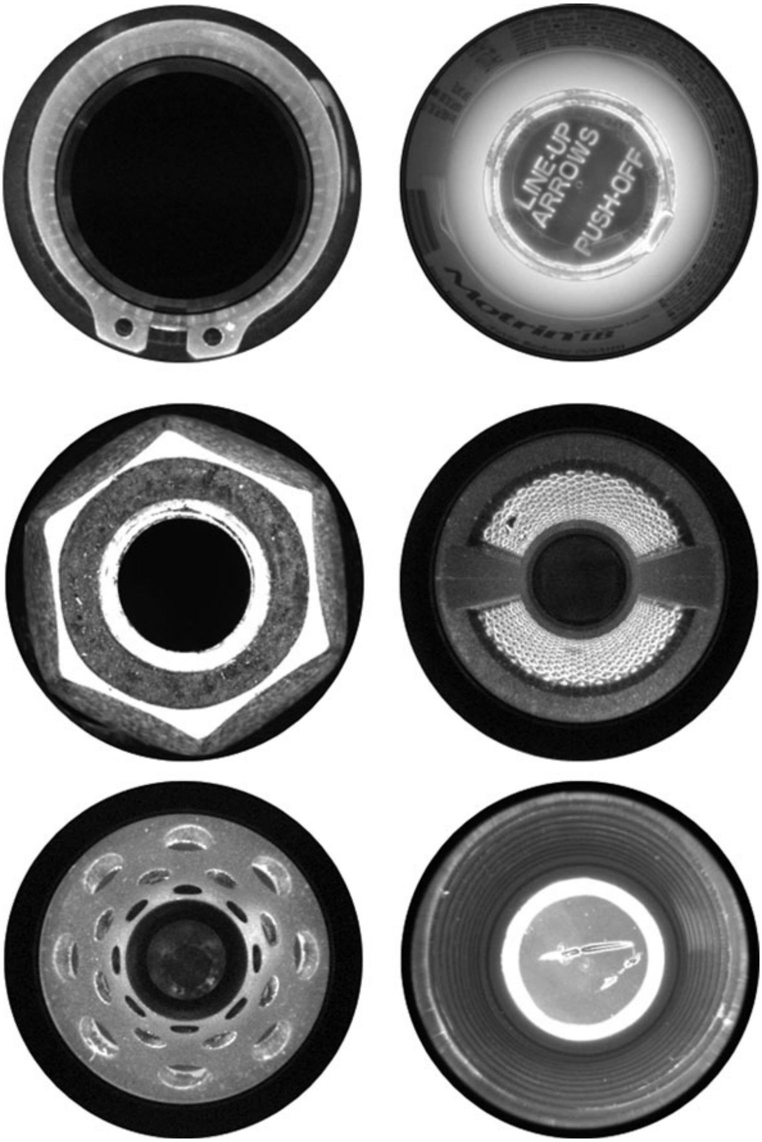


Fig. 6.12

Images obtained using a hypercentric lens. (a) Cube, gaming die. (b) Cylinder with six side ports, diameter approximately 16mm. (c) Screw head with a defect. (d) Car oil filter, viewed through a hole of approximately 19mm diameter. (e) Pipe, 35mm diameter, 100mm deep. (f) Medicine bottle. (g) Hexagonal nut (h) Cylindrical filter screen with a defect. (i) Cylinder with 24 side ports. (j) Pipe cap, internal diameter 25mm

Since they are flat, they occupy less space and are easier to mount than conventional curved mirrors. Unlike Fresnel lenses, they do not introduce any chromatic aberrration at all. The facets of a Fresnel mirror can be coated individually to construct a mirror that provides different deflection angles, or variable focal lengths, for different parts of the spectrum.

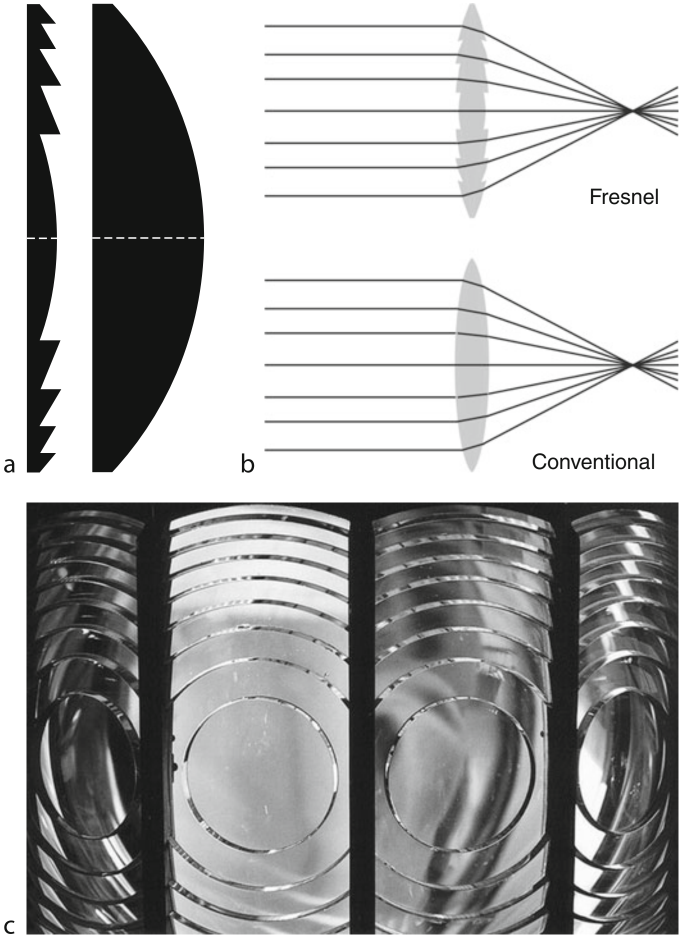


Fig. 6.13

Fresnel lens (a) Cross section, compared to that of a conventional lens. In most cases, the grooves are far smaller and more numerous than shown here. (b) Focussing in a Fresnel lens. Each groove acts a small prism, (c) Light-house Fresnel lens. Such a Fresnel lens may be a metre tall. An equivalent conventional lens would be extremely expensive and very heavy

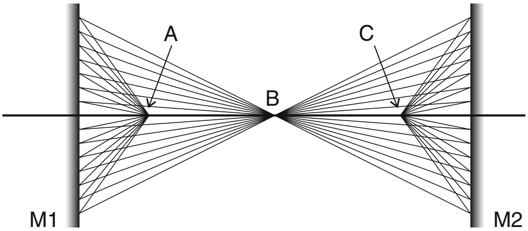


Fig. 6.14

Typical microlens structures. (a) The lenslet diameter is about 20μ. Closer packing and aspherical form are sometimes used. (b) Lenticular (1-dimensional) array

Fresnel lenses are used in domestic and commercial luminaires, traffic lights, maritime light-houses, navigation and automobile lights (➋ *Fig. 6.16*). They are used in overhead and video projectors, for image formation and are sometimes used to magnify the picture on small LCD displays. They are employed for light gathering, in devices such as solar cookers and furnaces. They are ideal in this type of application as a large low-cost Fresnel component can greatly increase the effective light-capture area of a photo-electric energy converter. Since they are light in weight and can be made to have considerable optical power, Fresnel lenses are suitable for making spectacles for people with certain vision defects including strabismus. Computer cameras, webcams and photocopiers all frequently employ them. Adjustable Fresnel

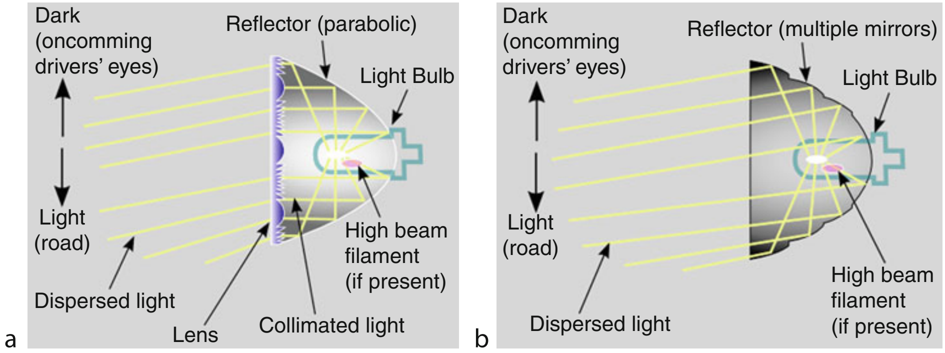


Fig. 6.15
Using two identical Fresnel mirrors (M1 and M2) to focus light from a "point" source (A), to another point (C). (For the sake of simplicity, the facets of the Fresnel reflectors are not shown.) An object placed at C will be illuminated at its side furthest from A. (Compare this to an elliptical mirror.) The distribution of light falling on point C is identical to that emitted by the source A. A pinhole placed at B will block most of the "stray" light coming directly from A (Compare this to a telecentric lens)

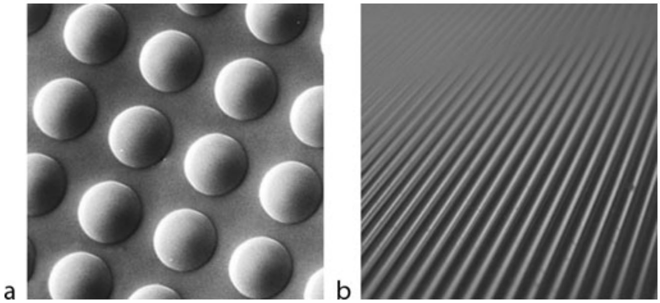


Fig. 6.16
Automobile headlamp designs. (European standard) (a) Fresnel lens. (b) Multi-facet Fresnel mirror. Reproduced under the GNU Free Documentation License, from [Headlamp, Wikipedia, URL http://en.wikipedia.org/wiki/Headlamp, Accessed 27th October 2010]

lenses have been made using piezo-electric crystals to control the focal length of the lens. The lens is heated when making the adjustment and then cooled to room temperature for use. Since very large Fresnel lenses can be made without incurring enormous cost, they are frequently used as objective elements in a telecentric-lens assemblies. (The objective element must be at least as large as the object being viewed.)

6.3.2 Microlens Arrays

A *microlens array* (also referred to as *microlenticular array* or *lenslet array*) is similar to a Fresnel lens, except that its surface consists of numerous tiny lenses, rather than prisms (➋ *Fig. 6.14*). The size, profile and transmission characteristics of these lenslets may vary across the array. Low quality arrays can be manufactured cheaply, by moulding materials such as Polymethyl Methacrylate (acrylic resin) or epoxy. Microlens arrays are available in sheet form. More precise arrays are constructed from high-grade optical materials, such as fused silica, using standard semiconductor manufacturing methods. This provides good uniform transmission over a wide range of wavelengths (0.19–3.0 μm) Silicon is sometimes used instead and transmits well over the NIR waveband (1.2–5μm). The shape, position and alignment of each individual lenslet can be controlled within close limits. Other commonly used materials are Gallium Phosphide and Calcium Fluoride. Appropriate optical coatings can be applied to control the transmission over selected ranges of wavelength. *Graded index* (*GRIN*) arrays depend not on the 3D shape of the lenslets but variations of the refractive index within a plate that has smooth parallel sides. M*icro-Fresnel lenses* are devices in which individual lenslets have Fresnel refracting surfaces. They can be made very thin and light in weight.

Single microlenses are used in communications systems, to couple light to individual optical fibres, LEDs and lasers. Arrays of microlenses can form excellent diffusers, providing a way to homogenise beams precisely and efficiently from a variety of sources including high-power LEDs and lasers, LED arrays and fibre-optic illumination bundles (➋ *Fig. 6.17*).

Microlens arrays are used in some digital projectors, to focus light to the active areas of an LCD spatial light-modulator. Combinations of microlens arrays have been designed that have novel imaging properties, such as the ability to form an image at unit magnification without inverting it. This is not nearly so convenient with conventional lenses. Microlens arrays are used in photocopiers and mobile-phone cameras. They can form several "copies" of an image, acting as a multi-way beam-splitter. Of course, they can act in the reverse direction, enabling two or more beams to be merged by interlacing them. Interlacing two images can be achieved with an array of alternating transmitting and opaque strips and is useful in 3D stereoscopy. These strips can be replaced by an array of cylindrical microlenses, forming what is known as a *lenticular screen* that is able to make efficient use of the illumination. A microlens array can provide variable magnification over its field of view. Such a device provides the basis for building a variable-resolution sensor akin to the eye and can be used in a projector to generate complex patterns of light from a simple "point" source.

Microlens arrays are used to increase the sensitivity of CCD image sensors, particularly interline-transfer devices, that suffer from reduced aperture due to metal shielding. One lenslet is placed over each photodetector, where it can collect and focus light that would otherwise have fallen on non-sensitive areas of the detector array (➋ *Fig. 6.18*). This increases its sensitivity, by making better use of the available light. The optical fill-factor can be improved significantly; the optical fill-factor of interline CCDs can be increased from below 20% to

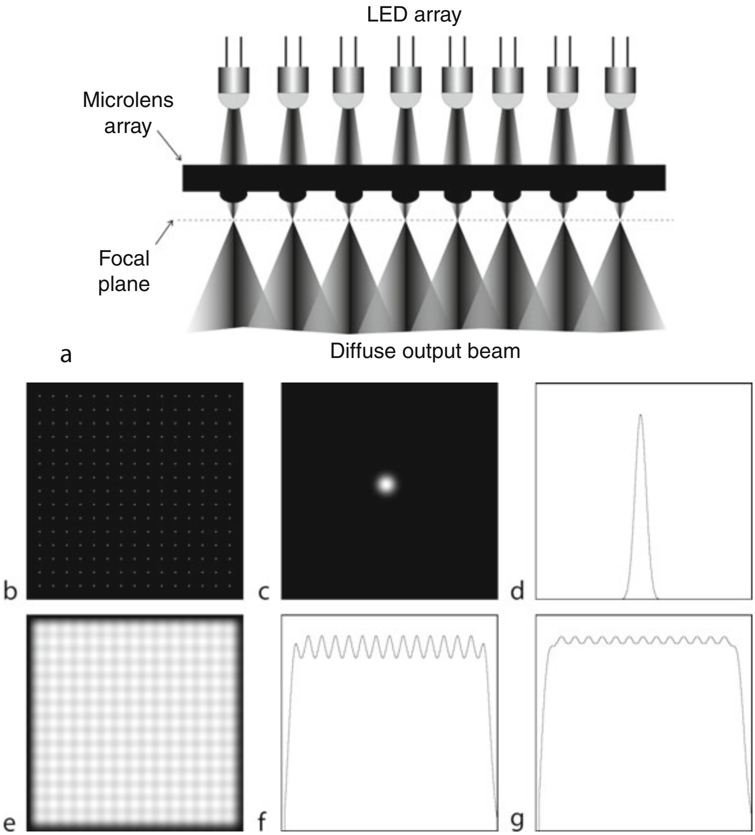


Fig. 6.17

Using a microlens array as a diffuser. (a) Layout, using one lenslet for each LED. The LED beams overlap beyond the microlens focal plane, thereby reducing local variations in the brightness of the illumination field. (b) Test pattern consisting of a series of "point sources". (The spots have been slightly dilated, for the sake of clarity.) (c) 2d microlens point-spread function. (d) 1D microlens point-spread function. (e) Illumination pattern. (f) Intensity profile across the centre of the illummination pattern. (g) Making the point-spread function wider (33%), reduces local variations in intensity (These are simulated results)

nearly 100% by placing a lenslet in front of each photosite. Moreover, the microlenses reduce the amount of stray light that might otherwise spoil the image quality. By controlling the incoming light better, fewer photons "spill over" into a neighbouring photo-receptor, to blur the image. The resemblance of the arrangement shown in ❯ *Fig. 6.18c* to an insect eye is obvious. One drawback of such an arrangement is that large variations in detector illumination angle can produce large variations in received image brightness.

The quality of the lenlets that can be fabricated using modern electronic manufacturing techniques (photolithography) is comparable to that of a standard ground lens. The diameter of an individual lenslet might be anywhere in the range 10–800 μm. There is no reason why all of the lenslets in an array should be all be same size, or have their optical axes all parallel to one another. Round lenslets may be packed hexagonally, while square lenslets can be packed more

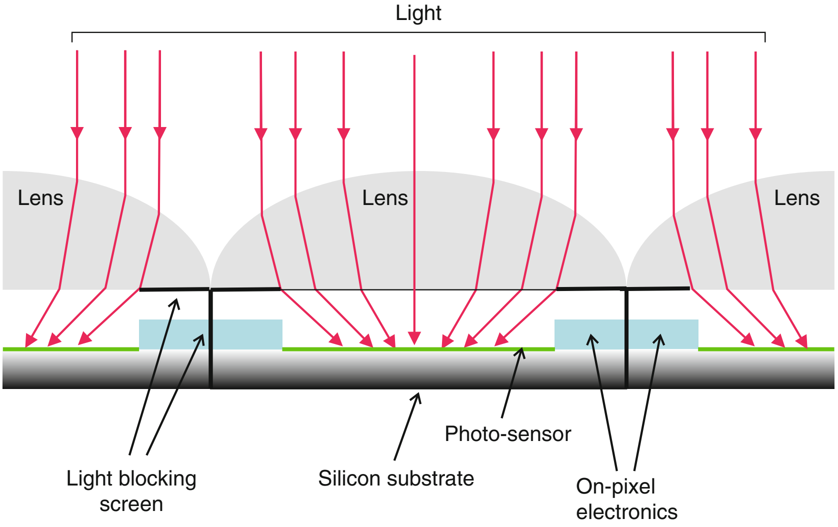


Fig. 6.18
Microlens array placed over a CCD image sensor. One lenslet is fitted to each photosensor to concentrate light on the light-sensitive area and away from the other electronic elements. Without a microlens, the detector would collect a significantly lower proportion of incoming photons. As a result, the fill factor is greater than it would be without the lens

closely together. When the lenslets are small than about 10 microns diffraction effects can become troublesome. The microlens array discussed so far rely on refraction. There is another type of device in which diffraction is deliberately exploited and these are discussed in the next section.

6.3.3 Binary Optics (Diffractive Optics)

Binary optics focus light by diffraction, not refraction. For ease of manufacturing, they are made with stepped, or multilevel micro-groove surfaces. Familar examples of the type of surface structure to be found on diffractive optics are to be found on CDs, DVDs, butterfly wings, fish scales and beetles. Binary optical devices have a range of specialised applications, principally being used in conjunction with coherent optical (laser) systems:

- Homogenising laser beam intensity
- Randomising beam phase
- Beam Splitting
- Kaleidoscope lens, creating several identical images from one image.
- Beam shaping
- Generating illumination patterns, such as gratings, cross hairs, multiple stripes, multiple spots, concentric circles, etc.
- Correcting for aberration in combination with refractive lenses
- Building Fresnel zone lenses
- Generating wavefronts of given shape
- Colour-selective filtering

6.4 Conclusions

The conventional optical devices described in the previous chapter are well known. The ability of telecentric lenses to produce images without perspective error, or parallax, has solved many critical inspection applications, especially those involving gauging. Controlling the direction of view is important but so is the ability to direct light at given angles onto an object. Again, telecentric lenses are able to help. Hypercentric lenses can produce a conical viewing/lighting volume., which enables them to see features that would otherwise require physical movement, multiple cameras, or a complex arrangement of beam-splitters and mirrors. The physical size of both telecentric and hypercentric lenses is limited by that of the front refracting element. Very large Fresnel lenses can be made at a modest cost, thereby enabling the benefits of telecentric and hypercentric lenses to be enjoyed over a much wider range of applications than would be the case with conventional lenses. A technology similar to that used for manufacturing Fresnel lenses is able to produce interesting and useful effects in microlens arrays. A wide variety of transformations is possible, including: field splitting, merging multiple beams to form a single image, polar-to-Cartesian axis mapping, correcting barrel or pin-cushion warping, etc. By making the features on the lens even smaller, diffractive effects dominate, enabling yet more interesting functions to be performed optically. It is well to remember that any operation that can be accomplished using optics is very much faster than the equivalent process performed in software, or even dedicated electronic hardware.

Further Reading

Advanced Imaging, URL http://directory.advancedimagingpro.com. Accessed 12th July 2009

Answers, URL http://www.answers.com/topic/telecentric-lens. Accessed 12th July 2009

Australia, URL http://azooptics.com. Accessed 12th July 2009

AZoOptics, Mona Vale, Accessed 12th July 2009

Coherent, URL http://www.coherent.com/Lasers/. Accessed 27th October 2010

Computer Optics, Inc, Hudson NH, USA, http://www.computeroptics.com/telecentric.html Accessed on 12th July 2009

Edmund Optics, York UK, URL http://www.edmundoptics.com/. Accessed on12th July 2009

Global Spec, URLhttp://www.globalspec.com/reference/7760/. Accessed on12th July 2009

EHD imaging GmbH,Damme/Germany, URL http://www.ehd.de/products/objectives/telecentric-lenses.htm. Accessed on12th July 2009

Goyo Optical, Inc., SAITAMA, JAPAN, URL http://www.goyooptical.com/products/telecentric/. Accessed on 12th July 2009

Graftek Imaging, Inc., Austin, Texas, USA, URL http://www.graftek.com. Accessed on12th July 2009

Hyper-Eye^{TM}, Light Works, http://www.lw4u.com/. Accessed 27th October 2010

ImvEurope, URL http://www.imveurope.com/products/product_details.php?product_id=508. Accessed 12th July 2009

Lambda Photometrics plc, Harpenden, UK, URL http://www.lambdaphoto.co.uk/products/122.300 Accessed 12th July 2009

Light Works LLC, Toledo, Ohio, USA, URL http://www.lw4u.com/ Accessed 12th July 2009

Machine Design, URL http://machinedesign.com/article/telecentric-illumination-for-vision-system-backlighting-0602. Accessed 12th July 2009

Test & Measurement World, Waltham, MA, USA, URL http://www.tmworld.com/article/CA6473121.html. Accessed 12th July 2009

Multipix, Petersfield, UK, URL, http://www.multipix.com/viewproduct.php?pid=177&cid=739. Accessed 12th July 2009

Nanjing AILI Optics & Electronics Co., Ltd., Nanjing, P.R.China, URL http://www.ailioptics.com/html/enlarging_projection_lens.htm Accessed 12th July 2009

Navitar, Inc, Rochester NY, USA, URL, http://machinevision.navitar.com/ Accessed 12th July 2009

Opto Engineering, URL http://www.opto-engineering.com/telecentric-lenses.php

PatentStorm,Washington, DC, USA, URL http://www.patentstorm.us/patents/6324016/description.html. Accessed 12th July 2009

Simanek DE, URL http://www.lhup.edu/~dsimanek/3d/telecent.htm. Accessed 12th July 2009

Voelkel R, Weible KJ (2010) Laser beam homogenizing: limitations and constraints, URL http://www.suss.com/fileadmin/files/technical_publications/WP_LaserBeamHomogenizing_0809.pdf Accessed 27th October 2010

Wikipedia, URL http://en.wikipedia.org/wiki/Telecentric_lens. Accessed 12th July 2009

7 Illumination Sources

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_7,

Abstract: Our everyday experience does not alert us to the need to choose lamps for Machine Vision carefully. The human eye-brain complex is able to compensate for slow variations in both the colour and intensity of ambient lighting, without our ever becoming aware that any change has taken place. As a result, people often remark with surprise how bright the room suddenly appears when a light bulb has been changed. Unlike people, machines are very sensitive to variations in both colour and brightness. Incandescent filament bulbs and fluorescent tubes are used in the home and office, while mercury and sodium discharge lamps are commonly employed in street lighting and in factories and warehouses. For many years, advertising has relied heavily on "neon" displays. In addition, there is a new variety of lamps available, including a wide range of LED-based lighting units, cold cathode tubes, compact fluorescent bulbs and plasma panels. Arc lamps have been used in specialised scientific and industrial applications. For Machine Vision, the following properties of lamps are important:

- 3D radiation pattern
- Ability to control the lamp brightness
- Brightness, including long-term variations due to lamp ageing
- Driver (AC /DC, frequency, closed-loop control)
- Efficiency, measured by the ratio of the optical emission power to the electrical power consumed
- Electrical noise generated
- Heat produced
- Infra-red content of the beam
- Life-time, measured simply by how long before the lamp becomes so dim as to be useless, or stops emitting altogether
- Long-term degradation of emitted power
- Power-line ripple
- Shape
- Short-tem stability of emitted power
- Spectrum (This requires a much more detailed analysis than simply stating the "colour".)
- Ultra-violet content

The features of the major types of lamp are discussed. LEDs are almost ideal for many industrial applications and in recent years, there have been major advances in the design of LED arrays for Machine Vision. It is possible to build arrays in a variety of shapes. An array may contain LEDs of many different and precisely known colours. LEDs are efficient and stable over a long period. The lighting pattern can be adjusted dynamically, under software control.

7.1 Introduction

Success in Machine Vision begins with good lighting! A well-designed illumination sub-system enhances the features being inspected, whereas a poor one will obscure them. Here is a universal truth: successful Machine Vision systems do not rely on environmental lighting. To build a reliable Machine Vision System, the lighting must be specially designed for each application. Simply pointing the first available light source, such as a hand-held torch or a desk light, at the object to be inspected is simply not good enough. (This comment does not, in any way, diminish the case for experimenting with lighting during prototype development.)

A poorly designed illumination system may produce glare (thereby saturating the camera), shadows (hiding defects or other important features), and/or low contrast images. Uneven illumination forces the vision engineer to use image processing algorithms that are slower, more expensive and less reliable than they need be. It always pays to optimise the lighting before proceeding with the rest of the design.

7.1.1 Measuring Light

It is important, first, to distinguish between photometric and radiometric measurements of light.

- *Radiometry* is concerned with the physical measurement of electromagnetic radiation within the range of wavelengths 10^{-2} to 10^{3} μm (i.e., frequency range 3×10^{11} to 3×10^{16} Hz). This includes ultraviolet (UV), visible (VIS) and infrared (IR) wavebands.
- *Photometry* is concerned with the measurement of light as perceived by the human eye (wavelength range 0.360–0.830 μm).

Photometric and radiometric concepts are similar, except that for the former, emission/reflection spectra are weighted by a luminosity function representing the spectral response of the eye of a standard observer (➔ *Fig. 7.1*) [1]. If the radiometric spectrum is represented by R(λ), the photometric spectrum by P(λ) and the standard luminosity function by L(λ), then

$$P(\lambda) = R(\lambda).\ L(\lambda)$$

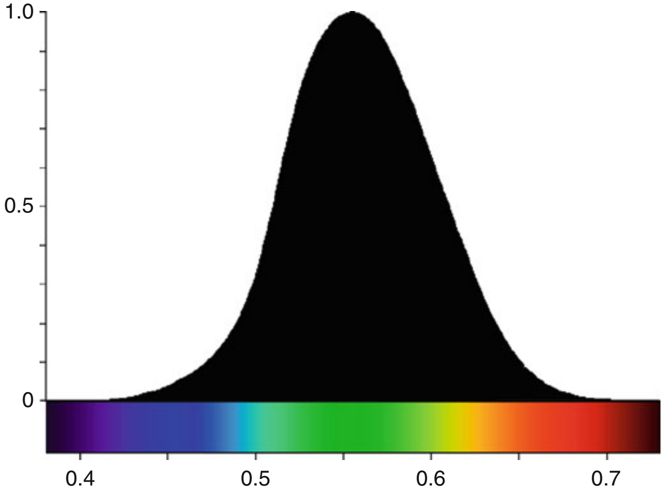


Fig. 7.1
The 1931 CIE standard luminosity function describing the average sensitivity of the human eye to different wavelengths. The photopic curve, shown here, relates to day-time light levels and is the basis for the definition of the 1931 CIE Colour Space. It can be used to convert radiometric spectra into photometric spectra. For low light levels (scotopic vision), the luminosity function has a similar shape but the peak is shifted towards the shorter wavelengths

$$P(\lambda) = R(\lambda) \cdot L(\lambda) \tag{7.1}$$

We are now able to define several measurements of light. Also see ➲ *Table 7.1*.

- *Candela* (cd) is the luminous intensity, in a given direction, of a source that emits monochromatic radiation of wavelength 0.555 μm (mid-green, frequency 5.40 × 10^{14} Hz) and has a radiant intensity in that direction of 1/683 W per steradian. (A common candle emits roughly 1 cd and a 100 W incandescent lamp emits about 120 cd.)
- *Lumen* (lm) is the SI unit of luminous flux. If a light source has a luminous intensity of 1.0 cd, the total luminous flux emitted into a solid angle of one steradian, the total luminous flux emitted is 1 lm. An isotropic 1 cd source emits a total luminous flux of 4π lumens.
- *Lux* (lx) is the SI unit of illuminance and measures the intensity of light weighted according to the *Luminosity Function* (➲ *Fig. 7.1*). The illuminance levels commonly encountered in everyday life are summarised in ➲ *Table 7.2*.

Table 7.1
Units of illumination

Quantity	SI unit	Abbreviation
Luminous intensity (SI base unit)	candela	cd
Luminous energy	lumen second	lm s
Luminous flux/luminous power	lumen (= cd sr)	lm
Luminance	candela per square metre	cd m^{-2}
Illuminance (light incident on a surface)	lux	lx = lm m^{-2}
Luminous emittance (light emitted from a surface)	lux	lx = lm m^{-2}
Luminous efficacy of a lamp (maximum is 683.002)	lumen per watt	lm W^{-1}

Table 7.2
Relating photometric illuminance to everyday experience

Illuminance (lux)	Example
5×10^{-5}	Starlight
10^{-3}	Moonless clear night sky (no city lights)
10^{-2}	Quarter Moon
2.5×10^{-1}	Full Moon, clear sky
10	Candle, 300 mm away
5×10^{1}	Household living space
4×10^{2}	Brightly lit office
10^{3}	Television studio lighting
3×10^{4}–10^{5}	Sunlight, average day

Colour rendering index (CRI) is a measure of how well balanced the different spectral components of nominally white light are. A black body radiator, such as an incandescent lamp has a CRI of nearly 100, while fluorescent tubes have CRIs of 50–99%. Colour temperature is another measure of the whiteness of a light source. Typical incandescent lighting has a colour temperature of 2,700 K (yellowish-white). Quartz halogen lighting is 3,000 K. “Warm-white” fluorescent lamps have a colour temperature of 2,700 K and are therefore widely accepted for domestic lighting. “Daylight” fluorescent lamps have a colour temperature of 5,000–6,500 K, which appears bluish-white.

7.2 Light Sources and Their Features

When considering a new application, the vision engineer must first decide what type of light source is most appropriate. This requires that we consider a broad range of features, not just how much light it emits.

7.2.1 General Remarks

Several types of light source are commonly used in Machine Vision and will be reviewed in this chapter:

- Incandescent filament lamps
- Fluorescent lamps (linear)
- Compact fluorescent lamps (These are simply “folded” fluorescent tubes.)
- Discharge lamps
- Light Emitting Diodes (LEDs)

In this chapter, we review these four major families of light source. Before then, we highlight some of the features that are likely to affect the choice. (For a summary of the salient features, see ➋ *Table 7.3*.)

Coherent and non-coherent light Human beings live in a world flooded with non-coherent light and normally encounter coherent light only when working with lasers. The light emitted by the Sun, flames, incandescent filament lamps, fluorescent tubes and other discharge lamps are non-coherent, as are computer screens and LEDs. On the other hand, lasers are able to generate almost pure monochromatic beams in which all rays are in phase. It is possible to produce very narrow and very intense beams from a laser. (The total light flux may, however, be quite small.) While phase coherence is important for holography, it normally provides little benefit for Machine Vision. In structured-light range finders, it is not an essential requirement; it simply has to be tolerated. Speckle patterns can be produced by projecting a phase coherent beam onto a surface. This will create a textured effect, even on a plain surface and may give a false impression of its physical structure and visual appearance with ordinary lighting. Speckle effectively produces a random noise-like effect and hence complicates the image processing required.

Imaging Optics Many light sources are large compared to the objects they are used to illuminate. This fact must be taken into account when designing the illumination optics. The latter are just as important as the imaging optics but are often neglected when designing the optical sub-system. The application may require a “point” or “line” source, while the light is generated by a large lamp, probably producing a completely different beam shape from

Table 7.3

Factors influencing the choice of light source for machine vision

Main feature	Category	Factors to be considered
Light characteristics	Intensity	Overall brightness
	Spectrum	Emulating day-light (black body source)
		Monochromatic
		Multiple peaks
		Smooth panchromatic with/without spiky peaks
		Ultra-violet
		Infra-red
	Beam shape	Omni-directional
		Directed
		Linear
		Rectangular
		Circular
		Beam uniformity
	Coherence	Coherent (Lasers)
		Non-coherent (Other sources)
	Temporal variation of intensity	Ageing
		Strobing
		Light fluctuating with AC power cycle
		Start-up
Use, implementation and operation	Cost	Installation
		Running
		Replacement
		Special electronic driver
	Electrical	Voltage
		Power
		AC or DC?
	Physical	Size of luminaire
		Maintenance (cleaning)
		Heat generated
	Sensitive operating environment	Operating temperature
		Vibration
Other	Failure mode	Slow degradation
		Sudden catastrophic failure
	Safety	Toxic chemicals (particularly mercury)
		Flashing lights (migraine & epilepsy)
		Broken glass
		Electric shock
		Eye & skin damage

that required. The apparent size of a light source can be reduced by the use of focussing optics. Notice, however, that the focussed image of the source limits the minimum size of the light spot. Light can be collected from the lamp and then focussed by a combination of lenses, mirrors and fibre optic devices. So called *condenser lenses* are designed specifically for this purpose. Since these have to be at least as large as the lamp, it may be necessary to use Fresnel (prismatic) lenses. An example of lenses of this type is to be found in maritime warning lights (light-houses). Sometimes, a concave mirror, with a parabolic, elliptical or similar form, is used as the primary light collector. Examples of this type are, of course, found in automobile head-lights, navigation lights, projectors and photographic enlargers. Some lamps incorporate their own light collecting/focussing optics: small quartz halogen lamps, used for spot-lights in retail displays, often incorporate a dichroic filter. The filter focusses visible light into a narrow beam but does not focus infra-red, which is therefore dispersed over a much wider solid angle.

Stray light Most light sources, except some lasers and LEDs, emit radiation over a large solid angle. Hence, it may not be possible to collect all of the light emitted by the lamp and project it in the correct direction. We must be careful to stop stray light from reaching the object being inspected. If stray light is allowed to illuminate the scene being viewed from the wrong direction, the image contrast light may be greatly reduced, perhaps disastrously. Stray light and ambient light must both be regarded as creating optical “noise” in the optical system and should be deflected, or blocked by baffles with light-absorbent surfaces.

Ambient light For a similar reason, ambient light should be blocked by opaque screens, or curtains. In some rare situations, it may be possible to project so much light onto the object being inspected that environmental lighting is insignificant. However, it is always good practice to eliminate ambient light by shrouding the inspection area, if only to reduce energy costs.

Keeping optics clean To ensure reliable performance, it is imperative that all optical surfaces be kept clean. This may not be easy for a large source working in a dirty atmosphere. Baffles used to block stray/ambient light make the luminaire larger and more complicated and may inhibit cleaning. Where possible, the whole illumination sub-system should be enclosed in a dust-proof protective box. A regular cleaning regime should also be established. If total enclosure is not possible, regular cleaning of all optical surfaces is vital.

Heat Infra-red and waste heat can damage some products/materials, such as chocolate. Many light sources require significant amounts of electrical power and then waste much of it by generating large amounts of heat. Incandescent filament lamps generate a great deal of waste heat, which is carried away by conduction and convection. The end caps of fluorescent and other discharge lamps can become quite hot. Some high-power, multi-LED assemblies generate enough heat to require cooling fins. Special precautions may be needed, in some circumstances, to remove waste heat and/or eliminate IR radiation from the beam. The light source may need no more than simple convection cooling. On other occasions, a fan might be needed. In really difficult situations, a circulating liquid coolant can be used. It may be necessary to block infra-red radiation using a hot mirror, or an IR-blocking filter.

Beam Profile The emission profile for many lamps is irregular. This is not surprising in the case of incandescent lamps whose filaments have complex shapes (e.g., C- or W-shaped). Moreover, the support wires cool the filament locally, causing dark spots (➋ *Fig. 7.2*). Linear fluorescent tubes do not emit uniformly along their whole length, particularly near the ends, which are usually darker than elsewhere. Compact fluorescent lamps often have convoluted tubes. (➋ *Figure 7.3* shows a tube in the form of a helix. Other lamps of this type have the form of a series of 2, 3 or more U-shaped tubes.) LED devices may be far from simple “point” sources; there may be several separate LEDs within a single housing. Whatever their origin,

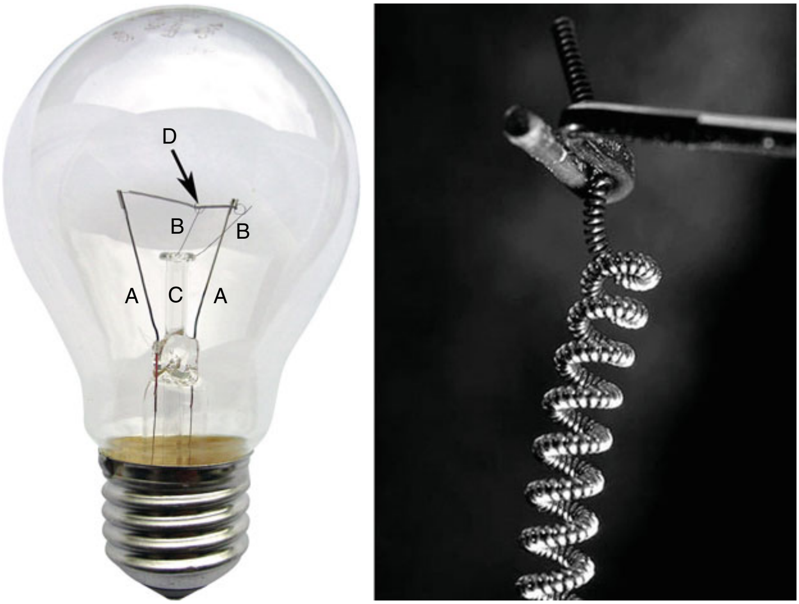


Fig. 7.2
Incandescent lamp filaments. (a) Structure of the traditional light bulb, which is now ''outlawed'' in many countries [2]. A – current-carrying wires. B – support wires. C – glass support column. D – local cold spot, due to conduction of heat away from the filament by the support wire. The shape of the filament leads to an uneven light distribution. A variety of other filament forms is illustrated in [3] (b) Coiled-coil filament and support wire in detail. The coiled-coil allows a very long wire to be used in a compact space (up to 2 m for 60 W lamp). Variable spacing of either the primary or secondary helix leads to uneven temperature distribution and uneven light output

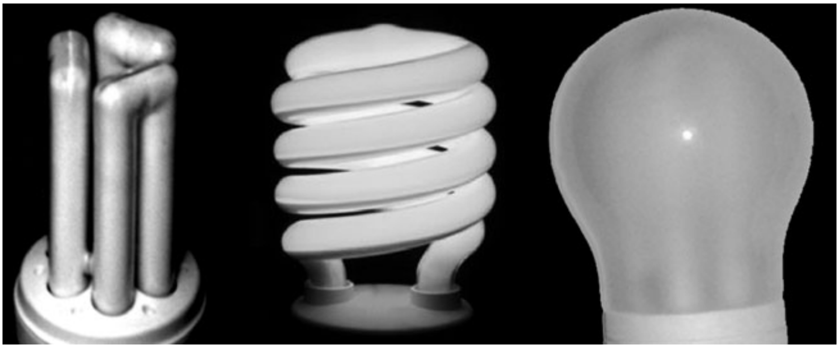


Fig. 7.3
Compact fluorescent lamps used for ambient lighting. In each case, the physical structure of the lamp causes uneven distribution of the emitted light. The internal structure of the lamp on the right is just visible. This consists of a folded tube inside a ''spherical'' diffuser

irregularities in the lamp emission profile have to be reduced to a minimum, using specially designed illumination optics and diffusers. The latter may be simple embossed glass/polycarbonate sheeting, thin ceramic plates, stippled metal reflectors, or for superior performance diffractive optics.

Lamp Ageing and Control The change in the light output from a lamp during its working life is a major factor which the designer of a vision system must always bear in mind. The brightness of a standard incandescent lamps falls as tungsten that has evaporated from the hot filament condenses on the inside of the glass bulb. It is not uncommon for the light output to fall by over 50% during a bulb's working life. Quartz halogen lamps overcome this problem to some extent, by redepositing metallic tungsten on the filament. (Tungsten halide forms in the cooler part of the bulb and then decomposes when it reaches the hot filament.) Fluorescent tubes also darken as they get older, as anyone who has ever replaced a tube at home will have noticed. Again a 50% reduction in the light output over the life-time of the tube is not unusual. LEDs also become darker as a result of ageing, although the change is not as great; a 30% reduction over a working life of 1,00,000 h is often quoted by suppliers.

Since the output light level changes significantly during the life of a lamp, it is often advisable to employ feedback control. Placing a photo-sensor in the periphery of the output beam would seem to be appropriate for maintaining a constant light output. The output of a differential amplifier would then be used to vary the drive current to the lamp. In principle, this seems straightforward but there are several points to bear in mind. Altering the current through an incandescent lamp changes the filament temperature and hence the colour of the light emitted. It is possible to adjust the brightness of fluorescent and discharge lamps using pulse-width modulation. This requires a fairly sophisticated controller but the colour of the light emitted does not change. Changing the current through LEDs does not alter their colour either A multi-LED assembly does not have an homogenous beam but the short-term variations among individual LEDs are not likely to be significant. Furthermore, the effect of any one LED is small in a large array. However the light is generated, it is a good idea to check the beam profile of multi-lamp sources regularly, using the camera and a plain calibration surface.

Efficacy Efficacy measures the ratio of the total light output for a lamp to the electrical power that it requires. The efficacy of various light sources are compared in ➔ Table 7.4.

Table 7.4

Comparing the efficacy of various light sources [4]

		Percent of input energy			
Type of lamp	Efficacy (lmW^{-1})	UV	Visible	IR	Heat
Incandescent filament	12.6	0.25	5.75	75	19
Xenon (2 kW)	35	2	15	58	25
Low pressure sodium	125	0.5	22	36	41.5
High pressure mercury	49	9	11	52	28
Low pressure mercury	6.4	56	5	15	24
Fluorescent (linear tube)	62	2.4	22	30	45.6
LED (white, space lighting)	26–70	0	26–70	0	

The values given for low pressure mercury lamps does not take into account the possibility of converting the UV radiation to visible light by a phosphor coating

7.3 Incandescent Filament Lamps

When a body is sufficiently hot, it radiates in the visible wave-band (➋ *Fig. 7.4*). This fact has been exploited for centuries to generate light. Material can be heated in a flame (e.g., lime lights used in nineteenth century theatres), chemically (e.g., by combustion and fire-flies), or by ohmic resistance to an electric current. (Eddy current, microwave and radiation heating are also possible ways to raise the temperature of a material to incandescence.)

In a domestic electric light bulb, an electric current passes through a thin tungsten wire raising its temperature (typically around 2,500–2,800 K). This causes it to release thermally generated photons. The enclosing glass bulb simply prevents air from reaching the hot filament, which would otherwise be quickly destroyed by oxidation. The bulb is normally filled with an inert gas (nitrogen, argon, or krypton), to inhibit evaporation of tungsten from the filament and to reduce the required strength of the glass envelope. Incandescent bulbs can be produced cheaply, for a wide range of operating voltages, ranging from one to several hundred volts.

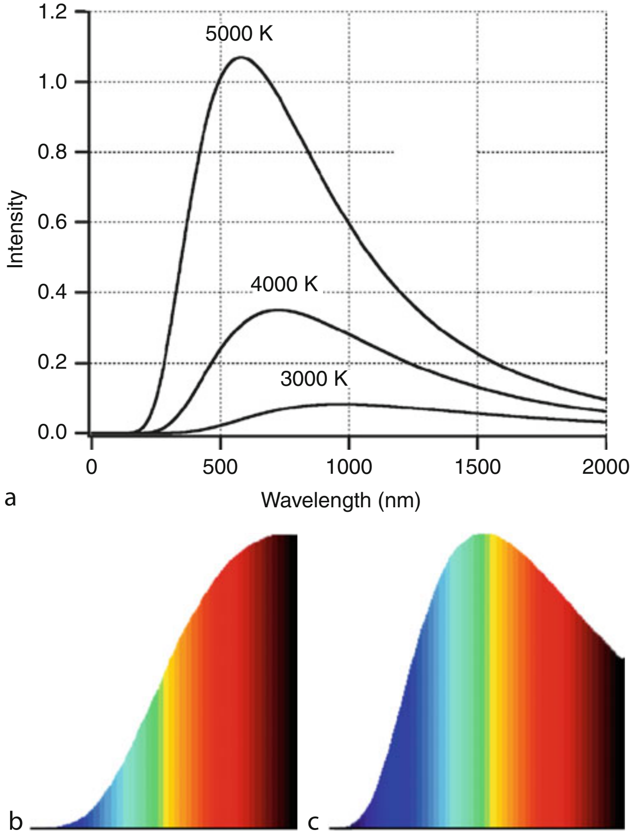


Fig. 7.4

Radiometric black body emission spectra. (a) Graphs for three different source temperatures. Abscissa: Wavelength (nm). Ordinate, intensity, arbitrary units. (b) Representing the same data in relation to perceived colour. Source temperature 3,000 K (about the same as that of a lamp filament). (c) Source temperature 5,000 K. The colour images were generated by a program in [5]

7.3.1 Quartz-Halogen Lamp

In a conventional lamp (not quartz halogen), the evaporated tungsten gradually condenses on the inner surface of the glass envelope, reducing light transmission. A vacuum-filled bulb darkens uniformly over the whole surface of the envelope, while a gas-filled bulb darkens on its uppermost part. A major development in the design of incandescent lamps resulted in improved life and greater efficiency. In a quartz halogen lamp, the tungsten filament is sealed into a small quartz envelope filled with a halogen gas: iodine or bromine. When tungsten evaporates from the filament, it combines with the gas to form a tungsten halide (Wo_2I_2 or Wo_2B_2), which does not react with the quartz envelope. However, tungsten halide does decompose on contact with the hot filament, depositing metallic tungsten preferentially on hot spots. The halide atoms are then free to capture further tungsten atoms as they boil off the filament. This is a self-correcting process, in which thin hot regions of the filament receive more tungsten atoms than thicker cool areas. Quartz must be used for the envelope, since ordinary glasses would melt at the higher operating temperature of this type of lamp. The high filament temperature increases the efficiency of energy conversion, apparent brightness and colour temperature. Raising the colour temperature increases the short-wave (blue) content of the emitted light.

The envelope can be modified by applying an optical coating, which might enhance, or inhibit, UV transmission. Thus, quartz halogen lamps can be made to generate visible light or UV-B radiation. A typical quartz halogen lamp will run for about 2,000 h, which is about twice as long as the life of a conventional filament lamp.

A further development that has improved quartz halogen lamp efficiency is an IR-reflective coating (IRC). The quartz envelope is coated with a multi-layered dichroic coating, which allows VIS light to be emitted and reflects some of the IR radiation back onto the filament. This type of lamp requires less power than standard halogen lamps to produce a given light output. The efficiency increase can be as much as 40%.

7.3.2 Analysing Perfomance

To a close approximation, a filament lamp behaves like a black body radiator. Hence, it is possible to represent its behaviour using the following equations [6–8].

Wien's Law (Explains the colour shift with varying filament temperature see ➲ Fig. 7.4).

$$\lambda_p = 2.898 \times 10^{-3}/T \tag{7.2}$$

Stefan's Law (Quantifies the total radiated power)

$$P = \sigma \,.\, A \,.\, T^4 \tag{7.3}$$

Planck Energy Distribution (Describes the spectrum for incandescent sources)

$$P_\lambda = \frac{2\pi hc^2}{\lambda^5\left(e^{(hc/\lambda kT)} - 1\right)} \tag{7.4}$$

where

A = Surface area of radiating body (i.e., the filament, m^2)
c = Speed of light in a vacuum, 3×10^8 ms^{-1}
h = Planck's constant (6.626×10^{-34} Js)

Table 7.5

Showing how the characteristics of an incandescent light bulb depend on the supply voltage, V

Operating characteristic	Varies in proportion to (approximate formulae)
Light output	$V^{3.4}$
Power consumption	$V^{1.6}$
Working life	V^{-16} (i.e., inversely related)
Colour temperature	$V^{0.42}$

These relationships are only valid for small variations of voltage (i.e., a few percent) around the nominal operating conditions. Notice that the working life is greatly reduced, even if the operating temperature is increased by only a few percent [2]

k = Boltzmann's Constant, 1.38×10^{-23} JK^{-1}
P = Power radiated (W, or Js^{-1})
P_λ= Power per unit area at wavelength λ (Wm^{-2})
T = Temperature of radiating body (K)
λ = Wavelength (m)
λ_p = Wavelength at which the black body radiation is a maximum (m)
σ = Stefan's Constant, 5.67×10^{-8} $Wm^{-2}K^{-4}$

The electrical performance of an incandescent lamp is summarised in ➲ *Table 7.5*. Notice the very rapid decline in working life as the filament temperature is raised.

7.3.3 Remarks

1. One of the problems that the standard incandescent light bulb presents is evaporation of metal (tungsten) from the filament. Inevitably, there are variations in resistivity along the filament, that cause non-uniform heating; hot spots form in regions of increased resistance. Filament thinning, by evaporation, occurs more in hot spots, thereby increasing their resistance, which in turn raises the filament temperature. This positive feedback shortens the life of the filament. Filament lamps are designed by reaching a compromise between operating efficiency (requiring high filament temperature) and longevity (low temperature).
2. Incandescent bulbs for general illumination applications can normally be expected to operate for several hundred hours. This is far shorter that the life expectancy of fluorescent and LED lamps. Special purpose theatrical, photographic, and motion-picture lamps may have a useful life of only a few hours. A distinction must be made between the rated and economic life of lamps used for machine vision. The rated life of an incandescent bulb is normally taken to be that time in which 50% of the lamps fail catastrophically. Replacement bulbs like this are relatively cheap, whereas the unscheduled downtime of production plant due to catastrophic lamp failure can be very high.
3. Incandescent lamps are physically fragile, having a thin glass envelope. When the envelope breaks, a cloud of particles (tungsten nitride and various tungsten oxides) is ejected

into the atmosphere and may be deposited on nearby optical surfaces. These will then require cleaning.

4. Apart from the obvious hazards of explosion, electric shock, burns to the skin and fire, some quartz halogen lamps can cause "sunburn" from the high levels of UV emitted. Eye damage (cataracts) can also be caused by UV. Surface contamination, especially from fingerprints, can damage the quartz envelope and lead to the creation of localised hot spots on the bulb surface. These may cause the bulb to fail prematurely, or even explode! Consequently, these lamps should be handled without ever touching the quartz envelope with bare hands. If the quartz is contaminated by fingerprints, it should be thoroughly cleaned using alcohol and then allowed to dry before use. To prevent injury due to explosion or fire, it may be necessary to operate quartz halogen bulbs within a protective enclosure.
5. The filaments of incandescent lamps are susceptible to vibration (microphony), which may cause the light output to fluctuate over a short time interval. The support wires cool the hot filament, causing local variations in light emission (➋ *Fig. 7.2*). Many lamps have C-, U-, or W-shaped filaments that do not radiate uniformly in all directions. Others have nominally linear filaments, which again radiate more in some directions than others. The diffusing inner surface of the envelope helps matters but does not provide a perfect solution. While these features are normally unimportant for illuminating a room, they are often critical for Machine Vision.
6. An important feature of incandescent lamps is that they change colour when the filament temperature varies. (The discussion at the beginning of ➋ Sect. 7.3 and ➋ *Fig. 7.4* show why this is so.) This is important, because it affects the way that we can control the lamp brightness. As the current through a light bulb increases, it normally becomes brighter and generates more short-wavelength radiation (UV and blue). To maintain good colour constancy throughout the working life of a lamp, it is often better to use a variable aperture, rather than changing the current.
7. Most of the power consumed by an incandescent lamp is converted into heat, rather than visible light. A standard incandescent light bulb produces more heat and requires more electrical power) than a quartz halogen lamp, a fluorescent lamp, or an LED device. For this and other reasons, incandescent light bulbs are falling out of favour, for domestic/commercial lighting. (At the time of writing, there are advocates in many countries pressing for a legally enforceable ban on incandescent lamps for environmental lighting.) Compact fluorescent lamps, and LEDs, use far less energy for a given light output. In Machine Vision, incandescent light bulbs are losing popularity to LEDs, for reasons that will become clear later.
8. The resistance of the filament is temperature dependent; the resistance when cold is much lower than when the lamp is lit. This means that there is a current surge when the lamp is first switched on (This is, of course, the moment when the filament is most likely to fuse.).
9. Since an incandescent lamp presents a resistive load, a simple dimmer, using pulse-width modulation and a standard power switch, can be used to control its brightness.
10. Thermal inertia of the filament of most incandescent filament lamps reduces the variation in light output during the mains power cycle. (50/60 Hz) However, some lamps (e.g., low power, high voltage) have thin filaments and do not provide a thermal inertia sufficient to prevent detectable flicker in a non-synchronous camera.
11. The life of an incandescent lamp is very sensitive to changes in the supply voltage; increasing the voltage by 5% halves the expected working life! (➋ *Table 7.5*).

12. The spectrum of an incandescent lamp is roughly similar to that of sunlight. In particular, there are no strong spectral peaks that might cause colours to be changed dramatically.
13. Despite all of their disadvantages, incandescent lamps are still widely used. However, the situation is likely to change in the near future; compact fluorescent tubes and LEDs are expected to take over many of the roles, outside Machine Vision, that have been previously been the domain of fragile glass bulbs containing hot wires.

7.4 Gas Discharge and Fluorescent Lamps

7.4.1 Physical Principles

A simple discharge lamp has two electrodes placed at the opposite ends of a glass tube, which contains an inert gas (e.g., argon, neon, krypton) at low pressure (typically 10^{-3} to 10^{-2} bar). A potential of several hundred volts is applied to the two electrodes, in order to start and then sustain a glow discharge. Initially, a few of the atoms within the gas are ionised. (Some mechanism may be built into the tube to initiate the discharge. This may consist of a hot filament, a weak source of beta radiation, or a rare-earth coating on the cathode.) Ions are propelled along the tube by the applied potential: electrons travel towards the anode and the heavier positive ions towards the cathode. High-speed ions colliding with neutral gas atoms will ionise them, thereby creating secondary ions, which are then accelerated by the applied field. The primary and secondary ions will then accelerate and collide with and ionise yet more atoms, thereby creating a cascade effect in which a large proportion of the atoms become ionised. The ionised gas is called a *plasma* and is able to conduct an electric current. This state is maintained as long as a sufficiently high potential difference exists between the electrodes. When positively charged ions strike the metal cathode, free atoms are released and join the inert gas atoms in the plasma. (This process is known as *sputtering*.) Once free, the metal atoms released from the cathode are able to join in the process of formation of a plasma. This therefore contains a mixture of free electrons, inert gas ions, metal ions, excited neutral inert-gas atoms and excited neutral metal atoms Neutral atoms become excited during low-energy collision between ions and neutral atoms. Excited atoms lose energy fairly quickly, by a variety of processes. Of prime concern for light generation is the fact that some of the excited atoms relax to a low-energy state by emitting photons in the ultra violet (high energy), or visible (lower energy) wavebands.

It is important to realise that the discharge does not glow uniformly throughout the space between the electrodes. ➋ *Figure 7.5* shows the main features. Notice the presence of dark regions. In a long tube, the positive column may be striated. In a short tube, fewer stripes are formed and the positive column becomes compressed. It may even disappear completely. However, the negative glow remains the same size.

7.4.2 Variations

Discharge lamps are available in many different forms and find a wide range of applications. Here is a list of the more important types:

- Fluorescent tubes (room lighting)
- Cold cathode fluorescent lamps (CCFL)

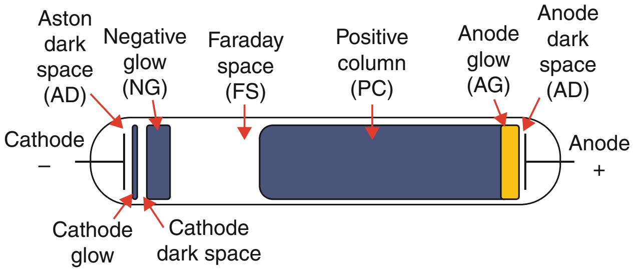


Fig. 7.5

Gas discharge lamp. The anode, cathode, Faraday and Aston spaces are all dark. Other regions emit visible and/or UV radiation

- Compact fluorescent (CFL)
- Mercury lamps (street lighting)
- Sodium lamps (street lighting)
- "Neon" display tubes (advertising signs)
- Plasma video displays
- Xenon flash tubes

The spectrum of the radiation generated by the plasma is determined by the composition of the gas and the cathode material. Adding a small amount of material to an inert gas can significantly alter the spectrum and the perceived colour of the emitted light. In a low pressure lamp, mercury or sodium is mixed with an inert gas, such as argon, neon, or a mixture of the two. The mixture is likely to contain 0.1–1.0% sodium/mercury vapor and 99.0–99.9% of the inert gas(es). Despite the low concentration of these additives, the emission spectrum is dominated by the energy transitions that occur as the metal atoms revert to their non-excited ("ground") state. These transitions involves well-defined energy jumps, and hence result in the emission of radiation within a series of narrow wavebands. At any given moment, most of the metal vapor atoms are non-excited and will readily re-absorb this radiation as they jump to the higher energy state that a neighbouring metal atom has just left. In order to reduce re-absorption, it is sensible to keep the concentration of the metal vapour low, compared to that of the inert gas. The concentration of the inert gas atoms controls the mean path length of free electrons travelled between collisions and hence is largely responsible for determining the electrical characteristics of the discharge.

7.4.3 Fluorescent Lamps

Fluorescent tubes are commonly used to illuminate rooms, corridors and industrial workstations (environmental lighting). Despite their familiarity, details of their operating characteristics are rarely appreciated.

A fluorescent lamp is a gas-discharge lamp that uses low-pressure mixture of ionised gaseous mercury and an inert gas (argon, xenon, krypton, neon, or a mixture of these). The UV radiation generated in this way is converted into visible light by a fluorescent coating on the inner surface of the tube. (It is usual to refer to this coating as a *phosphor*, although it is *fluorescence*, rather than phosphorescence, that generates most of the light.) A UV photon emitted by the plasma is absorbed by an atom within the coating, which then emits a long-wavelength (lower energy) VIS photon. If the UV photon has a higher energy than that needed to generate a VIS photon, the "excess" simply raises the temperature the phosphor coating.

Most of the UV energy generated by the mercury-inert gas plasma is at wavelengths of 0.254 µm (about 65%) and 0.185 µm (10–20%). The spectrum of the visible light generated by a fluorescent tube is determined by the fluorescent coating (➜ *Fig. 7.6*). By careful blending of various phosphor materials, it is possible to produce a fluorescent tube whose light produces approximately the same subjective sensation as day-light. This is not the same as saying that the two spectra are similar. In some cases, the output spectrum from a fluorescent tube has strong peaks, while in others, the spectrum is smooth. When viewed directly, or projected onto a white surface, the colour sensation produced by two tubes may appear to be very similar (e.g., approximating day-light) but quite different on a coloured surface (➜ *Fig. 7.6*). Many people find the light produced by some fluorescent lamps harsh and unpleasant and prefer the "warmer" (redder) glow from an incandescent filament lamp as being more relaxing. A healthy person may appear to have an unnatural and unhealthy skin tone when he/she sits beneath a fluorescent lamp. Modern "daylight" fluorescent tubes have, to a large extent overcome these objections to their use. When she is buying a dress, the author's wife still

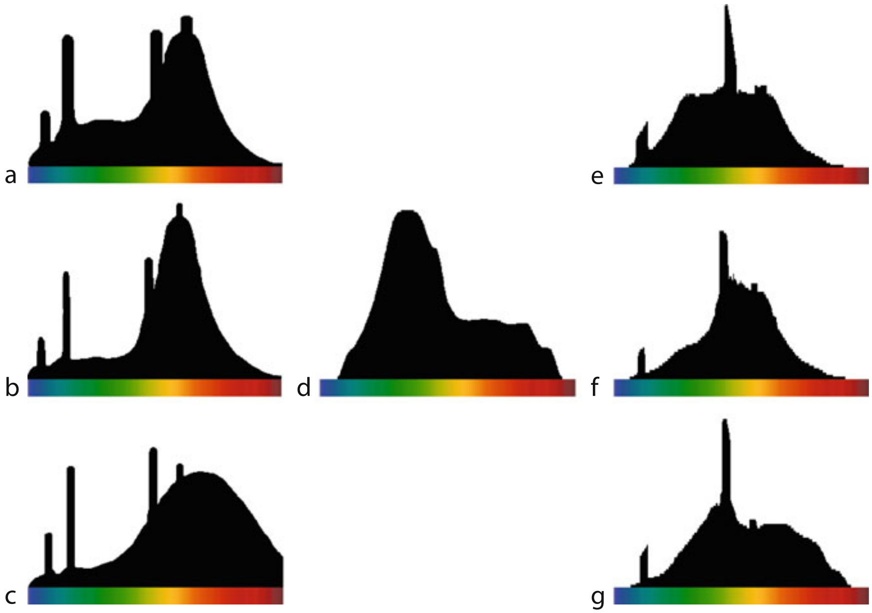


Fig. 7.6

Vividly coloured surfaces sometimes exaggerate the differences between light sources that subjectively appear similar. In the past, this made fluorescent lamps unpopular for domestic use. (a)–(c) Radiometric spectra of three fluorescent tubes. The manufacturer describes these lamps as "Cool white," "Warm white," and "Warm white deluxe," respectively. (d) Reflectance characteristic of a sample surface. This is the spectrum of the light reflected from a "green" surface illuminated with pure white light. (e)–(g) Spectra of the light reflected from the surface when it is illuminated by the fluorescent tubes (Adapted from data in [9])

takes it to a window to assess its "real colour." On the very rare occasions when he accompanies her on shopping expeditions, he observes that this behaviour is not unique! This is evidence that lamp technology requires further development.

7.4.4 Driver Circuits

Fluorescent lamps require an auxiliary device, called a ballast, to regulate the current flow through the tube. When operating from AC mains power, an inductor is normally connected in series with the plasma tube (➲ *Fig. 7.7*). This limits the current surge that would otherwise occur when the plasma starts to conduct current. (The plasma tube presents a negative resistance, which would lead to run-away operating conditions without the ballast.) This inductor also helps to maintain the glow for longer, when the supply voltage falls below the critical value needed to maintain the discharge. The ballast is also necessary to ensure that the voltage applied to the tube is high enough to initiate the discharge. It is common practice to use an auto-transformer to raise the voltage applied to the tube above the so-called *striking voltage*. An auto-transformer can also act as the ballast inductor. A capacitor may also be needed to adjust the current-voltage phase (i.e., ensure that the power factor is close to unity). When operated directly from a DC supply, the polarity of the applied voltage should be reversed regularly, to make sure that liquid mercury does not accumulates at one end of the tube. However, fluorescent lamps are only rarely operated directly from DC. For Machine Vision applications, it is common to employ an inverter running at a frequency considerably higher than the camera's field-scan frequency; the inverter frequency is typically 10–50 kHz. Fluorescent lamps running directly from mains power, flicker at twice the line frequency. (100/120 Hz) During each mains cycle, the supply voltage twice drops to a value that is unable to sustain current flow through the plasma. The resulting flicker is normally imperceptible to the human eye but, for a camera, it can be a serious nuisance; hum bars are visible when the video signal is viewed on a monitor (➲ *Fig. 7.8*). When the video signal is digitised, the hum bars produce

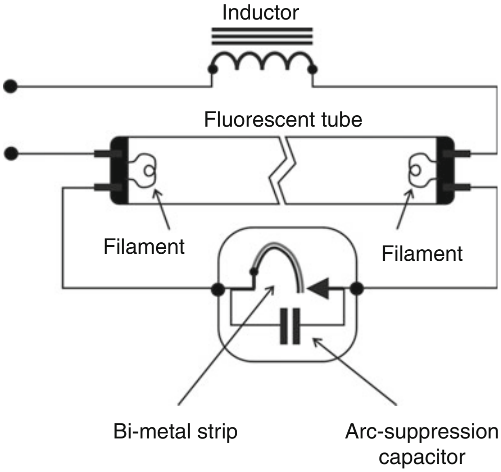


Fig. 7.7
Ballast and starter for a fluorescent tube

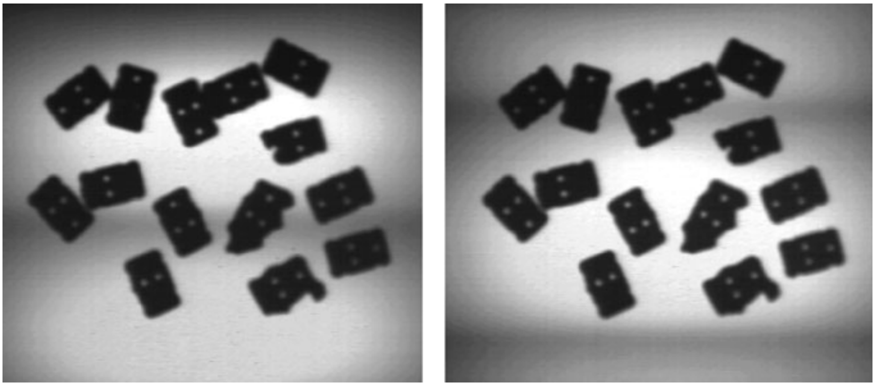


Fig. 7.8
Hum bars caused by fluorescent lighting that is not synchronised to the camera. The dark bars move slowly up/down the monitor screen (A similar effect occurs when LED lights are driven from an alternating current source)

a slow change in the intensity along the vertical axis of the image. Flicker is more pronounced if the tube is not operating symmetrically during the two half cycles of the supply voltage.

Most fluorescent light fixtures cannot be connected to a standard dimmer intended for incandescent lamps. Nevertheless, there are numerous commercial devices available to do this. These usually rely on electronic inverters to apply (square-wave) pulses of variable width to the tube. Most dimmers keep the filament(s) fully heated all through the AC power cycle. As the current through the plasma is reduced to zero, the ions quickly combine with free electrons to form neutral atoms. As the supply voltage rises again, the hot filament ensures that there is a plentiful supply of ionised atoms available to re-form the plasma and “re-ignite” the discharge.

7.4.5 Ageing

The efficiency of new fluorescent tubes may vary from about 16–95 lmW^{-1}. The phosphor degrades as the tube ages, resulting in a significant reduction in the average brightness over its working life. Most readers will be aware of this from their personal experience; when new fluorescent tubes are fitted in a room, there is a noticeable increase in brightness. After about 25,000 h, the lamp brightness will typically be half that of a new tube (Some manufacturers claim a longer working life.).

7.4.6 Compact Fluorescent Lamp (CFL)

A Compact Fluorescent Lamp (CFL) is a folded tube, fitted with a standard bayonet or screw connector, and an integral electronic starter/ballast unit. It is designed to be a direct replacement for a regular incandescent bulb, although it is often larger and heavier (*Fig. 7.3*). Since it uses a high-frequency driver, a CFL does not produce significant flicker. The author has often experienced difficulty in running two CFLs from the same domestic supply line; the electronic drivers appear to generate sufficient interference to cause one unit to stop the other from working properly. Sometimes, neither CFL will ignite.

7.4.7 UV Fluorescent Tubes

The phosphor and glass envelope in a standard fluorescent tube both attenuate UV emissions that are harmful to people. (The chief dangers are skin cancer and cataracts.) By simply omitting the phosphor altogether, it is possible to a lamp that is a strong source of short-wave UV radiation. To maximise the UV transmission through the glass tube, Quartz or Wood's glass is often used. The latter also attenuates visible light and hence increases the UV: VIS ratio. This ensures that UV-induced fluorescence can be seen more easily. Producing fluorescence is only one of the potential uses for UV light, which can also be used directly for imaging. Some of the possibilities are illustrated in Appendix I. However, optical components and cameras for use in the UV waveband are expensive. Despite the health hazards, UV fluorescent tubes are in wide-spread use: germicidal lamps, insect traps, viewing fluorescent ink/paint (e.g., for security marking), microphotolithography, erasing EPROMs, sun-beds and for curing materials, such as inks and adhesives (for dentistry and a range of industrial applications). "Blacklights" are fluorescent lamps that generate UV radiation just outside the visible waveband (UV-A, wavelength about 0.360 μm). They are built in a similar way to conventional fluorescent lamps but the glass tube is coated with a special phosphor that converts dangerous short-wave UV to safer UV-A.

7.4.8 Electrodeless Induction Lamp

An *Electrodeless Induction Lamp (EIL)* is a fluorescent tube in which the discharge is sustained by electro-magnetic induction, rather than an applied potential gradient between two electrodes. In an EIL, there are no electrodes within the glass tube containing the plasma. Instead, two coils, connected in series, form the primary of a transformer. The secondary is formed by a closed loop (i.e., a torus) of mercury-argon plasma. The primary coils and the mercury-argon (Hg-A) plasma torus are coupled magnetically. The device operates as a transformer with a negative resistance in the secondary circuit. Induction sustains the current flow in the plasma. The inside of the bulb has a fluorescent-conductive coating to suppress EM radiation. Since it has no filament, an electrodeless induction lamp has a very long life; one manufacturer (Osram Sylvania Inc., [10]) states that the light output for a 100 W lamp falls to 70% of its initial value after 60,000 h and is still above 65% after 1,00,000 h also see ❯ *Table 7.6*).

Table 7.6
Typical electrodeless induction lamp

Feature	After 100 h	After 60,000 h
Luminous Flux (lm)	8,000 lm	5,600 lm
Luminous Efficacy (lm/W)	80	56
Power consumption	23 W	
Warm up (90% of final output)	30 s	
Operating frequency	250 KHz	

The author once observed a standard 6′ (0.914 m) fluorescent tube fully illuminated by an intense 16 kHz field, which pervaded the whole of a large room in a low-frequency radio transmitter facility. However, the radiation level was so high that it would be unwise to spend long working there. The point is that there does not need to be a direct connection in order to sustain the discharge in a tube of this type. In certain situations, it is useful to be able to operate a light source remotely, without any direct connection to it.

7.4.9 Cold-Cathode Fluorescent Lamps

Unlike a normal fluorescent tube, a cold-cathode fluorescent lamp (CCFL) has no electrically heated filament. CCFLs sometimes have a rare earth coating on the cathode to enhance electron emission, while others contain a source of beta radiation, to initiate ionisation of the gas. Small CCFLs are often supplied with an electronic driver board, which is also conveniently small. This allows the device to be driven from a low-voltage supply. CCFLs are available in a range of colours and UV-A versions are available.

A typical laptop computer display uses one or more CCFLs for the backlight. A CCFL provides a bright white light source that can be diffused by a panel behind the LCD. Since it does not generate much heat, the CCFL does not raise the temperature of the surrounding components much above ambient level. The diameter of a CCFL for backlighting a computer screen may be a small as 2 mm and its length over 300 mm. It is easily broken! One of the most frequent uses for CCFLs is by car enthusiasts and computer hobbyists who adorn their beloved toys with lamps that cast an earie glow.

7.4.10 Remarks

The dangers of handling and safe disposal of fluorescent lamps are well understood, as they have been widely used over many years for environmental lighting. Fluorescent lamps contain mercury: typically 6 mg for a modern 25–40 W lamp. (Older devices contain considerably more than this.) When the tube has been in use for a long time, most of the mercury is in the phosphor, which is scattered as a fine powder when the tube breaks. Mercury presents a general health hazard, but particularly for pregnant women and children. Hence, when fluorescent lamps are to be used in the vicinity of food-stuffs and pharmaceuticals they should be protected within a suitable enclosure.

Despite the higher initial cost of a fluorescent tube, compared to an incandescent lamp, its lower energy consumption over its working life makes it cheaper.

Some people are pathologically sensitive to mains powered fluorescent light. People with epilepsy, autism, lupus and Asperger’s syndrome may feel ill when they are subjected to the flickering of fluorescent tubes. The author cannot tolerate flickering fluorescent lights for more than a few seconds, as he finds it extremely irritating!.

Fluorescent lamps require a ballast to stabilize the lamp and to provide the initial striking voltage required to initiate ionisation and creation of the plasma. Electromagnetic ballasts are often used for mains powered fluorescent light and may produce an annoying audible humming, or buzzing, noise.

Fluorescent lamps can be driven from a low-voltage DC (direct current) supply, by using an electronic inverter. (Remember that conventional lamp ballasts do not operate on direct current.) If this is not properly designed, the overall efficiency can be greatly reduced, due to ohmic energy loss in ballast resistors.

When a tube is driven from a high-voltage DC supply (i.e., high enough voltage to strike the arc), the mercury tends to migrate to one end of the tube. In this case, one end of the lamp produces most of the light. To avoid this, the lamp, or the polarity of the voltage applied to it, should be reversed frequently.

Fluorescent lamps present a non-linear load and generate troublesome harmonics, which can be troublesome on the mains power supply. Care should be taken to remove these by suitable filtering. Radio-frequency electrical noise is also generated by the arc.

Fluorescent lamps operate best at room temperature; at much lower or higher temperatures, efficiency decreases. At low temperatures (below freezing) special lamps may be needed. A "cold start" driver is available.

7.5 Light Emitting Diodes (LEDs)

For many Machine Vision applications, LEDs provide the ideal light source. Individual LEDs are physically small; a light-emitting chip is far smaller than the familiar packaged device. A large number of unpackaged LED chips can be mounted directly onto a rigid board, or flexible circuit, or assembled into a 3-dimensional structure.

Using LEDs, it is possible to build light sources with a very wide range of shapes and spectral characteristics. LEDs are highly efficient and are electrically safe, since they only require low operating voltages. As a result, they can be operated safely in an explosive atmosphere. Individual LEDs are often packaged so that they generate a narrow cone of light that can be directed as needed, rather than spreading unwanted light in all directions. This reduces the likelihood of triggering migraine and epileptic seizures, when they are strobed. LEDs are efficient and generate little unwanted heat. Strobing reduces heat generation to an absolute minimum. They can be overdriven in pulsed mode ("turbo-charged"), to provide a bright, short-duration pulse of light, just at the moment when it is needed. It should be noted, however, that some high-powered LEDs do not lend themselves to overdriving. LEDs are available in a wide range of colours. They are also available for working in the UV-A and NIR wavebands (➲ *Fig. 7.9*). Since they are so versatile, assemblies of LEDs present numerous possibilities that were not available previously. The principal characteristics of LEDs are summarised in ➲ *Table 7.7*.

7.5.1 LED Spectra

LEDs are available to cover the whole of the visible (VIS) spectrum, in increments of 0.01 μm. Moreover, LEDs are available for the near infra-red (NIR) and near ultra-violet (NUV, UV-A) wavebands. "White" LEDs can be built in one of two ways:

1. A single component package may contain three separate diodes, whose emissions combine to give the visual impression of white light. Devices of the kind are known as RGB LEDs. It is sometimes possible to drive each LED separately, thereby allowing the visual sensation of colour to be adjusted.
2. A UV LED is used to illuminate a fluorescent material (Sometimes, this is erroneously called a phosphor.).

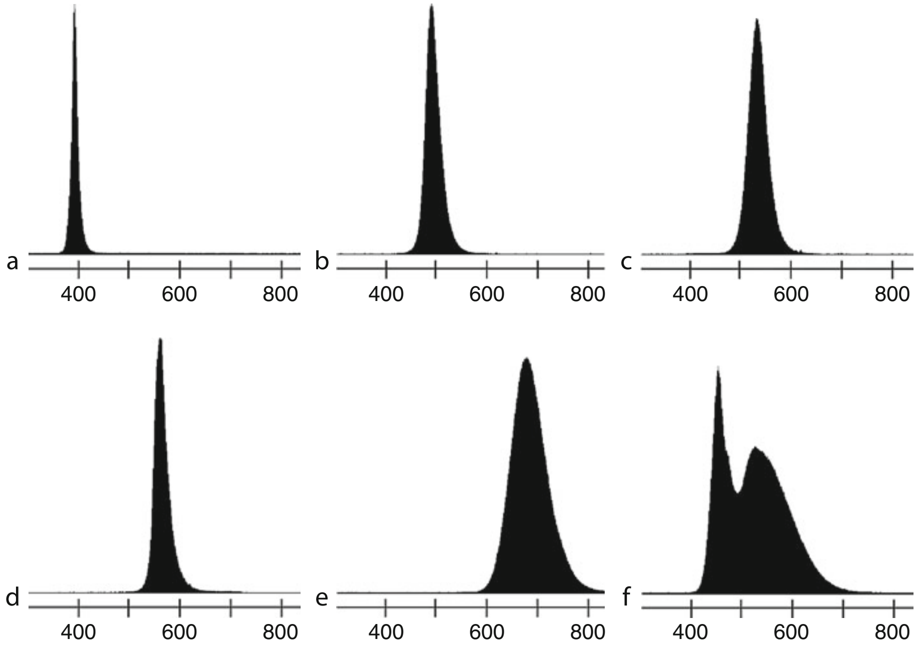


Fig. 7.9

Sample LED spectra. The horizontal scale is measured in nanometres. (a) Near UV (UV-A) (b) Blue. (c) Green. (d) Yellow. (e) Red/IR. (f) "White" LED, obtained by exploiting fluorescence. Although this gives the subjective impression of being nearly white, the spectrum in the VIS waveband is far from flat

The narrow waveband of a single-colour LED can be used to good effect in some applications requiring simple colour discrimination. (A similar effect can be obtained using a white light source and a narrow band-pass filter.) Using a narrow part of the spectrum can be beneficial in a less obvious way: the effects of lens aberrations are minimised, thereby allowing sharp focussing to be achieved. Narrow-band LEDs, used in conjunction with a matching narrow bandpass filter over the camera lens, provides an effective way to reduce the effects of ambient light, which almost always has a broad spectrum.

We have already pointed out that, by mixing the light from a suitably chosen set of LEDs, it is possible to approximate the visual effect produced by white light. More generally, we can adjust the mixing proportions of three "primary" light sources to generate a very wide range of colours, as perceived by a human observer (➲ *Fig 7.13b*). The reader may well take this as an indicator that we can build a programmable LED array whose colour can be changed at will. Such a device would be very useful for Machine Vision but there is a snag! In practice, a large number of quite different spectra may be perceived in the same way by a human being. (This is called *metamerism.*) A camera viewing two metameric surfaces that are both illuminated by the same three narrow-band LEDs, may well generate quite different RGB outputs. In other words, a given set of RGB camera outputs generated using this type of illumination source does not determine unequivocally what a person would see ➲ *Fig. 7.10* explains this. Illuminating a scene with multiple narrow-band LEDs has a somewhat limited value for fine colour discrimination.

Table 7.7
Summary of LED characteristics

Feature	Remarks
Size	The LED chip is physically small but it may be necessary to use a heat sink and provide environmental protection
Voltage	Low voltage; electrically safe to humans. Safe in an explosive atmosphere
Current	Low current; often $<$ 100 mA/LED. Easy to switch and strobe. A constant current source is usually prefered to a resistive current limiter
Power/heating	Low power. Little heat generated. Low thermal emission (for VIS & UV LEDs)
Power conversion	Efficient, comparable to discharge tubes
Operation (steady)	Simple circuitry
Operation (pulsed)	Peak output can be safely increased by strobing (called overdriving or *turbo-charging*)
Spectrum	LEDs naturally emit over a narrow spectral band. The spectrum can be adjusted (i.e., broadened/shifted) using fluorescence. Multi-spectral devices available. (two or three different LED chips are mounted in one package)
Beam angle	Most LED chips are not naturally narrow-beam emitters. However, they are often packaged with a lens, so that the output beam is concentrated into a narrow cone. This reduces light spillage, as well as improving efficiency and safety
Health/safety	Electrically safe
	Continuous operation: optically safe
	Pulsed operation: Risk of migraine & epilepsy low, due to narrow beam
Spectrum	Each LED emits over a narrow spectral band
	Emission spectrum can be broadened using fluorescence
	Wide choice of wavelengths: UV-VIS-IR in steps of 0.01 μm (➔ *Fig. 7.9*)
Packaging	Individual LEDs are very small. Heat sink may be needed for high power LEDs
	LEDs are available in encapsulated or "bare chip" form or in surface mount packages. Many LED chips, of different colours, can be mounted on a rigid board, semi-flexible sheeting, complex assemblies, endoscope, etc.
Working life	Measured in 10,000s of hours; typically 50,000 h
	For a discussion of the aging process see [11]
Multi-LED sources (Some possibilities – also see ➔ *Figs. 7.9* and ➔ *7.10*)	Single line of LEDs, uniform colour or white (most common)
	Three closely spaced lines of red, green and blue LEDs, switched separately
	Single line of LEDs; interleaved red, green and blue LEDs controlled separately
	Ring light; three circles of red, green and blue LEDs, controlled separately
	Ring light; interleaved red, green and blue LEDs, controlled separately
	Dark field; interleaved red, green and blue LEDs, controlled separately
	Back light; interleaved red, green and blue LEDs, controlled separately
	"Flying spot" moves along a linear, circular, spiral or random path
	"Rainbow" array; LEDs of different wavelengths (switched or steady operation)

7.5.2 LED Assemblies

LEDs are now finding application in domestic and commercial lighting fixtures, for example in retail displays. Multi-LED assemblies are now being offered as replacements for incandescent filament lamps, which are far less efficient and have a shorter working life. The data in *Table 7.8* relates to one such assembly: the TVR-7 spotlight, sold by Advanced Lumonics

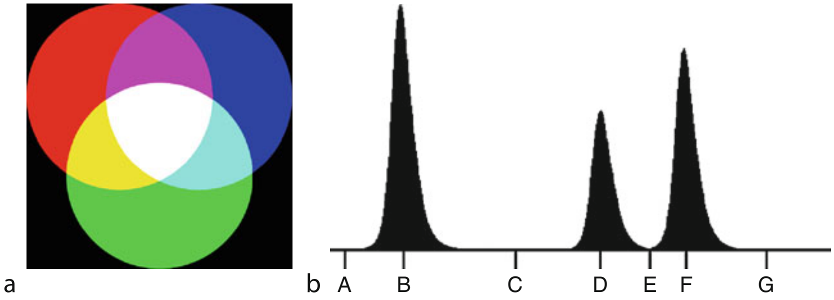


Fig. 7.10
Mixing light of different colours. (a) Classical representation of additive mixing of three primaries. A very wide range of colour sensations can be produced by varying the relative intensities of the three sources. (b) Spectrum of an additive mixture of light from 3 LEDs emitting light in narrow wavebands centred on B, D and F. By mixing light from these source, we can generate a wide variety of colours (including "white") However, no information can be gained about the spectral response of the subject at wavelengths, A, C, E and G

Table 7.8
LED assembly, sold as a replacement for a quartz halogen spotlight (see *Fig. 7.11*)

Feature	Comments
Physical construction	7 LEDs, 1 W each
	Aluminium radial fin heat sink
	Screw base
Input voltage	85–240 V (Same unit can be used in USA/Canada and in Europe)
Power consumption	7 W
Output	385 lm
Operating temperature	$\leq 53^\circ$
Beam Angle	30°
Performance	Equivalent to 80–100 W incandescent lamp
Life	>50,000 h (5.7 years continuous operation) (Life of an incandescent bulb: 1,000 h)
Cost (2007 prices)	Annual running cost: $3.00 (1 years @ 12 h/day)
	Over 11 years: US$100 (US$550 for a comparable incandescent bulb)
Environmental impact	Contains no hazardous substances. (Specifically no mercury)

Inc. [12] (see *Fig. 7.11*). This lamp is marketed as a replacement for a standard quartz halogen bulb and is intended to be used in architectural applications, such as creating mood lighting and illuminating wall hangings. The manufacturer recommends that this lamp be operated so that its beam is projected vertically downwards, in order to maximise convection cooling. Typically, cost and performance figures are based on the assumption that a LED lamp has a working life of 11 years. In this case, the manufacturer expects that it will last much longer,

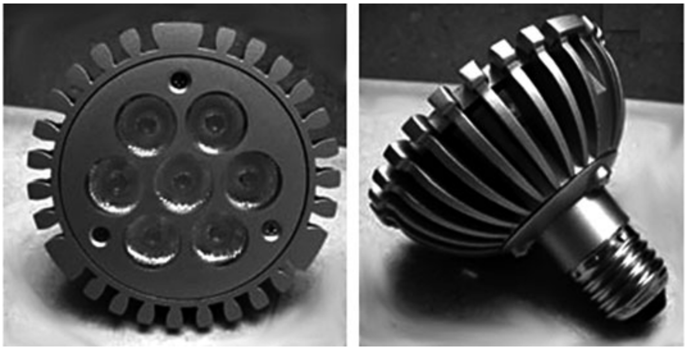


Fig. 7.11
A 7-LED spotlight, sold as a replacement for an 80 W incandescent lamp. Notice the finned heat-sink (From http://www.earthled.com/vr7.html accessed 7th April 2011)

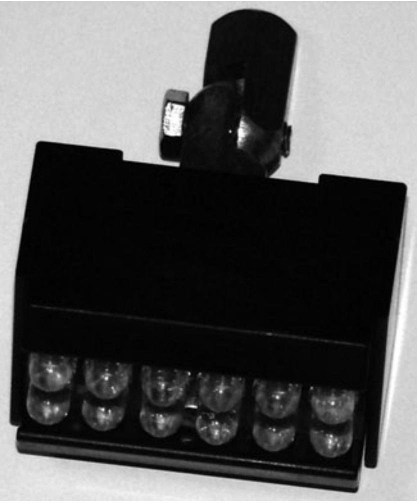


Fig. 7.12
Rugged 12-LED lighting module. These modules can be connected together in daisy-chain fashion and operated from a single DC power source, or strobe driver unit (Module supplied by LED lighting modules [14])

without giving a precise estimate of the life-time. Another series of LED lamps by the same manufacturer, is intended to provide replacements for fluorescent tubes [13]. One device of this type uses an array of 330 LEDs and provides an output of 1,350 lm. These two lamps are powered directly from the mains supply and therefore flash at 100/120 Hz. Although this is perfectly safe for human beings, LED lighting units powered directly from the mains supply are liable to generate hum bars when used in conjunction with some cameras.

Figure 7.12 shows another arrangement that is more suitable for industrial applications: 12 “white” LEDs are mounted in a ruggedised module. Several of these modules can be connected together in a “daisy chain” manner and can be driven by a pulsed, or direct current (DC) source. Clearly, there are numerous possible variations, with the LEDs arranged in lines, circles, rectangles, etc. LEDS can be mounted on rigid, semi-flexible or even flexible strips and sheets [15] (*Fig. 7.13*). LEDs can conveniently be mounted on a 3D structure, for example, along the lines of latitude, or longitude, on a hemisphere. “Naked” LED chips are also small enough to mount around the objective lens of a microscope, or endoscope. Some of the LED lighting unit that have been designed specifically for Machine Vision are shown in *Figs. 7.14* and *7.15*.

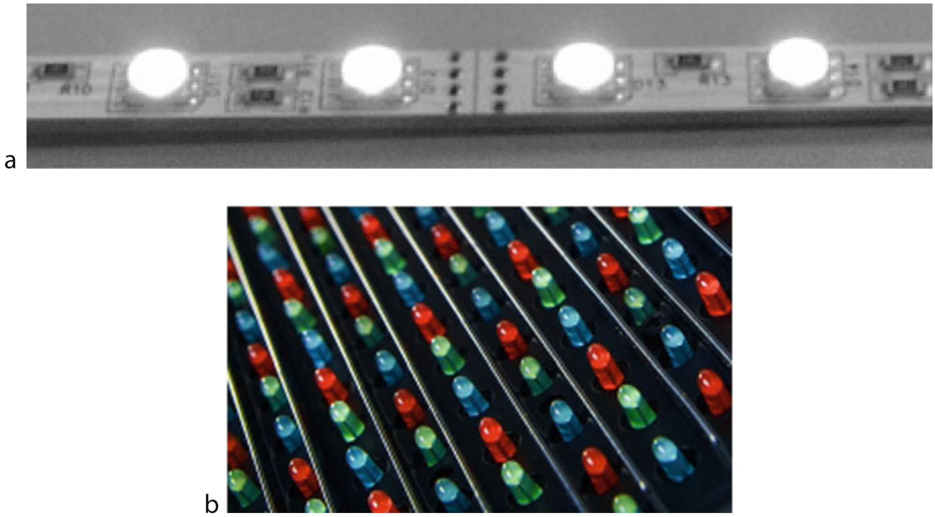


Fig. 7.13
The DIY approach to building LED arrays. (a) ‘‘Bare bones’’ LED light strip containing 30 RGB LEDs on a strip of length 575 mm. Strips can be butted together, or cut every 50 mm. The strip is physically and environmentally unprotected and is intended for use in retail displays. Although this particular device is not accurate enough for high precision Machine Vision applications, it suggests the possibility of creating illumination devices by literally cutting up and then re-assembling devices to suit the requirements of individual applications [16]. (b) An assembly of red, green and blue LEDs. By adjusting the drive currents for red, green and blue LEDs separately, the overall colour of the array appears to change and can even be made to approximate white. However, the spectrum always contains just three narrow peaks, with very little optical energy in the gaps between them

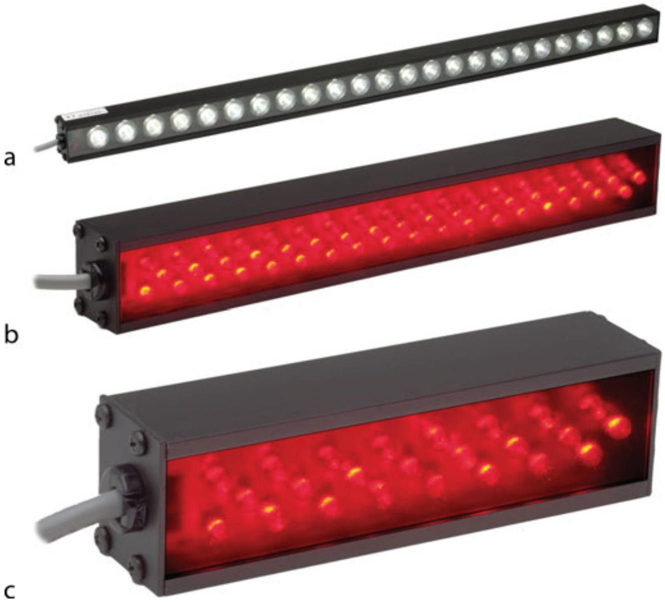


Fig. 7.14

Commercial LED assemblies. (a) Linear array. A diffuser might be fitted to provide more uniform back illumination. Available in lengths up to 2m. (b) and (c) Rectangular arrays, possibly incorporating LEDs of different colours. [Images kindly supplied by Advanced Illumination, Inc., Rochester VT 0576, USA.]

"Do it yourself" construction of a LED array requires great care. There are two principal difficulties:

1. Balancing the brightness of individual LEDs in the array.
2. Aligning the LEDs, so that their beams are all projected in the correct direction.

Specialist help is likely to be needed; several manufacturers will design a LED array for a given application. It is possible to specify the LED brightness, placement, wavelength(s) and operating conditions (DC or pulsed). One manufacturer has developed a lighting kit in which standard LED modules can be plugged into a PCB that has the wiring and drive circuits already installed. The LED module consists of a single LED placed at the focal point of a parabolic mirror. Each module emits a parallel beam (➲ *Fig. 7.16*). This concept could be developed further, to build a fully programmable array of RGB LEDs.

7.5.3 Driver Circuits

An LED has only two leads and is therefore a very simple device to operate: the drive circuit can be as simple as a DC power source and current-limiting resisitor (➲ *Fig. 7.17a*). Several LEDs can be operated together by connecting them is series (➲ *Fig. 7.17b*). If several LEDs are to be operated in parallel, separate current limiting resistors are needed for each one (➲ *Fig. 7.17c*). It is not good practice to connect LEDs directly in parallel. This will result in an uneven distribution of current through them, highly variable brightness and probably cascading failure of all LEDs.

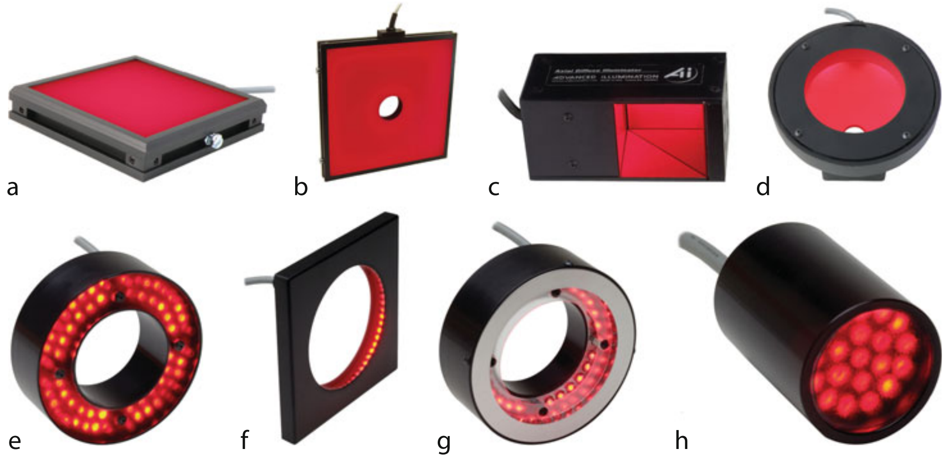


Fig. 7.15
More commercial LED modules for Machine Vision. These devices are available in an RGB "all colour" version. (a) Back-light. 100 × 100 mm. (b) Diffuse illuminator with central viewing aperture. Working area, 150 × 200 mm. (c) Coaxial illumination and viewing module. (d) Omni-directional light source. Inner diameter, 161 mm. (e) Ring light, projects light forwards. Inner diameter, 55 mm. (f) Ring light, projects light inwards. (g) Ring light, projects light beams forwards and/or inwards. Inner diameter, 55 mm. (h) Spot light. [Images kindly supplied and manufactured by Advanced Illumination, Inc., Rochester VT 0576, USA.]

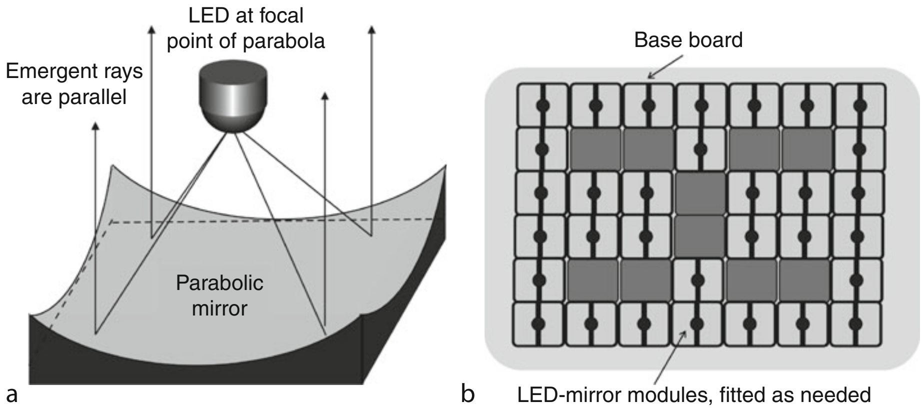

LED at focal point of parabola
Emergent rays are parallel
Parabolic mirror
a
Base board
b
LED-mirror modules, fitted as needed

Fig. 7.16
Bespoke LED illumination system. (a) Module consisting of a single LED and a parabolic mirror. Power leads for the diode are not shown. (b) Modules are fitted to the base board as needed. This contains wiring and drivers for a fully populated board. The LEDs may be of different colours

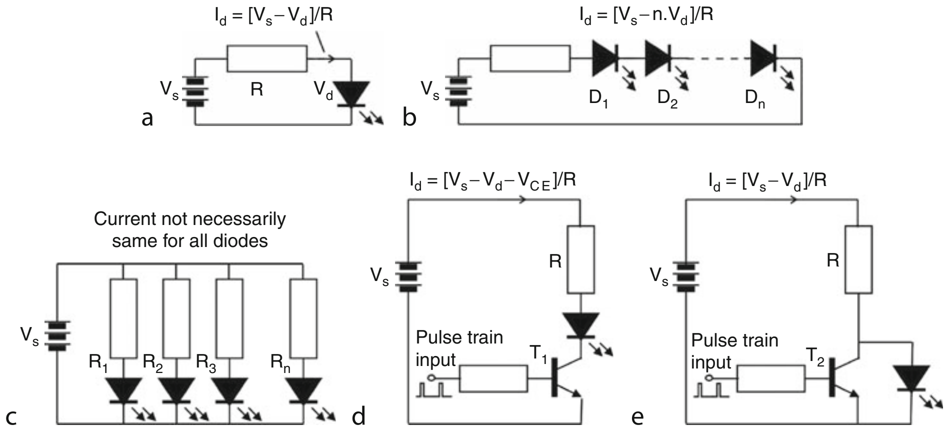


Fig. 7.17

LED drivers. (a) Single LED. Its brightness is proportional to I_d. (b) LEDs in series; all receive the same current. (c) LEDs in parallel require separate resistors. (d) Strobing an LED using a series transistor switch is series with the LED. When T_1 is ON, the LED is also ON. (e) Strobing a LED using a shunt transistor switch. When T_2 is OFF, the LED is ON. This arrangement is wasteful of power but the transistor has no effect on the LED brightness

There are two ways to control the brightness of a LED:

1. Adjusting the drive current
2. Pulse width modulation (PWM)

A simple way to control the drive current is to vary either the supply voltage or the series resistor. An adaptive drive circuit can be built around an operational amplifier that receives a feedback signal from a photo-sensor measuring the light output level. Analogue control like this does readily not allow turbo-charging, or regular strobing.

In fact, digital control is easier to achieve. Bear in mind that a video camera effectively counts the number of photons falling on its photo-sensors over a finite period. (The light-integration period is a significant proportion of the video frame period.) Within this integration period, it does not matter how the light level varies with time; the only significant parameter is the integral over time of the light flux falling on each photo-sensor. There is no need for analogue control of the drive current. Synchronisation of a pulsed LED driver with the camera's light integration period may be necessary, to avoid creating hum bars. (➜ *Figure 7.8* shows hum bars generated by fluorescent lighting. The effect with LEDs is broadly similar.) Using simple feed-back control, it is possible to ensure that optimal image quality is maintained, even if the LED brightness varies significantly over time. Notice that strobing saves electrical power, making battery-powered lighting a possibility in certain situations. Even continuously operated LED lighting compares reasonably well with fluorescent and discharge lamps. In many applications, supplying short pulses of power to an LED lighting unit only when needed can reduce the power requirements by 1–2 orders of magnitude. Since LED lighting is highly directional, very little unwanted light is generated, thereby increasing the overall efficiency even further. Reducing stray light also invariably improves image quality too. When we consider all aspects of LED lighting, we see that it is almost ideal for Machine Vision although we must repeat the warning that strobed lighting presents significant health hazards. At least, the low light spillage from LEDs minimises the risk but careful screening and safety switches are still needed.

7.6 Supplement

7.6.1 Representative Spectra

➲ *Figure 7.18* shows a series of representative spectra from a variety of discharge lamps and compares them to daylight and an incandescent filament lamp. Notice the strong spectral spikes that typically occur with standard fluorescent tubes and other discharge lamps, which should only be used with great caution in applications involving colour-sensitive inspection. The spectral spikes occur at fixed wavelengths, determined by the materials forming the plasma. It is therefore impossible for a vision engineer to choose the wavelengths of the peaks a multi-spike spectrum. (This can be done with multi-element LED array.) Notice that incandescent lamps normally produce a smooth spectrum.

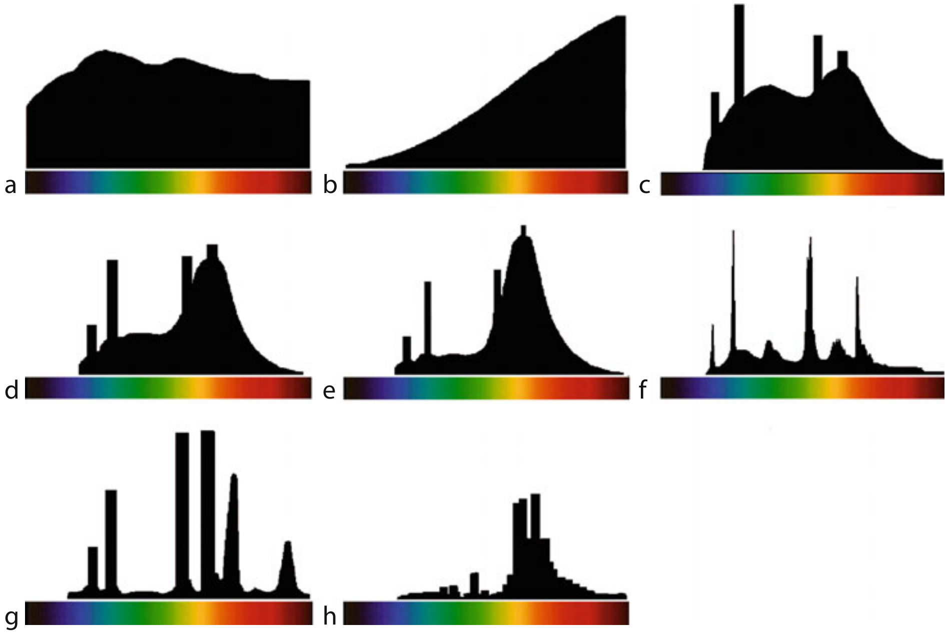


Fig. 7.18

Radiometric spectra (a) Daylight. (b) Standard incandescent filament lamp; a close approximation to a black-body radiator. (c) "Daylight" fluorescent. (d) "Cool white" fluorescent. (e) "Warm white" fluorescent. (f) Triphosphor fluorescent. (g) White Mercury HID. The two highest peaks are shown truncated. (g) High intensity discharge (HID). (h) Another HID (From http://www.gelighting.com Additional photometric spectra are given in http://ioannis.virtualcomposer2000.com, accessed 7th April 2011)

7.6.2 Describing the Colour of a Lamp

7.6.2.1 Black Body and Colour Temperature

The term *Black Body* refers to an object that absorbs all electromagnetic radiation incident upon it; no radiation is transmitted through it and none is reflected. All of the electromagnetic energy emitted by a black body is generated by the thermal agitation of atoms within it. This is an idealised, abstract concept, since no material acts a perfect black body radiator. An open window is a close approximation, as is a "white hot" incandescent tungsten wire.

The concept of *Colour temperature* was devised as way of characterising the colour of a black body light source. The precise form of the spectrum for an ideal black body has been calculated theoretically and can be characterised in closed mathematical form (➔ *Fig. 7.1*) (also see ➔ *Fig. 7.4*). Conversely, it is possible to estimate the temperature of a black body by examining its spectrum. The temperature of a black body radiator defines its spectrum exactly. Hence, we are able to represent the colour of a black body radiator by a single number, called the *Colour Temperature*. The colour temperature of a real light source is determined experimentally, by comparing its chromaticity visually with that of a black-body radiator. The temperature (in Kelvin, K) at which the ideal black-body radiator most closely matches the colour of the light source is taken to be its colour temperature. If a source is a perfect black body radiator, its physical temperature and colour temperature are the same. An incandescent filament lamp is a good approximation to a black body; its colour temperature is therefore close to that of its filament.

For a non-black body source, the *correlated colour temperature (CCT)* is a more appropriate way to characterise its colour. This is the physical temperature of a black body radiator, that is judged by the human eye to match the colour of the light from the lamp most closely. The CCT of a fluorescent lamp does not correspond to the physical temperature of any of its physical components, since the light is emitted from a (cold) phosphor bombarded by ultra-violet photons. Similarly, the CCT of an RGB LED source has no physical significance. The concept of colour temperature has no relevance whatsoever for nearly monochromatic sources, such as a green LED, a sodium vapour lamp, or a HeNe laser.

It is important to bear in mind that any wavelength-dependent absorption along the optical path will change the apparent colour temperature of the light. For example, when the Sun is low over the horizon, its effective colour temperature is about 2,500 K. At mid-day, when there is less atmospheric scattering/absorption, it is about 5,770 K. The illumination pathway in a vision system can modify the colour of the light beam and therefore alter its effective colour temperature. This includes the light gathering optics on the luminaire.

Colour temperature is important for historical reasons: it has long been used by photographers and environmental lighting engineers. However, it is a crude way of representing the colour of many types of lamp; more refined consideration of the spectrum is often required.

7.6.2.2 Colour Rendering Index (CRI)

The *colour Rendering Index* (or *Colour Rendition Index)*, measures the ability of a light source to reproduce the colours of various objects being illuminated by that source. Measuring the CRI for a given lamp requires subjective testing. The reader is referred elsewhere for a detailed description of the way in which the CRI is measured [17]. The CRI can only be used to compare two light

Table 7.9

Typical colour rendering indices for various light sources

Source	CRI
Fluorescent – "deluxe cool white"	86
Fluorescent – "daylight"	79
Fluorescent – "cool white"	65
Fluorescent – "deluxe warm white"	73
Fluorescent – "warm white"	55
Incandescent filament lamp	~ 100
Mercury – clear mercury	17
Mercury – white deluxe mercury	45
Metal halide (4,200 K)	85–95
Metal halide (5,400 K)	93
Sodium – high pressure sodium	25
Sodium – low pressure sodium	0–18
RGB LED array	Low values

sources that have very nearly the same colour temperature. The best possible rendition of colours is indicated by a CRI of 100. If a lamp has a CRI of zero, the lamp does not allow any colour discrimination whatsoever. Hence, a low-pressure sodium discharge lamp, which generates nearly monochromatic yellow light, has a CRI of nearly zero. On the other hand, an incandescent filament lamp has a CRI close to 100. A standard "cool white" fluorescent lamp has a CRI of 60–65. A "triphosphor" fluorescent lamp has a CRI of 80–90. LED sources that are nearly monochromatic have very low CRIs. Notice that a light source with a CRI = 80 and CT = 5,000 K is not necessarily better than one with CRI = 70 and CT = 4,000 K (➜ *Table 7.9*).

7.6.3 Analysing Beam Profiles

Almost all illumination systems are required to project light uniformly over the object/scene to be inspected. Since it is only necessary on rare occasions to create a non-uniform lighting pattern, we shall concentrate here on the simple task of characterising a uniform illumination field, using an image processor. In what follows here, we shall assume that the reader is familiar with the command repertoire of the QT image processor (➜ Chaps. 21 and ➜ 41).

7.6.3.1 Experimental Set-up

Place a plain matt white planar surface *in lieu* of the object to be inspected. The camera should view that surface normally.

Pre-processing (two suggestions, may be used singly or in tandem)

- *mdf(5)* *% Median filter – adjust parameter to taste*
- *caf(5)* *% Low pass (blurring) filter – adjust parameter to taste*

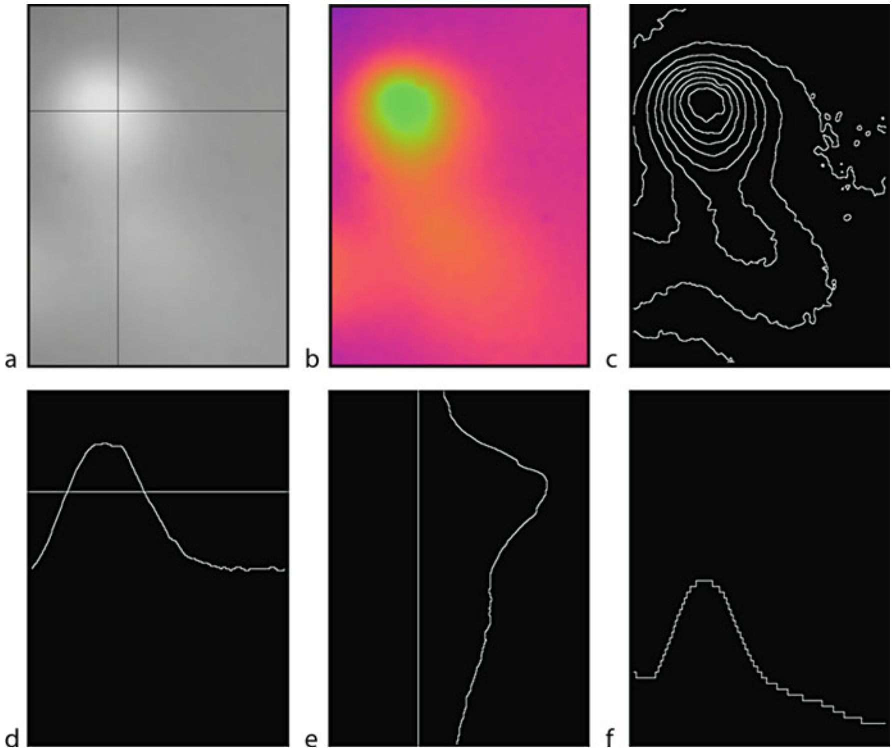


Fig. 7.19

Analysing lighting patterns using simple image processing operations. (a) Lighting pattern, contrived to have local bright spots. The black cross-lines indicate the sections for the profiles plotted in parts (d) and (f). (b) Pseudo-colour plot, provides a convenient way to identify bright/dark spots. (c) Intensity contours (isophotes). (d) Intensity profile, taken along a horizontal section. (e) Intensity profile, taken along a vertical section. (f) Plot of standard deviation along each column

Visual inspection (three options)

- *psc* % Pseudo colour (➲ *Fig. 7.19b*)
- *heq, psc* % Histogram equalisation & pseudo colour
- *enc, psc* % Enhance contrast & pseudo colour

Global Measurements (These suggestions may be used singly or together)

- *a = avg* *% Average intensity (a)*
- *b = dev* *% Standard deviation of intensity (b)*
- *[c,d] = gli;* *% Minimum & maximum intensities (c & d respectively)*
- *e = ihm; thr(x,x)* *% Most frequently occupied intensity level (e)*
- *wri, heq, thr, rei, mni, [x,f] = gli* *% Median intensity (variable f)*

Column Values (Curves plotted)

- *plt* % Intensity profile (➲ *Fig. 7.19d, e*)
- *csd,plt* *% Standard deviation along each column* (➲ *Fig. 7.19f*)

- *cox,[a,b] = dgw,plt(b)* *% Column maximum*
- *neg,cox,neg,[a,b] = dgw,plt(b)* *% Column minimum*

Isophotes (Intensity contours. See ➜ *Fig. 7.19c*)

mdf(5)	*% Median filter* – adjust parameter to taste
caf(5)	*% Low pass filtering* – adjust parameter to taste
sca(4)	*% Posterise – reduced intensity resolution (to four bits)*
sed	*% Edge detector*
thr(32)	*% Threshold* – adjust parameter to taste

7.6.4 Web Information Sources

Advanced Illumination	http://www.advancedillumination.com/
CVI Melles Griot	http://www.cvimellesgriot.com/
Edmund Optics	http://www.edmundoptics.com/
Fiberoptics Technology, Inc.	http://fiberoptix.com/
General Electric	http://www.gelighting.com/
Lumitex	http://www.lumitex.com/
Nerlite	http://www.microscan.com/en-us/
Newport	http://www.newport.com/
Perkin Elmer	http://www.perkinelmer.com/
Philips	http://www.lighting.philips.com/
Stemmer Imaging	http://www.stemmer-imaging.de/
Stocker Yale	http://www.stockeryale.com/
Sylvania	http://www.sylvania.com/
Vision Light Tech	http://www.visionlighttech.com/

References

1. Luminosity function, Wikipedia. http://en.wikipedia.org/wiki/Luminosity_function. Accessed 23 Feb 2011
2. Incandescent light bulb, Wikipedia. http://en.wikipedia.org/wiki/Incandescent_light_bul. Accessed 24 June 2010
3. Most common filament discription, standard products. http://www.standardpro.com/product/categories/incanguide.aspx. Accessed 24 June 2010
4. Henderson ST, Marsden AM (1972) Lamps and lighting, 2nd edn. Edward Arnold (Publishers) Ltd, London. ISBN 0 7131 3267 1
5. Lee M, Christian W (2011) My blackbody applet. http://webphysics.davidson.edu/alumni/MiLee/java/bb_mjl.htm. Accessed 23 Feb 2011
6. Light measurement handbook, International Light Technologies, Inc, Peabody, MAS, USA. http://www.intl-lighttech.com/services/light-measurement-handbook. Accessed 23 Feb 2011
7. Pattison G, Keith M, Gregory A (2011) Black body radiation. http://www.egglescliffe.org.uk/physics/astronomy/blackbody/bbody.html. Accessed 23 Feb 2011
8. Steer A (2008) Colour temperature. http://www.techmind.org/colour/coltemp.html. Accessed 23 Feb 2011
9. General Electric. http://www.gelighting.com. Accessed 23 Feb 2011
10. Osram Endura, Osram. http://www.osram.com. Accessed 23 Feb 2011

11. LED life expectancy, Electronics Weekly. http://www.electronicsweekly.com. Accessed 23 Feb 2011
12. TVR-7 spotlight, advanced lumonics. http://www.earthled.com. Accessed 23 Feb 2011
13. DirectLED, EarthLED (2010). http://www.earthled.com/flseries.html. Accessed 23 Feb 2011
14. LED lighting modules, imaging source. http://www.theimagingsource.com. Accessed 21 Aug 2007
15. Flexible light bars & Strips, super bright LEDs. http:// www.superbrightleds.com/fit.htm. Accessed 23 Feb 2011
16. LED ribbon tapes, Exled. http://www.led-lightbulbs.co.uk/. Accessed 23 Feb 2011
17. Calculating the CRI, J K Consulting. http://www.kruschwitz.com/cri.htm. Accessed 23 Feb 2011

8 Lighting-Viewing Methods

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_8,

Abstract: The image acquisition sub-system consists of one, or more, light sources, lenses and possibly mirrors, filters, prisms, polarisers, etc. It also contains one, or more, opto-electronic transducers (i.e., cameras and/or scanners). Simply knowing what "building bricks" are available is not enough; we must be able to construct an integrated system. Object presentation and transport are important and require that we address mechanical handling issues: grasping the object, as well as moving it before, during and after image capture. The type of viewing background is important, as are numerous seemingly mundane tasks, such as eliminating ambient light, building vibration-proof optical mountings and keeping optical components clean, dry and cool. In most applications, the placement of the camera relative to the object under inspection and the lights is critical; the image quality derived from the same object may vary from very poor to excellent, depending on the lighting-viewing conditions. This chapter introduces the catalogue of Lighting Viewing Methods (LVMs) in ➋ Chap. 40. The latter presents qualitative descriptions of a variety of "front end" configurations for a vision system. There can be no rigid formula-based approach to designing a lighting-viewing system. This does not, in any way, diminish the need for sound mathematical analysis, where it can be applied appropriately (e.g., lens design). There can be no equation to tell us when it is most appropriate to use, say, diffuse, or dark-field illumination, monochromatic or polarised light, or a combination of these. The system designer must use his/her judgment and experience. Most importantly, he/she must look at the widget and its surroundings! Two diagrammatic representations of the spatial relationship between the lights and camera are presented here. These are useful for recording and summarising the results of experimentation, which the author asserts is the only practical approach to designing the image acquisition sub-system.

This short chapter provides an introduction to the Catalogue of Lighting-Viewing Methods in ➋ Chap. 40.

8.1 Lighting-Viewing Sub-system (LVS)

The LVS is taken to include the following:

- The primary light source(s) and its associated power supply/driver circuit.
- Illumination optics.
- Imaging optics. (This includes devices such as filters, mirrors, lenses, polarisers, flexible or rigid coherent fibre-optic bundles, etc.)
- Integral camera optics (i.e., lens fitted directly to the body of the camera).
- Protective optical windows. (Both cameras and lights may need to be protected in harsh/dirty environments.)
- Camera. (This may employ any one of a range of scan patterns: line, area, radial ("wheel-spoke"), spiral, random, variable zoom. It may sense visible (VIS), ultra-violet (UV), or infra-red (IR) radiation.) It may use a solid-state photo-detector array, or less likely nowadays, a cathode ray tube.

The LVS is of crucial importance for the success of any industrial vision system; good lighting can often transform what would otherwise be an extremely difficult application into one that requires only the most straightforward image processing operations.

There is no well-defined optimum lighting pattern for even moderately complex objects (i.e., those having multiple facets, consisting of several different materials and those having several different types of surface finish). As a result, there can be no prescription for designing a good LVS; the design process relies heavily on human observation and intelligence. Experience counts, too! Knowledge about an object's material composition, surface finish and geometry can often help guide the design process but, in practice, cannot eliminate experimental investigation altogether. There are two reasons. Firstly, for most applications, it would not be cost-effective to design an LVS by modelling "good" widgets. (Here and in ➋ Chap. 40, the term *widget* is used to denote the object being examined.) Secondly, product defects are so highly variable and unpredictable that effective modelling is out of the question.

8.2 Parameters

Machine Vision owes its versatility in large part to the variety of different *lighting-viewing methods* (*LVMs*) that are available. (About 130 LVMs are listed in ➋ Chap. 40 and they can be used in combination.) One of the distinctive features of industrial Machine Vision is that we are very usually free to employ whatever LVM best suits the application. Within the synthetic, controlled environment of a well-run factory, there are often several options for the lighting-viewing conditions, thereby allowing us to optimise image quality. We can, for example, adjust all of the following:

- The number and spatial distribution of the light sources placed around the widget.
- The spatial pattern of light projected onto the widget.
- The intensity of each light source.
- The spectral emission of each light source. (While we can control the colour of the incident light, we are not limited to using only the VIS wave-band.)
- The polarisation of the light impinging on the widget.
- The spectral response of the camera, by placing suitable optical filters in its optical path.
- The polarisation of the light entering the camera.
- The temporal pattern of light projected onto the widget.
- The motion of the widget relative to the lighting-optical system and the lighting-optical system relative to the camera.
- The spatial coherence of the light projected onto the widget.

8.3 Light Sources

There are many different types of primary light sources available (➋ Chap. 7).

- Incandescent filament
- Fluorescent tubes
- Compact fluorescent tubes ("energy saving" types)
- Cold fluorescent tubes
- Gas discharge tube (mercury, sodium, "neon")
- Xenon flash tubes
- Laser, particularly diode lasers

- Electric arc
- Light Emitting Diodes (LEDs)
- Day-light
- Natural light
- Flames and objects heated by flames
- Self-luminous (The widget glows because it is hot.)

8.4 Optical Devices

We can also employ a wide range of optical devices to convey light to the widget and then to form, adjust and transmit an image to the camera. (➋ Chaps. 4 and ➋ 5) The list includes:

- Mirrors. These may be plane, convex, concave, or anamorphic.
- Lenses. These may be convex-convex, convex-concave, plano-convex, plano-concave, Fresnel, or cylindrical. They may be used singly, or in tandem, Micro-lens arrays are also available.
- Polarisers. These may be linear, circular or elliptical.
- Prisms. These come in a variety of shapes providing a range of functions.
- Filters. These may be short-pass, long-pass, band-pass, band-stop.
- Beam-splitters and devices made of birefringent materials.
- Gratings and diffraction filters.
- Quarter-wave plates.
- Diffusers. These may rely on transmission or reflection.
- Fibre-optics. These may be flexible, or rigid, often in the form of a rod or plate.

8.5 Types of Cameras

Several different types of image sensor can be used:

- Array-scan
- Line-scan
- Circular scan (At the time of writing, these are no longer available commercially. Such a sensor can be simulated by sampling a high-resolution image array.)
- Random (program controlled) scan
- Laser scanner
- Range sensor (Several types of range sensor are available. See ➋ Chaps. 9 and ➋ 15)

An image sensor may detect radiation in any of the following spectral bands:

- Visible wave-band (VIS, monochrome or colour)
- Infra-red (IR) radiation
- Thermal radiation (part of the IR wave-band)
- Ultra-violet (UV) radiation
- Multi-spectral, possibly covering VIS, UV and IR
- X-rays
- Gamma rays

Specialised image sensors are also used in Machine Vision and may be based on detecting:

- Eddy currents
- Electron beams
- Magnetic fields
- Microwaves
- Neutron beams
- Nuclear magnetic resonance
- Pressure
- Surface height using a tactile sensor
- Ultra-sound

8.6 Mechanical Handling

So far, we have merely indicated what individual items of equipment are available to the vision engineer; there has been no attempt to list how an LVS can be designed. There are other choices to be made:

- Where are the lights to be placed relative to the widget, optical components and camera?
- How are the widgets to be presented for inspection? They may be presented to the camera individually or in groups, depending on the mechanical handling.
- If the widgets are grouped together, are they touching or overlapping?
- If the widgets arrive singly, are the position and/or orientation known precisely?
- Is the widget static or moving at the moment when the image is acquired?
- If the widgets are moving, is the speed and direction of travel known precisely?
- Is the widget motion linear or rotary?

In many cases, we can use the "natural" motion of artefacts as they are being transported around the factory, or arrange for the widgets to be moved in some controlled way past the camera.

8.7 Experimenting with Lighting

The wide range of options available for each part of the LVS provides great versatility but imposes a considerable burden on the vision engineer when choosing a suitable lighting-viewing method. The task of finding a good LVM is often perplexing, particularly for newcomers to Machine Vision. They frequently make the mistake of restricting their attention to familiar domestic/office illumination equipment (e.g., a desk lamp) and standard illumination-viewing configurations. They do so because they under-estimate the wide temporal and spatial variations in brightness and colour that occur in natural and artificial environmental lighting. These parameters do not normally affect human vision significantly but can seriously disturb automated image processing. It is unrealistic to believe that software can always compensate for poor lighting. Adopting a casual approach to lighting will almost always make the overall system much more complex and far less reliable than it should be. It is only by understanding the general physical principles involved in image formation and appreciating the relative merits of a wide range of LVMs that an optimal system can be selected/designed.

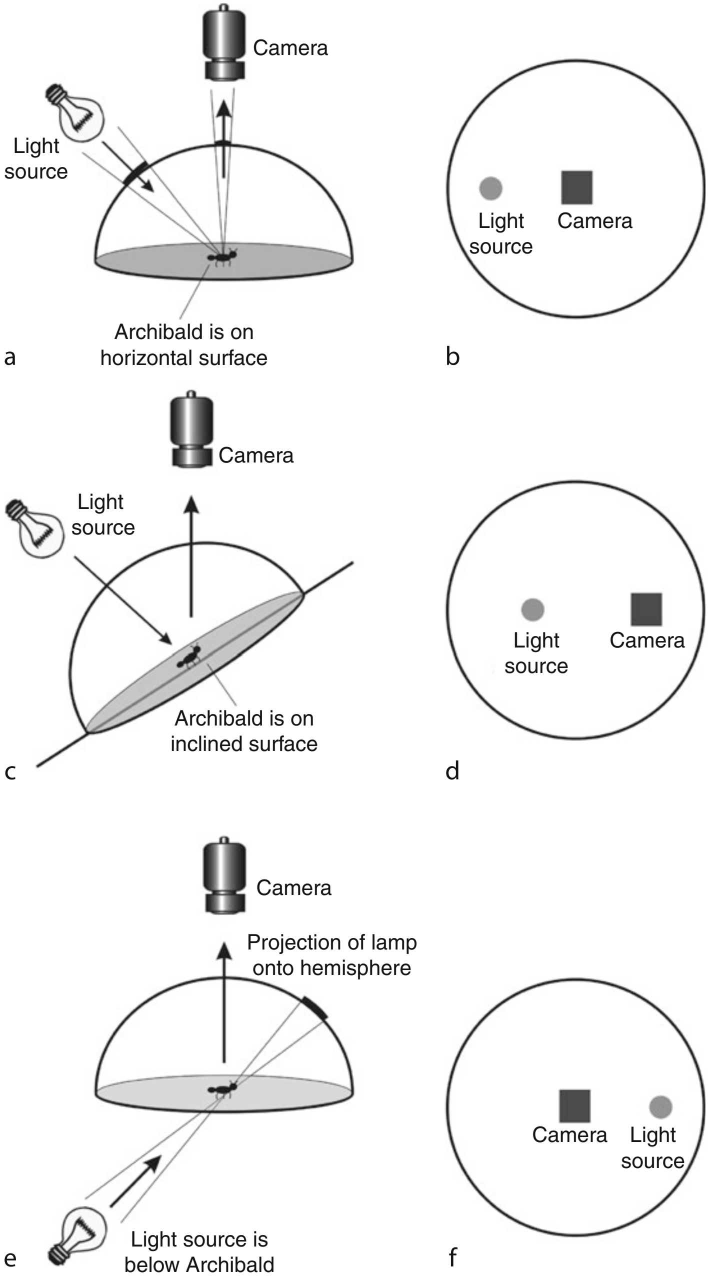

Camera
Light source
Archibald is on horizontal surface
a
Light source
Camera
b
Camera
Light source
Archibald is on inclined surface
c
Light source
Camera
d
Camera
Projection of lamp onto hemisphere
Light source is below Archibald
e
Camera
Light source
f

Fig. 8.1 (Continued)

8.7.1 Taking an Initial Look at the Widget

The most powerful weapon in the vision engineer's armoury is his/her own sight. *First of all, look at the widget*! By moving a "point" source of light around the widget and observing its changing appearance, it is often possible to gain a great deal of valuable insight about which LVMs might be appropriate. To be most effective, this should be done in an otherwise darkened room. A white LED torch provides a narrow directional beam and is ideal for this stage of the experimentation. (Make sure that the LED generates a broad spectrum of light and does not rely on mixing three beams with narrow-spectral peaks from separate red green and blue LED chips.) This preliminary visual examination will often identify the characteristics (e.g., reflectance, colour, height, shape and surface orientation) that determine or limit the contrast of important features. In many cases, it is also worthwhile examining the widgets using a simple camera-monitor video link. This will probably give a more accurate impression of what a vision system "sees" than a person can with the unaided eye. In a large majority of cases, explorative visual examination like this can help the engineer a great deal.

8.7.2 Prototyping the LVS

The next stage of the design process is to construct a *crude* prototype LVS, by manually adjusting the relative positions of the widget, lamp(s) and camera, until a high-contrast image has been obtained. This may simply involve nothing more complicated than holding a lamp in place by hand and judging image quality by eye. However, it is usually better to use an interactive image processing system, as this can provide objective measurements of image quality. (Algorithms for analysing the images are often invented/discovered here as well!) To be most effective, the experimental work should be conducted in a darkened laboratory equipped with a wide range of light sources and optical components: lenses, mirrors, prisms, gratings, filters, beam-splitters, polarisers, diffusers, component mountings, clamps, etc. Together with an interactive image processor (e.g., QT, ➲ Chap. 21), these constitute a very useful prototyping tool kit.

Using the same optical and lighting equipment, but held firmly in place using appropriate clamps and mountings, a second-stage prototype is built, then modified in a more rigorous

◘ Fig. 8.1
Archibald and Illumination Diagrams. Archibald is a tiny "ant" which crawls over the surface of the widget but is unaware of its orientation. The Archibald Diagram is a map, drawn on the plane tangential to the widget surface and indicates where Archibald sees the lights and camera. (a) Front illumination. Archibald is on a horizontal surface. The light source is above the horizon. (b) Archibald and illumination diagrams. These are the same for the lighting-viewing configuration shown in (a). (c) Front illumination. Archibald is on an inclined surface. However, the Illumination Diagram for this situation is identical to that shown in (b). (d) Archibald diagram for the configuration in (c). Notice that the projections of the light(s) and camera have both moved. (e) Back illumination. The light source is below the horizon. (f) Archibald and Illumination Diagrams are identical, since Archibald is on an horizontal surface. The Archibald Diagram for back illumination is drawn by projecting the light source onto the hemisphere above Archibald

experimental investigation. The aim is to produce a conceptual design for the target (i.e., factory-floor) system.

Over a period of more than 30 years, the author collected together a range of illumination and optical devices specifically for experimentation in this manner and has seen several other university and company laboratories do the same. In the author's own personal experience, the experimental approach has been successful in *several hundred* very varied applications studies. These did not all result in factory installations but, in the vast majority of them, it was possible to demonstrate technical feasibility, or lack of it, in a very short time. Over the same period, the author has been closely involved with the development of a series interactive image processors, the latest of which is QT (➜ Chaps. 21 and ➜ 41). Both parts of the prototyping tool kit (i.e., optics-lighting and image processing) are essential for this approach to be really effective.

As a general principle, the initial design work for an LVS should never be undertaken sitting at a desk; a few hours in the laboratory is far more profitably spent!

8.8 Characterising the LVS

Once the prototype LVS has been built and proven, it is necessary to design a well-engineered version or installation on the factory floor. In order to do this, it is necessary to measure the prototype LVS carefully. This can, of course, be done painstakingly and tediously using a ruler but there are other faster ways to achieve the same result using a camera and the interactive image processor again. One of these methods uses an hemispherical mirror and the other a fish-eye lens. These are used in conjunction with a camera used solely for this purpose. (This calibration camera is not part of the prototype LVS.) The angular positions of the lights can be

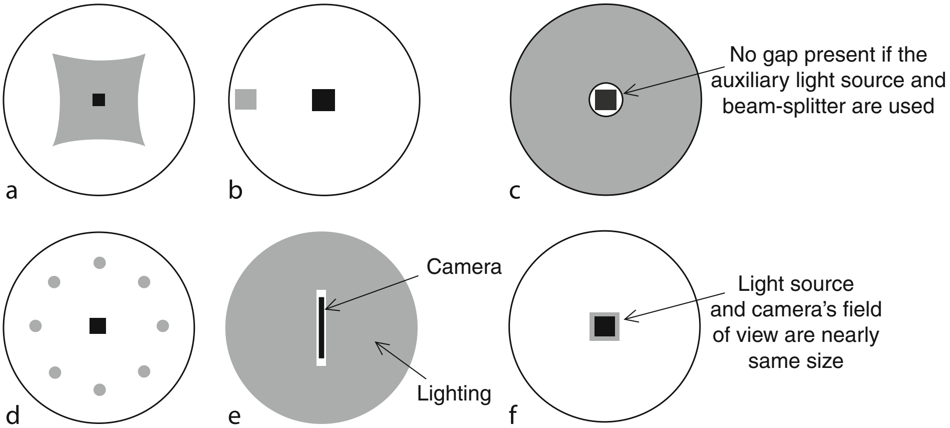

a
b
c
No gap present if the auxiliary light source and beam-splitter are used
d
e
Camera
Lighting
f
Light source and camera's field of view are nearly same size

Fig. 8.2

Illumination diagrams for various lighting configurations. (a) Large square light source. (LVM 31) (b) Grazing illumination from a single "point" source. (LVM 46) (c) "Cloudy day" light source, projects light in all directions. (LVM 85) (d) Eight "point sources" providing multi-directional grazing illumination. (e) Dark-field illumination, with line-scan camera (LVM 27) (f) Coaxial illumination and viewing. Light effectively comes from the camera (LVM 15)

measured automatically, very quickly and easily. The result is a diagram akin to the maps of stars in the night sky and called an *Illumination Diagram*. This and the related *Archibald Diagram* are explained in ➋ *Fig. 8.1*. Although they are more complicated, Archibald Diagrams are able to detect the possibility of specular reflection (glinting) better than Illumination Diagrams. However, to do this, an Archibald Diagram has to be drawn for each surface angle. For this reason, Illumination Diagrams are normally preferred. Six sample Illumination Diagrams are shown in ➋ *Fig. 8.2*. Others are shown in ➋ Chap. 40.

8.9 About the LVM Catalogue (➋ Chap. 40)

The entries in the catalogue should be regarded as being merely prompts guiding the practitioner towards a few possible solutions for a given application. Having thus identified a set of plausible suggestions, he/she can then concentrate on evaluating them experimentally, to find the best one.

The principal LVMs are presented in ➋ Chap. 40 in general non-quantitative form, without reference to physical scale, or cost. In some cases, proprietary lighting units are available to implement these ideas.

The LVMs can be used singly, or in various combinations. For example, it is possible to perform a useful inspection function using simultaneous back illumination at one wavelength and front illumination at another (LVM 20). Another example is to combine range maps and colour images, or x-rays and UV-VIS fluorescence. Clearly, we cannot list all possible combinations and, in practice, a great deal of experimentation and cunning is required, to obtain the best results.

The LVMs are presented in alphabetical order, since it is not possible to define any useful and more "natural" ordering.

Light sources are represented graphically, using icons suggesting the use of filament lamps. It should be understood, however, that in many cases, other light sources, such as discharge tubes, compact fluorescent lamps and LEDs might be more appropriate.

Further Reading

Batchelor BG (1982) A laboratory-based approach for designing automated inspection systems. In: Proceedings of the international workshop on industrial applications of machine vision, Research Triangle, May 1982. IEEE Computer Society, pp 80–86

Batchelor BG (1991) Tools for designing industrial vision systems. In: SPIE conference on machine vision systems integration, Boston, 6–7 November 1990. SPIE, vol. CR36, pp 138–175, ISBN 0-8194-0471-3

Batchelor BG (1994) HyperCard lighting advisor. In: Proceedings of the conference on machine vision applications, architectures and systems III, Boston, November 1994. SPIE, Bellingham, vol. 2347, pp 180–188, ISBN 0-8194-1682-7

Batchelor BG, Steel AK (1985) A flexible inspection cell. In: Proceedings of the international conference on automation in manufacturing, Part 4: automated vision systems, Singapore, September 1985. Singapore Exhibition, pp 108–134

Batchelor BG, Whelan PF (eds) (1994) Industrial vision systems. In: SPIE Milestone Series, vol. MS 97. SPIE – The International Society for Optical Engineering, Bellingham, ISBN 0-8194-1580-4

Batchelor BG, Daley MW, Hitchell RJ, Hunter GJ, Jones GE, Karantalis G (1999) Remotely operated prototyping environment for automated visual inspection. In: Proceedings, IMVIP'99: Irish machine vision and image processing conference 1999, Dublin City University, Dublin

9 Laser Scanners

Andrew K. Forrest
London, East Finchley, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_9,

Abstract: This chapter gives an overview of why scanning systems are used rather than more straightforward imaging systems. Different classes of scanner and optical systems are described along with the tradeoffs in performance that have to be made when balancing different performance requirements. Detailed performance of a particular parabolic mirror, large area scanner is presented.

9.1 Introduction

Two dimensional detectors are ubiquitous and relatively cheap. CCD or CMOS cameras are used for a large proportion of machine vision applications. Frame and field rates of these cameras are usually in the range 50–120 Hz and resolution is usually less than 1 Mega-pixel. For applications for which these specifications are not a limitation there are a wide range of cameras with standard lens fittings e.g. C-mount, and video interfaces to PCI cards to interface with a desktop computer. Software to control these cameras is equally easy to find. To obtain cameras outside these specifications gets much more difficult and costly. High speed cameras are available but require special interfaces. If the data rate is very high, separate memory may have to be used to obtain the speed required. Higher spatial resolution is equally difficult to obtain. Higher resolution implies a larger semiconductor chip size with consequent lower yields. Worse the detector will probably not be in production anyway because the volume of potential sales does not justify the high one off cost of a new chip design. This limitation has been circumvented by using multiple chips, especially in astronomy, but there are severe technical difficulties in mounting the chips close together and with enough accuracy to use the resulting data on a single rectangular grid.

It is in these slightly unusual circumstances that the use of a scanner may give the best combination of cost and performance, although they will always be a bit more difficult to use than a simple imaging system.

A scanner works by having fewer detectors than is required in the image, but moving either the image, the object or the detectors so that the scanner obtains the data sequentially. In the most extreme case only one detector is needed. In some circumstances, such as the far infrared, it is impossible at the current state of technology to build arrays of detectors. The scanner therefore is the only way to obtain high resolution images at these wavelengths.

There are very many permutations of scanner possible. Resolution, detector speed, detector efficiency, and the availability or otherwise of multiple detectors all affect the optimum design for a particular imaging system. This chapter attempts to group the possible solutions into some kind of order although there are probably exceptions to every generalisation one can make about scanner design.

9.1.1 Advantages

Scanners reduce the number of detectors needed from the usual one or more per pixel. A typical colour CCD camera uses three detectors per pixel. Reduction of the number of detectors has hidden benefits as well as the obvious ones. If for example one detector is used there are few calibration problems. The detector is the same for every pixel so detector noise, efficiency etc. is also the same for every pixel. Fewer detectors also means less electronics.

Two dimensional and even one-dimensional i.e. line detectors lose efficiency due to the areas between detectors which are insensitive to light. With modern Detective Quantum Efficiencies (DQE) of better than 50% this factor can become quite significant. A scanner can be made to use closely fitting pixels or even overlapping pixels if detection is more important than imaging resolution.

The characteristics of a scanner e.g. resolution, imaging speed, etc. can be varied by large factors without changes to hardware. A typical CCD camera will have a fixed or very nearly fixed frame rate and spatial resolution.

The detector(s) can be sensitive all of the time. Two dimensional arrays typically have an imaging and data read out phase of operation. Typical scanner detectors run continuously.

If only part of the final image is being measured at any one time then only that part of the object needs to be illuminated. This allows very bright laser illumination to produce fluorescence for example. This idea can be extended, and usually is, so that the spatial resolution of the device is determined by the illuminated spot rather than the size and optics of the detector. It is relatively easy to produce a small illuminated spot compared to producing the same resolution over a large two dimensional field. A scanner is therefore the best way to produce very high resolution images. To take an extreme example, atomic force microscopes are essentially single detector scanners using various physical effects on a single detector to obtain resolutions down to atomic diameters.

Resolution is usually the reason that scanners are used. This may not be just resolution of very small features but be resolution of relatively large features but on very large objects. This is often the case in material production, where glass or aluminium sheet of several metres width is monitored for flaws during production.

If, as in the above example, the surface to be imaged is moving in a known manner, then only sequential one dimensional images need to be acquired to get a full two-dimensional image. There are then no problems with images overlapping or leaving gaps, or relating one image to the next.

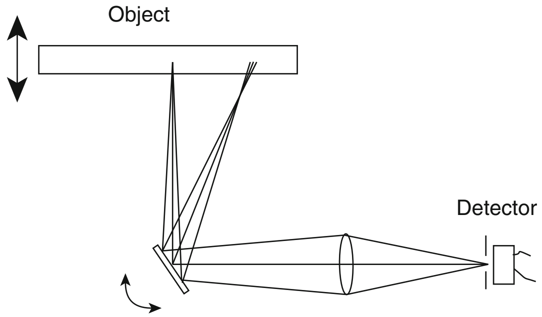
9.1.2 Disadvantages

The main disadvantage of scanning is that some moving parts are required. This must be true if the number of detectors is smaller than the number of pixels in the image. How this is achieved is a major part of scanner design. There is currently no good way to steer optical beams without moving parts. The closest is using opto-acoustic crystals which can be electrically excited to diffract light, but the angular deflection available is small. The only scanners that don't need moving parts use line detectors which are moved bodily (see next section).

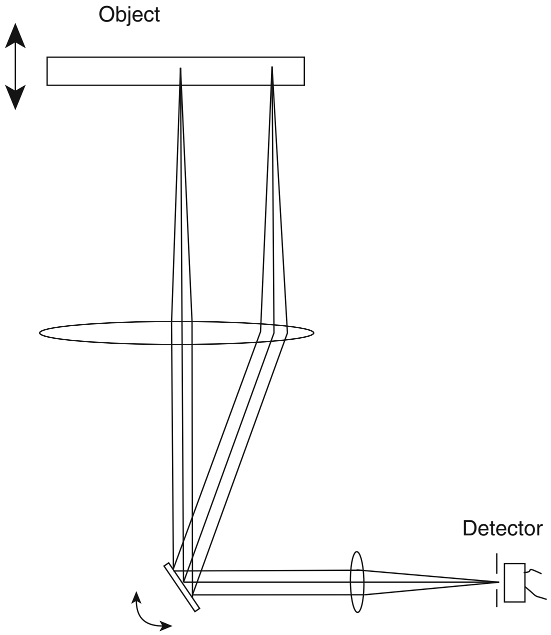
9.2 Types of Scanner

9.2.1 Resolution by Imaging

The very simplest form of imaging would be to move a detector over the surface of interest in a two dimensional raster. This has very little to recommend it. The next simplest is the arrangement shown in ➋ *Fig. 9.1*. The surface is illuminated over the entire range of



■ **Fig. 9.1**
The simplest kind of scanner sometimes called a post-objective scanner



■ **Fig. 9.2**
A pre-objective scanner which requires an optical component at least the size of the scan

the scan. Points on the scan are sequentially imaged onto a detector. The size of the stop at the detector sets the spatial resolution. The imaged area is scanned across the object by rotating a mirror.

The arrangement has a serious drawback; the distance from lens to surface is not constant through the scan so the system cannot be in perfect focus for the entire scan. A lesser problem is that the angle at which the surface is imaged changes with scan position which will again give differing results across the scan. Both of these drawbacks can be removed by using the configuration in *Fig. 9.2*. This is sometimes called pre-objective scanning as opposed to the system in *Fig. 9.1*, which is post-objective scanning. The drawback of this system is that a lens of the same diameter as the object being imaged is needed. If the large lens is at its focal length from the surface then perfect imaging can be obtained and the angle at which the surface is scanned stays constant for the entire scan. The above assumes, of course that the lenses are perfect and less obviously that the diameter of the beam illuminating the large lens approaches zero. This is because for a perfect lens rays emanating from the centre of the scanner mirror will all end up parallel as they approach the surface. This is not true of the rays not emanating from the centre of the mirror. The other optical limitation is connected to the first. The angular resolution of an optical system can only be approximated by

$$R = 1.2\ \text{Lambda}/d$$

where d is the diameter of the beam and Lambda is its wavelength. This is a direct consequence of the wave nature of light so can only be alleviated by using shorter, that is bluer, wavelengths. This requirement is in direct opposition to the previous requirement for perfect imaging that the beam be vanishingly narrow.

The interplay between these two constraints and the necessity for at least one large curved optical component have largely been responsible for the many permutations of scanner that have been developed.

9.2.2 Resolution by Spot Projection

An alternative to illuminating the entire scan and imaging a part of it, is to illuminate one spot on the scan at a time and have a detector that collects light equally from the entire length of the scan. In this case the detectors in *Figs. 9.1* and *9.2* would be replaced by a light source and the illumination system by a detector.

Where resolution is by spot projection, in most cases, the light source will be a laser of some sort. This optical arrangement allows the entire output of the laser to illuminate one spot on the object at a time. This is very efficient in use of laser power, and it is relatively easy to get a very bright spot which can be useful in specialised cases such as stimulating fluorescence. The drawback is that a large detector is necessary. In any optical system the luminosity is conserved or made worse

$$wA = \text{constant}$$

or in words, the product of solid angle and area (of light beam) is constant. This therefore implies that to detect a large proportion of the light coming from a large area of an object the detector must also be correspondingly large. The most efficient way to use detector area is to illuminate it with as large a range of angles as possible, but of course a hemisphere is the largest solid angle with which one can illuminate a flat detector. However clever the optics, there is a limit to how efficient the detector system can be made. *Figure 9.3* shows a combined system where the same optics are used for illumination and light detection. The semi-silvered mirror should be 50% reflecting and transmitting for best signal, which results in a loss of three quarters of the available light. The focal ratio or F number of the beams in this system are also high, so although the arrangement looks attractive it is not very efficient.

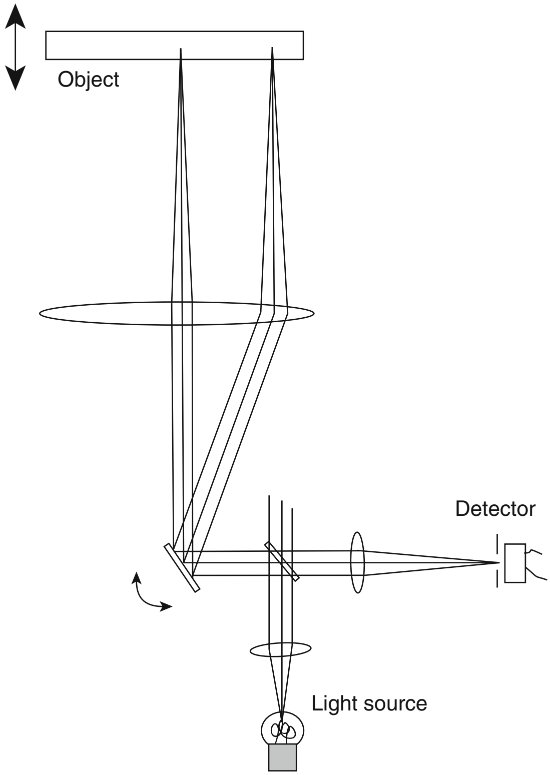


Fig. 9.3
A pre-objective scanner. Note the system for using the same optics for spot projection and imaging

9.3 Uses

9.3.1 Input

9.3.1.1 Document Reading

This is one of the most common use of scanners. They are normally of "flat bed" form, that is the document is placed on a flat sheet of glass to be scanned. These scanners leave the document stationary and move the whole scanning apparatus across the document. The entire scan length is illuminated at all times, and resolution is obtained by a stop affecting the detector.

Scanners of this type often have very impressive optical performance, especially considering their low cost. Resolution of 5,000 lines per inch can be obtained. The use of inches in this context is almost universal. The example above equates to about 200 lines/mm. This performance is achieved by the use of plastic aspheric lenses. The aberrations mentioned in the introduction can be carefully balanced. Unit costs of plastic lenses are very low, but the one off costs of producing the high-pressure forming tools are high. These kind of lenses are therefore only available to volume manufacturers. It should also be noted that the quoted resolution is not at a contrast of 100%. It is often not stated what contrast has been used for the measurement, so actual resolution may not be quite as good as expected. The other performance criteria usually used is the dynamic range of the intensity response. This is usually quoted as a power of ten, i.e. 3.3 means an intensity range of 2,000 or $10^{3.3}$. A large dynamic range is needed to get all the information in pictures, and especially when film is being scanned. High dynamic range requires the removal of as much stray light as possible by clever design, baffles etc. and by anti-reflection coating the optical components. This can raise production costs significantly so the difference between budget and premium scanners can often be seen best in their dynamic ranges.

It is difficult to design a scanner that gives equally bright images across the length of the scan. Typically the image will darken at the ends of the scan. This is usually combated by having the scanner scan a patch of reflective or clear glass before the scan proper starts. This allows the scanner to adjust the output intensity values in the rest of the scan to give an even response.

Even with the techniques above, it is difficult to get very high resolutions over a large document. In a single scan, possibly 10,000 elements can be obtained but 100,000 cannot. This problem is removed in some large flatbed scanners by scanning the document in 'swathes'. A bit like mowing a lawn in strips. The strips are then joined together using software so that the user is not conscious of the operation. Another approach is to have a lens system that can be changed to give very high resolution over a small scan for scanning film for example or a lower resolution over a bigger scan for paper.

9.3.1.2 Bar Codes

Bar code scanners are probably the most numerous scanners in existence. They range from relatively simple devices to extremely clever and difficult systems. An example of a simple system is that used by check out counters in shops. The scanner is placed against a bar code of the correct size and measures the width of each bar resulting in a binary number denoting the object. The cheapest of these are not actually scanners but linear array imaging devices. At the other end of the range are systems used to measure bar coded tags on luggage as the luggage passes on a conveyor. This needs to be accomplished unattended. This is usually done by using a large number of scanning beams at different angles and positions such that at least one beam will cross a bar code at an angle at which reading it is possible. The signature of a beam crossing a barcode must be separated from ambient light, flickering fluorescent lamps, etc. Details of the modulation and detection circuitry used are proprietary secrets. The best way to find out about these system is the drawings made public in patent applications (see ➲ *Fig. 9.20*).

9.3.1.3 Film to Video

This is a commercially important area as data storage takes over from analogue media. It is not particularly difficult in optical terms because of the relatively small number of pixels in video image. Scans should however be done at higher than the output resolution. The film image can be processed to improve tonal characteristics, colour and even remove blemishes such as dust and scratches. Scratches and dust can be preferentially imaged using near infrared and this information used to adjust the RGB channels.

9.3.1.4 Film to Still

Digital cameras are beginning to take over from film-based cameras, but where very high resolution is needed, film is still superior. A CCD imager with ten million pixels is now common, but a 4 × 5 in. sheet of film digitised in colour at 5,000 lines per inch gives 500 million pixels of each primary colour. Very high-resolution CCDs are now available but are very expensive and still cannot match these numbers. Because ultimate performance is the reason for using film in these situations, the film is normally scanned with a drum scanner, see ➋ Sect. 9.6. These scanners have the best optical performance of any scanner, and the fact that the scanned film must be bent for operation of the scanner is not a drawback.

9.3.1.5 Quality Control

Some products lend themselves particularly well to imaging using a scanner, especially products produced as a web, i.e. a continuous sheet. By imaging a line across the web as it is produced, the entire area is covered without any problems of overlap etc. Particular cases are float glass, aluminium sheet and various plastic sheeting materials. These all have the advantage that they have very flat and reasonably shiny surfaces. A blemish will depart from the flatness and reflectivity of the correct material. A scanner of the type in ➋ *Fig. 9.3* can be effective even without the large lens. The scanner can be arranged at an angle to the surface so that reflected light for a good surface does not reflect back to the detector. A blemish however will reflect light. This arrangement has the advantage that for a good surface the 'image' is black. Blemishes appear as lighter areas. This makes image processing extremely easy and also allows the scanner to detect blemishes smaller than its optical resolution. Even a blemish smaller than the resolution limit will reflect light which can be detected.

A property of scanners that is often overlooked is that the optics can be at an angle to the scanned surface. This will produce images which are similar to the ones one would see looking at the surface from that angle but without the geometrical distortion caused by perspective. Many blemishes, such as scratches, are much easier to see both when obliquely illuminated and obliquely viewed. In general, surfaces get more reflective at oblique angles. A scanner can accomplish this while maintaining the shape of the object in the resultant image.

9.3.2 Remote Sensing, Military

Multi-element detectors for visible wavelengths are now easy to obtain, but infrared and ultraviolet are still problematic. Scanners have the characteristic of reducing the number of detectors required, sometimes to just one, so that extreme wavelengths can be used for imaging. Another aspect of this is that the beam coming from the scanner can be separated into separate wavelengths and separate detectors. In effect, an entire spectrum can be obtained from each point in the image. This characteristic can be used to detect very subtle effects such as diseased plants from satellite orbit. The military possibilities are obvious. A major problem with such systems is the combination of high spatial and high spectral resolution creates very large quantities of data.

➋ *Figure 9.4* show a reconnaissance imaging system using a line detector and optics placed using the Schiempflug condition [6] to map a swathe of land. This is a rare example of a scanner with no moving parts, unless you count the aircraft.

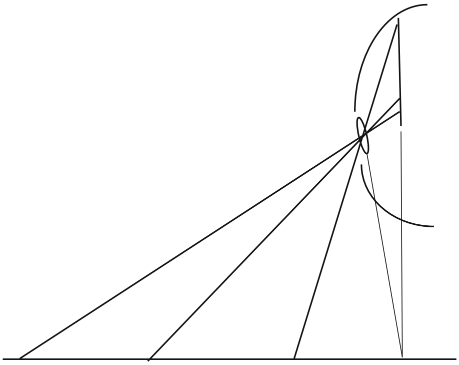


Fig. 9.4
Surveillance or mapping scans taken by flying a plane across an area. The only moving part is the plane. Imaging conforms to Schiempflug condition [6]

9.3.3 Three Dimensional

Three dimensional scanning depends on parallax between views or in other words stereo. As long as a unique point in space can be viewed from two separate positions the coordinates of the point can be calculated by geometry using the angles to the point from the two viewing positions. Although the idea is conceptually simple there are great practical difficulties. The worst problem is correspondence i.e. viewing the same point in the second view corresponding to the point chosen in the first view. Obtaining this correspondence in the general case from two randomly chosen viewpoints is very unreliable. The design of three dimensional scanners and indeed stereo systems is largely composed of techniques to make this matching of points reliable.

The correspondence problem is difficult because there is no reliable way to tell if a match has been made correctly between two views. More constraints are needed to allow self consistency checks, or to put it another way add some redundancy to the system. In stereo systems this can be accomplished by using a third viewpoint. If all three views agree on a three dimensional position for a point then there is little chance that the match is incorrect.

In the case of scanners a typical solution to this problem is to constrain the geometry of the setup so that matching is unequivocal. An example is shown in *Fig. 9.5*. A line of light is imaged on the object surface. The surface is then viewed by a camera in the same plane as the projection system. The horizontal displacement in the image is a measure of the position of the projected light on the surface. If a one dimensional detector placed along the plane of the optics is used the position of detector with largest response measures the position of the object surface. By rotating the object, a complete perimeter of the object can be measured. By moving the whole optical arrangement up and down a complete surface can be generated. This can also be achieved by just moving the linear detector up and down.

The weakness of this kind of system is that it requires that the directions of the optics, images and rotations all be accurately aligned. The ultimate resolution of this kind of system is that corresponding to one pixel of the detector. This is typically 10 μ across. Setting the rest of the system up to this accuracy is almost impossibly difficult. An alternative using two-dimensional detectors is shown in *Fig. 9.6*. This system uses three optical systems, two for

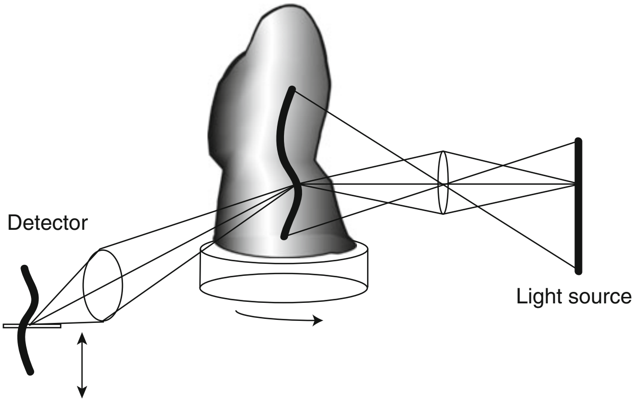


Fig. 9.5
Three-dimensional scanner using scanner geometry as reference

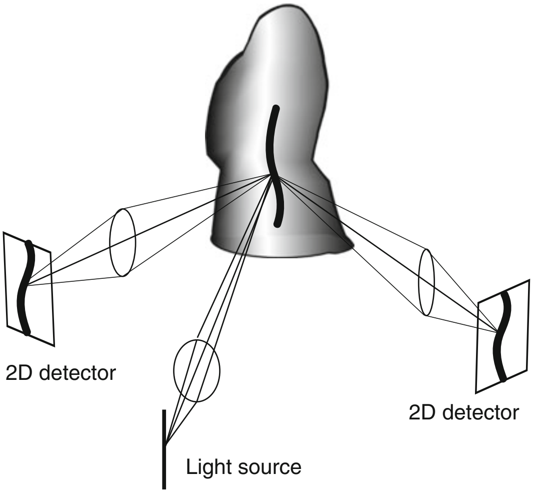


Fig. 9.6
Three-dimensional scanner using self-calibration as reference

imaging and one for light projection. The light projection system illuminates just one point on the projected line at a time. The benefit of this arrangement is that it is completely general in the sense that whatever the relative angles of the cameras and projection system, if the cameras can see the projected spot, then an accurate calculation of spot position can be made. This presupposes that the system has been accurately calibrated by viewing known, accurate three-dimensional positions.

The two systems, although superficially similar, are based on two different premises. The system in *Fig. 9.5* depends on the accurate physical construction of the mechanical and optical components. The system in *Fig. 9.6* depends on accurately measuring the characteristics of the optical system using known positions. It could be argued that the *Fig. 9.5* system could also be calibrated in this way, but for calibration to be successful all the variables affecting the system must be taken into account by the calibration. This is almost certainly impossible with *Fig. 9.5*. In *Fig. 9.6*, the calibration depends entirely on getting the characteristics of the two cameras. This is relatively easy if the cameras geometry is completely linear, i.e. acts like a pinhole camera. Even if this is not the case, there is considerable literature on obtaining non-linearities by observing known objects.

The area of three dimensional scanners is broad one. It has been used for the accurate measurement of turbine blades, reverse engineering mechanical components for which plans are not available to measuring the shape of people to obtain optimum dress sizes and measure drifts in population characteristics. The permutation of optical arrangements is almost limitless, however systems that can be rigorously calibrated are in my experience more likely to be effective than those that depend on accurate construction.

9.4 Output

9.4.1 Laser Printers

Laser printers typically work by writing the image on a drum and then printing the resulting image onto paper. The drum is electrostatically charged but covered in a photoconductor. Exposing any part of the drum to light allows the charge for that area to leak away. If the drum is exposed to very finely divided carbon, the toner, the toner will stick to the charged parts of the drum only. By exposing the drum to the inverse image we want printed i.e. light exposed gives white, we can produce a convenient form of printing.

A scanner is used to expose the drum. For an A4 printer of 1,200 dpi, there are approximately 10,000 dots across the width of the page. This is about the limit for reasonably priced scanners. This is a scanner using resolution by spot projection as already discussed but without the need for a detector.

9.4.2 Image Setters

Image setting is the process of turning digital data describing images or text into a mask or printing plate that can be used for high volume printing. Quality of image is usually a prime concern. Glossy magazines and books are produced in this way. Time to produce a plate is not critical. Most image setters therefore use a drum scanner. The mask or plate is wrapped around a drum which is rotated at high speed. A writing head moves linearly down the drum writing the image. See the section on drum scanners. The resolution is very high because of the easy constraints on the optics, but mechanical accuracy has to match the optical performance.

9.4.3 Semiconductors

This is a very specialised use of scanning. Some of the images produced probably have the best resolution of any images ever produced. Techniques are being developed all the time to produce finer resolution circuit elements which allow more gates and therefore a better or faster computer chip. Traditionally a mask for a particular evaporation or doping process would be written using a high resolution image setter to produce a mask approximately 10 times the size required. This mask is then imaged at the correct size onto a photosensitive surface using extremely high-quality optics and ultraviolet light. Ultraviolet light is convenient for the exposure of the photo sensitive surface but, more importantly, the short wavelength allows finer detail. The resolution of any optical system is limited by the wavelength being used.

Even ultraviolet is now too long wavelength for some uses, so electron beam scanners which have a much smaller wavelength are used instead. This is the same kind of technology used in scanning electron microscopes. Because they are charged, electron beams can be steered and focussed by electric fields, but this is taking us outside the range of this chapter.

9.4.4 Printed Circuit Boards

The masks are produced in much the same way as for film image setting. The required resolution even for small surface mount components is quite easy to obtain.

9.5 Performance

9.5.1 Modulation Transfer Function

Probably the best way to express performance is the Modulation Transfer Function (MTF). This plots the contrast for a sine wave intensity image viewed by the instrument. An example is shown in ➋ *Fig. 9.18*. Note that even a perfect optical system does not give 100% contrast for anything other than very low spatial frequencies. Quoting a dots per inch or lines per millimetre figure is really not useful without saying at what contrast this spatial frequency is imaged. A typical assumption is that contrast is 30%, but this is by no means guaranteed. Given the MTF, it is quite easy to simulate the acquired image for a given perfect image and hence decide on performance issues.

9.5.2 Scan Length/Speed

These terms are mainly self-explanatory, although scan speed can be misleading. Most systems require a gap between traverses due to the scanning technique see ➋ Sect. 9.8, so the spot speed is likely to be higher than the scans per second would suggest. See ➋ *Fig. 9.12* for a way of reducing dead time by using two input beams to a rotating prism scanner. Depending on the scanning system, the scan can by mono or bidirectional.

9.5.3 Number of Channels Out

By the nature of scanners the number of channels out can be large. Total data rates can be extremely large.

9.5.4 Linearity, Aberrations

We have been assuming that the scanner will produce at a constant rate of movement of spot or image position. The angular movement of the scanner mirror will not produce a linear scan unless great care is taken. The optics are unlikely to produce a constant spot movement for a given change in mirror angle. Lenses can be designed to approach this and are sometimes called f-theta lens. Scanners with galvonometric scan mirrors (Sect. 9.9) will not move at a perfectly constant rate to give a triangular angle versus time response, especially as the maximum speed of the scanner is approached.

Many scanners use mirrors for the main optical components. This implies in many cases that they must be used slightly off-axis. This results in the nominal scan line actually being slightly curved. This translates to a scanned perfectly straight line appearing to be slightly curved. The spot size of the imaging or projection system will vary over the length of the scan giving a non-constant resolution or MTF. Optical efficiency may also vary for different positions on the scan giving apparent brightness changes across a scan. Chromatic aberrations are also possible for systems that use transmitting optical components as well as mirrors.

Some scanners have very high performance in some aspects such as resolution, but it is important to note that much of the published work on scanners is on ways of balancing imperfections in different aspects of performance to obtain optimum total performance.

9.5.5 Dead Time

Most scanners cannot produce a spot on the scan line 100% of the time. If a mirror needs to have its rotation direction reversed, this will take time. Similarly, rotating prism scanners do not produce a spot on the scan line all the time see Sect. 9.9.

9.5.6 Contrast, Artefacts

The dynamic range of a scan is a very important attribute, especially if amplitude information is being used. Decreasing stray light involves baffles, anti-reflective coatings on optical surfaces and careful design. Getting very good performance is hard, for example light can bounce or scatter from the ground periphery of circular optical components, see *Fig. 9.7*. This is very difficult to reduce. The best cure is to prevent it happening by judicious use of field and aperture stops.

A particular problem with scanners is that there may be scattered light at a few discrete angles of scan that find their way to the detector. This will result in streaks in the image produced which are very visible and hard to remove.

Scanning prism systems can produce a variety of unwanted effects. If the angle of each facet is not identical or the rotational axis does not go through the centre of symmetry of the polygon scan paths can cross. This results in obvious artefacts in the image, especially in features parallel to the scan direction. Other types of scanners using lead screws are also prone to this kind of error. The easiest way to avoid these problems is to have identical optical and electrical characteristics for each scan line, e.g. galvanometric scanning in one direction only and discard the return scan.

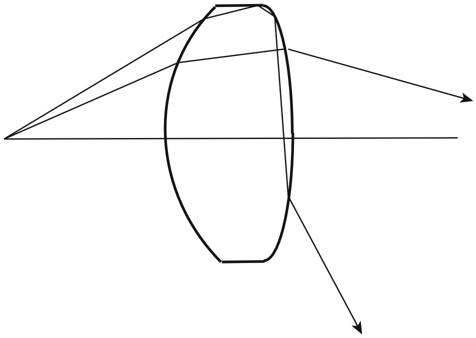


Fig. 9.7
Shows how light may be internally reflected in optical components, reducing dynamic range and contrast

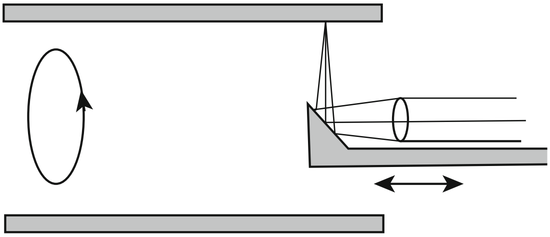


Fig. 9.8
Configuration of basic drum scanner using internal scanning

9.6 Scanner Configurations

9.6.1 Internal Drum

The basic internal drum scanner is shown in *Fig. 9.8*. Note that the lens is attached to the moving carriage. The beam will traverse the inside of the drum in a continuous spiral as the drum rotates and the carriage moves through the drum. There are many permutations of this arrangement. If the light reflecting back from the internal surface of the drum is collected as in *Fig. 9.3*, then this diagram is complete. For transparent sheets the drum needs to be transparent and a detector can be placed on the outside of the drum picking up the light from the spot focussed on the interior of the drum.

In this example, the film or document is placed on the inside of the drum, and just centrifugal forces may be enough to keep the document flat against the internal surface of the drum. Conversely, the document or transparency can be placed on the outside of the drum. This makes loading a document slightly easier, but the document must be held in place possibly by electrostatic forces produced by charging the drum.

The above systems use a rotating drum. It is possible to rotate the beam around the inside of the drum possibly using a rotating pentaprism as in *Fig. 9.9*. This will obviously only work

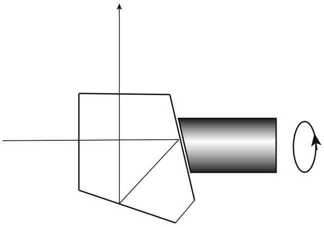


Fig. 9.9
Alternative scanning head for internal drum scanner

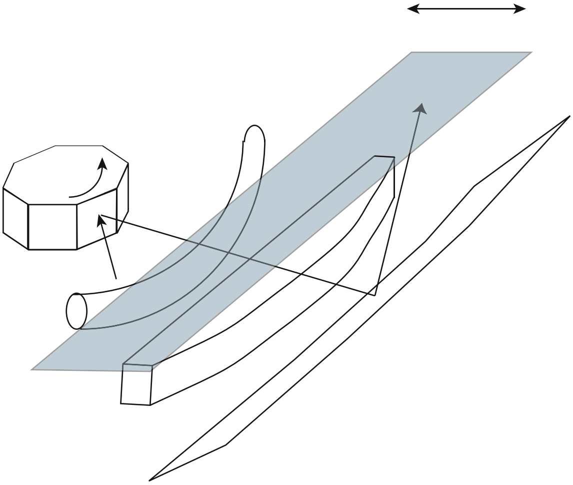


Fig. 9.10
Schematic of typical flat bed scanner showing use of aspheric plastic optics

for a detector which uses the projection optical path as in *Fig. 9.3*. The advantage of this arrangement is that the only moving component is small and light which allows high-speed scans. The fast moving part is also safely contained within the drum. The advantage of the drum scanner is that the optics work at a fixed angle and position with a very small field of view allowing very good resolution.

9.6.2 Flatbed

A typical flatbed scanner configuration is shown in *Fig. 9.10*. The beam is steered by a rotating prism. The optics are all aspheric starting with a bent cylinder going on to aspheric transmissive components and an aspheric mirror. The figure shows resolution by projecting a spot of light on the surface but typically the entire width of the surface would be illuminated, and the field of view of the detector would be scanned across the surface.

9.7 Light Sources

For systems that use imaging for resolution rather than spot projection, the most common illumination is either high frequency fluorescent or multiple LED. Fluorescents have a convenient shape, are efficient and can have a tailored output with wavelength. It is of course very important that the brightness variations do not interact with the imaging. Domestic fluorescents for example can flicker at 100 Hz. The fluorescents in this use are driven at a frequency that is higher than the relaxation time of the phosphor. This ensures constant output.

Light emitting diodes are also very efficient and easy to drive. For a domestic colour scanner the colours of the illuminating diodes can be matched to the response of the red, green and blue detectors giving accurate perceived colour rendition. Multiple LEDs are used rather than trying to use optics to spread the light over the line of the scan.

Scanners whose resolution depends on the projected spot typically use lasers. These produce a high luminosity beam which is relatively easy and efficient to concentrate as a spot on the object. Laser diodes are now relatively cheap and are an obvious choice for this application. There are two problems for this use however. Laser diodes produce an asymmetric beam profile which is usually unhelpful. Usually cylindrical lenses or a pair of prisms, as in ➲ *Fig. 9.11*, are used to make the beam more symmetric, although by conservation of luminosity, all characteristics of the beam cannot be made symmetric at once. The other problem is that the drive electronics for a diode of medium or high power are not simple. Laser diodes tend to exhibit negative resistance characteristics and should be driven with an approximately constant current source. In some cases, a resonant drive circuit is used which results in the diode switching on and off at several Kilohertz. These are obviously not suitable for use in a scanner. Laser diodes are normally used at near their maximum ratings, requiring good accuracy and reliability from the drive electronics.

9.8 Detectors

Scanners are often selected so that an unusual detector can be used to create images. There is essentially no limit to the detectors that can be used on scanners because even a single detector can be made to produce images. Having said that, there are some common threads in the choice of scanner detectors. The ability to image from a small number of detectors is bought at the expense of time division multiplexing the detector. This implies that the detector must operate rapidly. Most scanners are not very optically efficient; that is they operate at high *F* numbers. Comparing a CCD two-dimensional 1,000 × 1,000 imaging system with a *F*1.4 lens

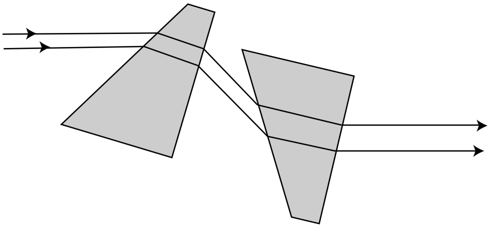


Fig. 9.11
Two prisms used to make diode output approach axial symmetry

to a scanner using one detector and a typical *F* number of 40, the difference in light collected per unit time is obvious. This implies that the scanner detector as well as being fast must also be very sensitive.

Photodiodes can be used as scanner detectors, but it is difficult to arrange good efficiency above about 2 MHz bandwidth. This is because the self-capacitance of the diode limits the bandwidth see [4]. A modern alternative is the avalanche photodiode or APD. The inherent gain of these devices, around 10–1,000 in most cases, allows higher speed operation. Photomultipliers, although not being solid state, have perfect characteristics for scanner use. They have high photon-multiplying gain, up to 10^6, and also very good frequency response into the 100 MHz range. They also have good quantum efficiency. Modern devices can also be quite small and robust, although they do require a high voltage, but even APDs require a high bias voltage, although not as high as a photomultiplier.

Flat bed scanners often use line detectors to speed operation. Line detectors of 10,000 pixels are now available. This can result in a system without a scanning mirror where the surface is just sequentially imaged onto the line detector by moving the detector and its optics across the surface. The movement is normally made using a stepper motor. This results in a system which is half way between a normal two-dimensional imaging system and a scanner. Line detectors with large numbers of elements are usually based on CCD or CMOS technology. Small line detectors, i.e. 16 elements, are often photodiodes or APDs.

9.9 Beam Deflection

9.9.1 Galvonometric

Galvonometric scanners work in the same way as an analogue panel meter. They are essentially a small permanent magnet direct current motor, but constrained to move less than half a revolution and therefore not requiring commutation. They are extremely easy to use; deflection being proportional to applied current. The axis of the rotation can be made coincident with the surface of the scanning mirror making the optical design simpler. Two systems can be put in series to produce XY scanning.

The main drawback of this type of scanner is speed. This limitation gets worse as the mirrors size and weight increases. A small gain in speed can be accomplished by making the scanner resonant. This has the attendant limitation of working only over a very narrow range of frequencies and producing a sinusoidal scan.

9.9.2 Rotating Prism

A rotating prism is a convenient way of obtaining rapid scanning. During operation no accelerations are needed removing problems of vibration, speed of response etc. The rotating prism also has some disadvantages. When housed in a protective shield the construction is similar to a siren. The rate of facet arrival at is likely to be in the kilohertz range. They can therefore be very noisy. Because there is more than one facet, usually 8 or 10, each scan line does not use the same optics. Any variations or inaccuracies in the facets will produce visible results in the scan. The alignment of rotation axis with facets is also critical. Because the rotation speed is constant it is not possible to linearise scans by changing the scan rate in parts of the scan.

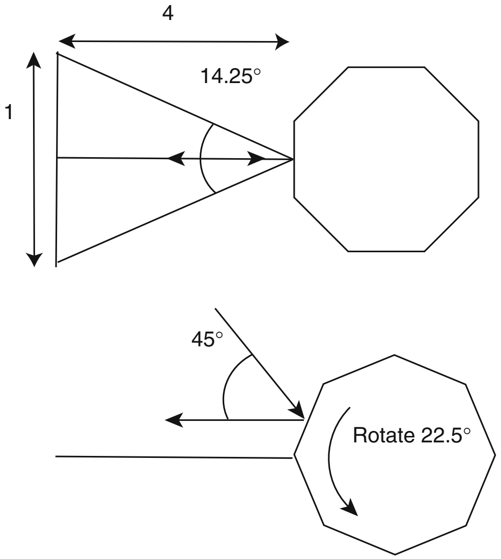


Fig. 9.12
Using two input beams to a polygonal scanner to obtain better time efficiency

The optics of this type of scanner are difficult to handle. The faces of the prism do not rotate about the point that the principle ray hits the surface. This will result in a beam displacement as well as change in angle. This needs to be included in the modelling of the system. The final problem is that they are inefficient in the sense that for most optical set ups the beam will be traversing the wanted range of angles for a relatively small proportion of the time.

The system in ➋ *Fig. 9.12* is *F*4, so the total scan angle will be 14.25° (tan $\theta/2 = 1/8$). To obtain this angle, the mirror will only need to move 7.125°. If the prism has 8 sides, then the angle between prisms is 360/8 or 45°. Every facet will therefore be useful for 7.125/45 or 15.8% of the time. In this case, the scanner is operational for less than one sixth of the time.

The above figures can be made better by increasing the number of facets which reduces the size of each facet. The entire prism thus needs to be larger with consequential increase in cost. An alternative is to illuminate more than one facet from different angles. This technique is also shown in ➋ *Fig. 9.12*. In the top diagram, the light is projected on the prism horizontally from the left and is reflected back horizontally. When the prism has rotated half a facet we want the output beam to again be horizontal. This is shown in the bottom half of the diagram. Note that there is a vertical displacement in the beam from the first example. A similar technique can be used with three input beams. The optics of this can get complicated but is useful for low-performance scanners such bar code readers, where speed of operation is more important than ultimate resolution.

9.9.3 Rotating Wedge

A rotating wedge in the projected beam will scan the beam in a circle. It would be possible to use two contra-rotating wedges to produce an approximately linear scan (➋ *Fig. 9.14*); however, I have never seen this implemented. By superimposing the scan from two wedges, one

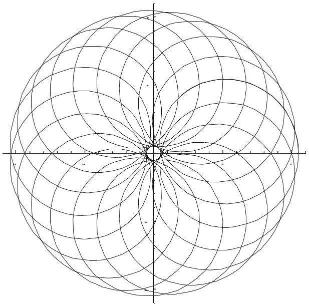


Fig. 9.13
Scan pattern from two rotating wedges in series

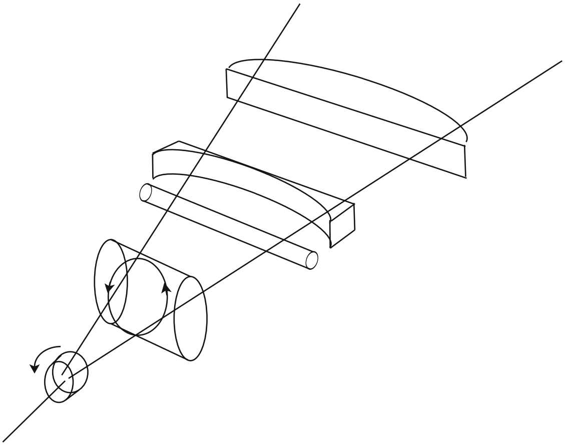


Fig. 9.14
Rotating wedge used in combination with cylindrical lenses to produce a linear scan

going 20 times as fast as the other, the scan pattern in *Fig. 9.13* can be obtained. This is a good approximation of using all angles within a solid angle and is useful for barcode reading.

9.9.4 Prismatic

Another form of scanning prism is shown in *Fig. 9.15*. This scans and changes direction and is often used in barcode scanners. This kind of prism is not generally available in small quantities.

9.9.5 Optoacoustic etc.

It would be very convenient to have a scanning system that used no moving parts; this is after all the main drawback of scanners. Two techniques have been developed to do this, but neither has been generally adopted.

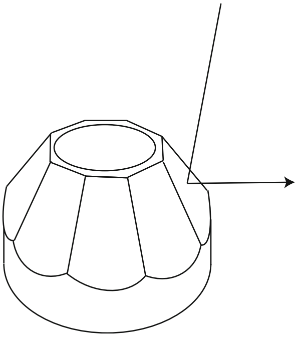


Fig. 9.15
Prismatic scanner used to scan and change beam direction

By setting up an acoustic standing wave in a crystal, one can obtain a regular variation in density and refractive index through the crystal. This can be used as a grating to deflect light of a particular wavelength. Changing the grating spacing will change the angle of deflection and this can be accomplished by changing the driving frequency causing the acoustic waves. This technique can only produce small changes in refractive index and small changes in beam deflection.

Another technique is to use an electro-optic crystal such as KDP (KH_2PO_4) to rotate the plane of polarisation of a scanning beam. The polarisation of the resulting beam can then be used to scan the angle of the resulting beam using another crystal with asymmetric structure such as calcite. Again the angles available are small, but this method can work at 1 MHz so is one of the fastest canning techniques available.

The invention of a practical and reasonably cheap scanning system without moving parts would revolutionise this area of technology.

9.10 Optics

For anything but the simplest of scanners, an optical component the size of the scan is required. The nature of this component is therefore critical to the design. Mirrors have a major advantage in this use. They do not suffer from chromatic aberration, and it is easier and cheaper to make large curved mirrors than it is to make lenses. Large lenses are also very heavy and because of the large path length through glass can be inefficient. Having decided on use of a mirror what is the best shape? In general, spherical mirrors are a poor choice because of spherical aberration. In this case, there is little chance to correct this in other optical components.

Parabolic mirrors are very nearly as easy to make as spherical ones. The differences between the two shapes are actually very small for a mirror of reasonably high *F* number. A parabolic mirror produces perfect images at infinity at the parabolic focus. To turn this around, if a beam is scanned in angle from the focus, this will result in perfectly parallel rays emerging from the mirror. This, unfortunately, is only true for an infinitely narrow beam, but it is a much better starting point than a spherical mirror. See ➲ *Fig. 9.16*. This scans a parallel pencil of beams from the parabolic focus to produce a focussed spot at the focal length. This produces a very good spot on axis but a much poorer one-off axis. In reference [5], a optimum arrangement of

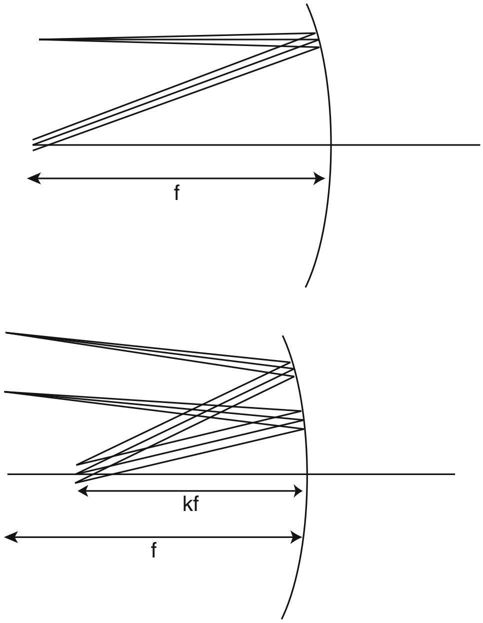


Fig. 9.16
The top is the optical arrangement for a parabolic mirror with scan mirror at its focus. The bottom arrangement discovered by [5] is usually a better compromise with *k* approx. 0.7

parabolic scanner is developed. This balances spot size and linearity of scan. The value of *k* is around 0.7. Not that in this arrangement even the principle rays are not parallel to the axis. For an actual optical diagram and performance model for a real system, see Sect. 9.11.

Note that the effective F number of most scanners is very high. If an *F*4 parabolic mirror is used in the above case, this does not mean that the light collection ability of the system is equivalent to an *F*4 camera. Only a small part of the mirror is used at any one time, resulting in an effective *F* number of 40 or 50.

The optics of drum scanners are much easier because the scanning is all done mechanically so the scanner optics only need to accurately image one on-axis point. Drum scanners therefore have a big advantage in the production of high-resolution and high dynamic range scans. They can also be made optically efficient, i.e. a low *F* number because the imaging optics are very close to the imaged surface, see *Figs. 9.8* and *9.9*.

9.11 Real Examples

9.11.1 Large Throw Scanner

Figure 9.17 shows the configuration of a parabolic scanner with a scan length of 75 cm designed for the quality control of ceramic tiles. This uses the optimisations suggested by [5] and is also operated slightly off-axis to prevent the beam scanning mirror getting in the way of the light pencil.

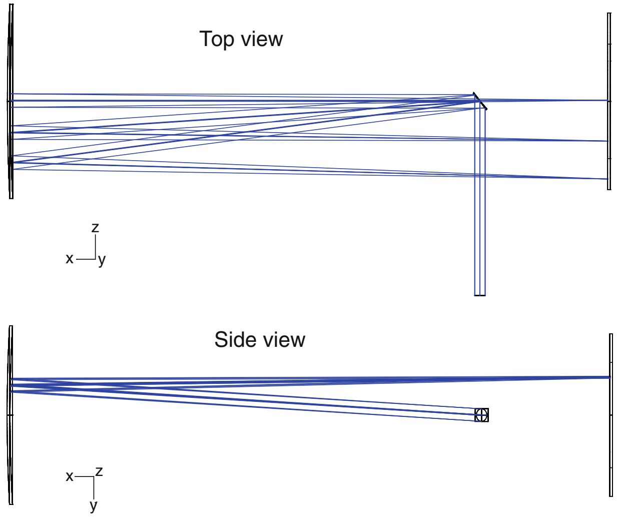


Fig. 9.17
Practical design of a scanner [3] using principles as *Fig. 9.16*. The diagram is to scale and was obtained from the output of an optical CAD programme called "Zemax"

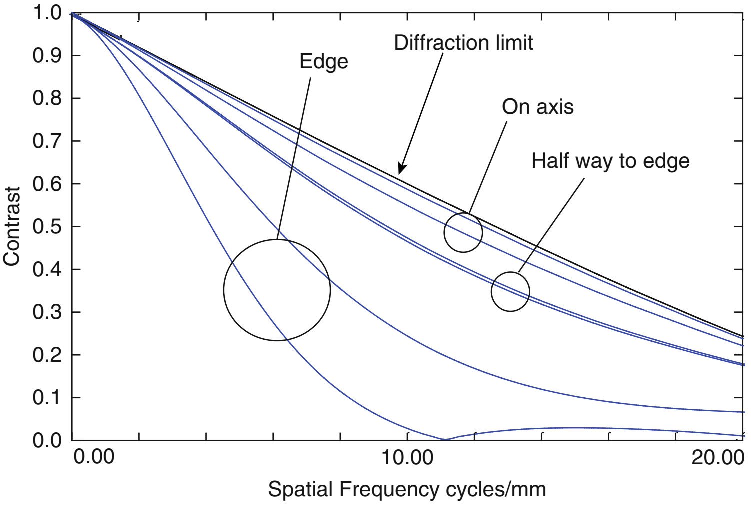


Fig. 9.18
Modulation transfer characteristics of the design in *Fig. 9.17*

The MTF is shown in *Fig. 9.18*. There are seven lines altogether. The top line is the performance of a diffraction-limited system with the same beam size. The next two lines down are the MTF horizontal and vertical of the beam half way to the edge of the scan. Note that MTF is direction dependant, and that the performance at this point is nearly as good as the theoretical limit. Next two lines down are the performance on axis, and the last two are the

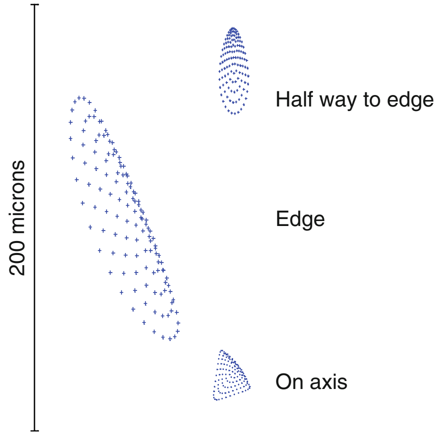


Fig. 9.19
The revolution performance from *Fig. 9.17* shown in a different way as spot size and shape at different parts of the scan

performance at the edge of the scan. The same performance is shown in a different way in *Fig. 9.19*, where the spot size and shape is shown at the same three points on the scan.

Making the beam size bigger would have improved the performance at the centre of the scan which is almost as good as theoretical but decreased the performance at the edge of the scan. If a different scan length or different *F* number mirror were used, the trade-offs of the design would all be subtly different.

9.11.2 Barcode Reader

Getting details of barcode systems is difficult because designs are financially sensitive to the companies that produce them; however, the patent system has as one of its principles that designs must be disclosed so patents are a good source of information.

Figure 9.20 shows a simple barcode system. Light is projected from a diode source through a collimating lens and bounced from a flat mirror onto the scanning mirror. The scanning is essentially a galvanometric scanner using a spring strip to give both suspension of the mirror and restoring force to centre the scan. The mirror is driven by a permanent magnet suspended in the centre of a magnet coil. This produces a much simpler system then a typical moving coil metre movement with hair springs for restoring force.

The received light traverses a similar optical path but uses a curved mirror to collect as much light as possible to be focussed on the detector. The arrangement of curved and flat mirror used in both the inward and outward path is much more efficient than the maximum 25% available from a semi-silvered mirror. The flat portion of the mirror probably only occupies 10% of the curved mirror giving approximately 100% efficiency for the outward beam and possibly 90% for the returning beam.

The combination of design ideas gives a simple compact and efficient design, however, this scanner only produces a linear scan across any object. *Figure 9.21* shows how a rotating prism scanner with faces which are not all at the same angle combined with two or more fixed steering

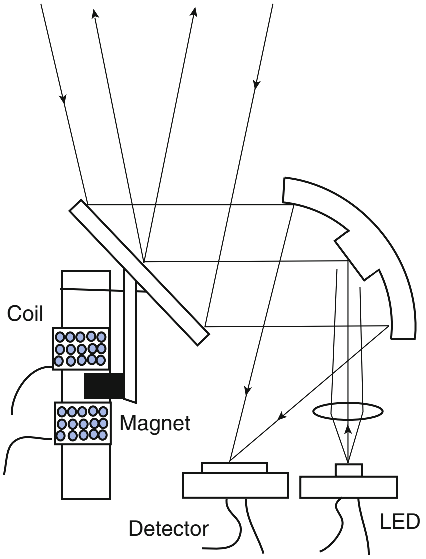


Fig. 9.20

Barcode scanning system from [2] showing clever combination of techniques to give a linear scan

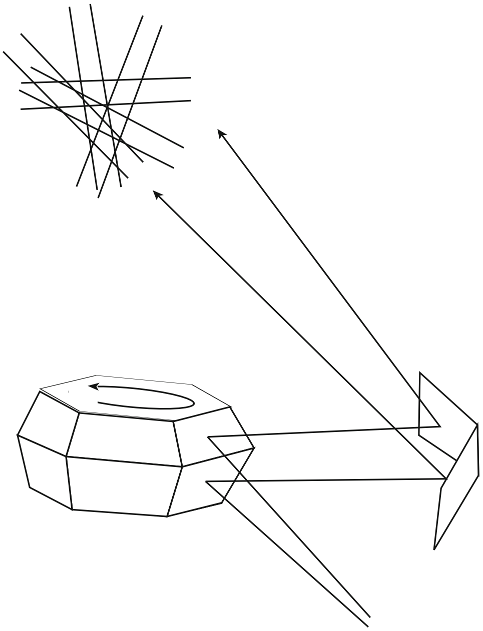


Fig. 9.21

Basic principle of producing multiple scan lines at different placements and angles

mirrors can produce a scan with multiple output beams separated in *x* and *y* directions ad at different angles. ➔ *Figure 9.13* shows the pattern produced by two rotating wedges rotated at different speeds, an alternative way of covering an area.

9.12 Conclusions

There are a very large range of scanner configurations covering an extreme range of uses. Scanners are usually used for specialised tasks which either cannot be accomplished with straightforward imaging or are more convenient for a specific use. Anything which requires extreme resolution in terms of total number of pixels produced, or large numbers of output channels or extreme wavelengths is a candidate for a scanner solution.

This chapter has concentrated on the configuration and optics of scanners; however, for any particular use, the data collection and image processing should be considered as an integral part of the design.

References

1. Dablestein et al (1987) Light scanning device for determining the condition of a surface with a light concentrator. US patent 4,693,601, 15 Sept 1987
2. Detwiler P (2000) Tilted offset barcode scanner. US Patent No. 6,045,045
3. Forrest AK (2007) Scanner with multi-angle response. J Opt A-Pure Appl Op 9:335–340
4. Graeme J (1995) Photodiode amplifiers: op amp solutions. McGraw-Hill, New York. ISBN 0-07-024247-X
5. Matsuda T, Abe F, Takahashi H (1978) Laser printer scanning system with a parabolic mirror. Appl Opt 17(6):878–884
6. Scheimpflug T (1904) Improved method and apparatus for the systematic alteration or distortion of plane pictures and images by means of lenses and mirrors for photography and for other purposes. GB Patent No. 1196, 16 Jan 1904, issued 12 May 1904
7. Weber K (1990) Optical scanning apparatus for detecting faults on a surface. US Patent 4,924,086, 8 May 1990

10 Cameras

David L. Gilblom
Alternative Vision, Tucson, AZ, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_10,

10.1 Image Acquisition Devices

10.1.1 Introduction

In machine vision systems, the job of the image acquisition device is to receive a pattern of illumination from an optical system and convert this varying photon signal into a varying electrical signal that is a true representation of its spatial and temporal variations. To build such a device requires combining an image sensor with electronics to operate it in a form that meets the optical, electrical, mechanical and environmental requirements of the system in which it must operate. This combination is called a camera. Because the operation and performance of a camera is often largely constrained by the image sensor in it, an understanding of sensor characteristics can speed identification of cameras suitable for particular tasks.

Like all transducers, image sensors have limitations – in signal handling capacity, in noise, in linearity, in stability and in time response. Uniquely, image sensors are subject to limitations in spatial resolution, in optical crosstalk, and in spatial, spectral and temporal uniformity.

Selecting a sensor is thus an exercise in understanding the characteristics of the optical signal available, defining the data to be extracted from the optical signal and selecting an image sensor that can bridge the gap between the two. Unfortunately, this exercise often results in zero candidate image sensors, prompting additional effort to make the required data more apparent in the optical signal, to raise the sophistication level of the post-acquisition data processing or to identify compromises that produce the best overall result. Just as often, candidate image sensors prove not to provide the expected data quality due to behaviour not initially recognized as relevant or due to discrepancies between the performance promised in the data sheet and the performance measured on the bench.

To further complicate selection, all the protracted and sincere efforts to normalize the presentation of image sensor performance specifications have still left wide discrepancies in test methods and presentation, forcing the imaging practitioner to understand what image sensors do and how they do it at a fundamental level.

10.2 Image Sensors

While an enormous number of physical mechanisms have been identified that can convert an incoming photon flux into an electrical signal, there are three groups of these that have been most often used in image sensors. The earliest sensors were based on the photoelectric effect, in which a material will emit electron into a vacuum when struck by photons with sufficient energy. In most practical sensors, these materials are transparent so that the photons can enter one side (generally the outside of a glass vacuum envelope) and the electrons exit the other. These materials are called photocathodes. Though still common in low-light-level instrumentation and x-ray imaging, they are rare in machine vision systems because the sensors using them tend to suffer from noise, distortion and signal non-linearities.

A second, mostly abandoned, group is photoconductors, in which the resistance of the material varies with exposure to light. Simple CdS cells used to control lighting are an example of this device type. Photoconductors were widely used in camera tubes, especially antimony trisulfide (Sb_2S_3) and various selenium compounds because they were relatively easy to produce in uniform films and operate at relatively low voltages. However, these materials suffer from signal non-linearities and from various forms of signal fatigue, which leads relatively quickly to burned-in images in static scene environments like those found in most machine vision tasks. In addition, all camera tubes are subject to geometric image distortions and to influences from external magnetic and electrical fields. Still, some photoconductors have survived in solid-state devices, especially in infrared applications.

Most current images sensor are made from photovoltaics, materials in which incoming light generates a voltage across a diode fabricated in the material. However, these materials are generally not run in the voltage-generating (or forward-biased) mode like solar cells, rather, they are reverse biased by an external voltage and generate a signal through the recombination of charge initiated by incoming photons. These materials are particularly well-suited to machine vision applications because they generate charge signals that are exactly proportional to the number of incoming photons and because they can be readily fabricated into geometrically stable monolithic planes. Even so, there are numerous sources of error in detectors made from these materials that the potential user needs to consider.

10.2.1 Basic Image Sensor Requirements

By far, the most commonly used image sensor material is silicon because silicon:

- responds strongly to visible light
- can be manufactured in sizes useful for image sensing,
- uses manufacturing processes and tools well-developed for other silicon devices,
- has performance characteristics compatible with the needs of high image data quality and
- can support fabrication of a wide selection of image sensor configurations at reasonable cost.

The process of building image sensors starts with the same base silicon material used in transistors, microprocessors and other silicon devices. These are wafers, typically 0.3 mm thick, with diameters from 100 to 300 mm cut from large cylindrical single crystals. The wafer is termed the *substrate* and is generally intrinsic silicon. That is, it is essentially pure silicon. With rare exceptions, discussed later, intrinsic silicon is not useful for making image sensors so impurities are introduced into the silicon crystal – a process called *doping* – to provide free electrons or holes that provide electrical conductivity. These impurities could be introduced into the wafer by thermal diffusion or by ion implantation but most commonly, they are incorporated into a new silicon layer grown on the wafer by a process called vapour phase epitaxy. This very uniform, doped *epitaxial* (literally "fixed on the surface") layer, perhaps 50 μm thick, will form the foundation for the fabrication of the image sensor.

To form an image sensor requires providing the silicon layer with a set of physical features that define the desired functions and performance. Fundamentally, any image sensor must take in incoming light, convert it to charge, keep the charge localized to the place where it was converted, hold the charge until the time for readout has arrived and then read out the signal generated by the charge. Each of these requirements adds complexity to the structure of the imager.

Any photodiode can perform the first two steps, converting incoming light to charge, but photodiodes do not preserve any information about the place of arrival of the light and do not hold the charge for readout, instead producing a continuous photocurrent. An array of photodiodes can provide spatial location information and a little additional circuitry can hold the signal and read it out on command. Thus an imager could be simply an array of photodiodes with some extra circuitry fabricated directly on the silicon wafer in a size appropriate to the imaging application. Some imagers are precisely this combination but many take advantage of the additional capabilities afforded by the ability of silicon to perform other useful functions.

10.2.2 Pixel Terminology

The terminology in use to describe pixels is very inconsistent in the technical literature, in product documentation and even in standards. In this chapter, terms have the meanings given here:

Pixel – Since its inception, this term has assumed so many different meanings that it often produces argument rather than communication. Entire articles have been written on this problem. In this book, the term *pixel* refers to any individually addressable detection or display element. This could be a photodiode in an image sensor or an LCD transmission location on a display. By extension, an *RGB pixel* refers to a triad of pixels associated with each other to detect or produce colour and a *multichannel pixel* refers to any set of pixels associated with each other to detect any sort of multispectral energy. The convention will be always to attach a modifier to the word "pixel" when referring to any collection of independently-addressable detection or display elements. A *superpixel* is a group of pixels combined spatially for some purpose. *Subpixel* refers to characteristics of pixels useful for consideration at dimensions smaller than the pixel itself.

Pixel Pitch – The vector distance between adjacent pixels. In the case where some pixels perform different functions from their nearest neighbours the pixel pitch and the pixel size of pixels of identical function are not equal. In image sensors or displays with non-rectangular pixel arrangements, both x and y (horizontal and vertical) dimensions may need to be specified.

Pixel Size – The entire spatial extent taken up by a pixel including its named functional purpose and all local auxiliary apparatus. Not necessarily equal to pixel pitch. Pixel size generally requires specification of two dimensions even in single-row image sensors.

10.2.3 Building Blocks

Image sensors are made up of arrays of small structures designed to perform the basic functions of converting light to charge, holding the charge and reading out the signal from the charge and to provide a variety of auxiliary functions to improve flexibility of operation or performance. Some of these structures operate at the pixel level while others may operate on rows or columns of pixels or on the image sensor more generally. The span of operation of particular structures helps define the functions a particular image sensor can perform.

10.2.3.1 Charge Generation and Storage

The most important element in an image sensor is a small volume of silicon in which the normal charge carriers present have been removed to allow storage of new charge carriers generated by incoming photons. Two methods are used to generate these storage volumes – charge storage wells, primarily in CCD image sensors and reverse-biased photodiodes, primarily in CMOS imagers. However, there are many examples of CCDs in which the photon detection is performed by reverse biased photodiodes.

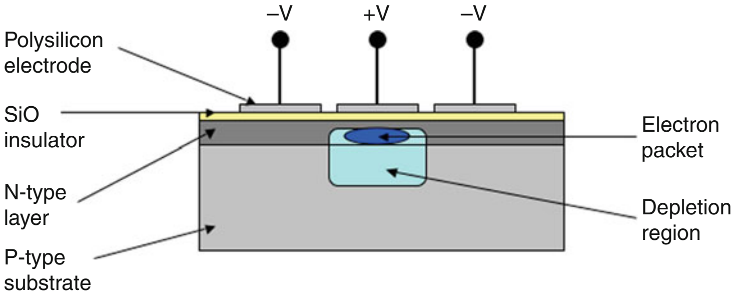
CCD Potential Well

In a CCD, the charge storage well is generated by the application of external electric fields to a silicon P-N junction formed by a P-doped substrate and an N-doped channel (➋ *Fig. 10.1*) using insulated electrodes on the silicon surface. These are structurally metal-oxide-semiconductor (MOS) devices. A positive voltage is applied to the top of the silicon through an insulating layer while the negative voltage is connected to the bottom of the silicon. The resulting field sweeps the charge away from the junction forming a depletion region called the *potential well.* Incoming photons generate hole-electron pairs. The holes travel to the negative connection and the electrons accumulate under the insulating plate but away from the silicon surface. This can continue to occur until enough electrons have been created to neutralize the effect of the positive voltage. Any further hole-electron pairs generated will simply recombine after a while. The number of electrons that can be stored is called the *full-well capacity.* Most CCDs have a full-well capacity of 20,000–100,000 electrons, depending primarily on the area of the well. The upper limit is around 700,000 electrons because charge packets larger than that are very difficult to transport without significant loss.

Note that the CCD structure can work without the top N-doped layer: in this case, the electrons simply accumulate under the insulator. However, allowing the signal charge to collect at the surface of the silicon reduces the charge transfer efficiency and increases dark current. As a result, essentially all modern CCDs use the P-doped *buried channel.*

Photodiode

The most common photodiodes are simply reverse-biased P-N junction diodes arranged so that light can strike them (➋ *Fig. 10.2*). The N-doped side of the diode is connected to a positive voltage and the P-doped side is connected to a negative voltage. Reverse-biasing produces a depletion region by extracting charge on both sides of the junction between the N and P type silicon. Arriving photons generate charge anywhere in the silicon. The charges separate and move to opposite sides of the junction causing current to flow. While the continuous production of current from arriving photons is useful in single-diode measurement applications, the diodes in image sensors must be equipped to share readout circuitry. In imaging sensors, the positive voltage is only connected for a short time to deplete the device and is then disconnected using a switch. This allows the photon-generated charge to slowly accumulate,



■ **Fig. 10.1**

CCD device, consisting of a P-doped substrate and an N-doped channel. The potential well is generated by applying an external electric field

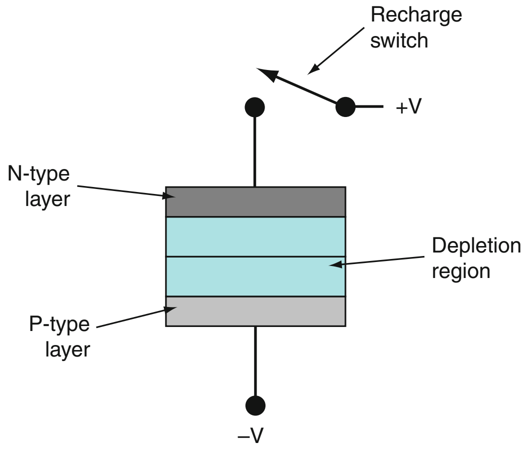


Fig. 10.2
Photodiode using a reverse-biased P-N junction

neutralizing the depletion region. The amount of charge required to completely neutralize the depletion layer is the photodiode's full well capacity. The signal from the photodiode can be read either by measuring the amount of charge that flows when the switch is closed or by measuring the voltage across the diode just before the switch is closed.

10.2.3.2 Signal Transport and Output

Signals generated by the photodetectors must be available for reading off the sensor chip to be of any use. There are two ways to do this – transport the charge packet itself to the output or measure the voltage generated by the packet and transmit this to the output. In CCDs, the charge packet is almost always carried to the output using *transport registers*, whether it was generated by the storage well or a separate photodiode. Since there is no good way to transport charge packets off an image sensor chip, it is still necessary to convert the packets to a voltage right at the last step. The conversion is made by a *sampling node*, which is a small capacitor that receives the charge at the end of the transport.

In most CMOS sensors, the charge to voltage conversion is done using a sampling node at each photodiode. The voltages measured there are simply copied thorough various gating structures to the output. This is the source of a major difference in the operation of CCD and CMOS image sensors. Because the CCD has only one sampling node (at the output) this same sampling node is used for every charge packet. This arrangement allows easy access to the sampling node for performing functions other than charge packet readout, including *correlated double sampling* (CDS), which will be discussed later. In contrast, CMOS image sensors have sampling nodes for every photodiode so these are not easily accessible for external use. Separate sampling nodes also raise the possibility that the sampling process can introduce non-uniformity in the image due to slight differences in the sampling node properties. This leads to one component of *fixed pattern noise* (FPN) in CMOS imagers.

Once the charge is converted to voltage, it can be amplified and buffered before leaving the chip. However, CCDs can support only one polarity of MOS transistor, either NMOS or PMOS, depending on the device structure. With only one polarity available, the only amplifier available is a source follower with a gain less than one. The typical CCD has a two-stage follower with a gain in the range of 0.8–0.9. CMOS image sensors can include complementary transistor pairs, which can provide gain of any desired value and essentially any other analogue or digital function possible with CMOS circuitry. This additional flexibility is another major difference between CCD and CMOS imagers.

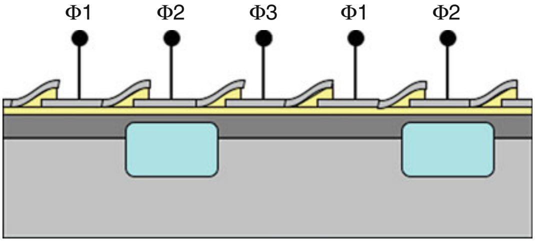
10.2.4 CCD Structures

Although the CCD image sensor is conceptually quite simple, consisting minimally of a set of vertical transfer registers, a horizontal transport register and a readout node, many varieties of these have been developed to meet different requirements. Knowing which variety was chosen for a particular image sensor can provide substantial information about performance characteristics and operating requirements of the sensor if the effects of choosing particular varieties are understood.

10.2.4.1 Transport Registers

The simplest CCD transport register (➔ *Fig. 10.3*) consists sets of three electrodes. A positive voltage, typically in the range of 5–15 V, is applied to electrode set Φ2 to create a series of potential wells. Over time, light arrives and generates electrons that are held in the wells. When the exposure is over, a positive voltage is applied to electrodes Φ3 to create potential wells there and then the voltage over the original Φ2 wells is slowly lowered to force the electrons to move into the Φ3 wells under the influence of the potential difference. The process is then repeated sequentially with so the charge can continue to be moved to the right. The slight electrode overlap assures that the potential wells will connect when the voltage is moved from one electrode to the next.

A register with three sets of electrodes is called a *three-phase* register. This is the simplest register because it has the fewest number of electrodes that can keep adjacent charge packets separated without additional structures in the silicon. In a three-phase register, there must be



■ Fig. 10.3
Simple CCD transport register. Charge packages are moved along the register by a three-phase clock applied to electrodes Φ1–Φ3

two electrodes at zero between charge packets. If there was only one, then when the voltages changed to move the packets, all of the electrodes would be positive at once and the charge packets could merge. It is possible to make a *two-phase* register with alternating on and off electrodes but this requires adding a diffusion at one edge of all of the electrodes at the silicon surface to provide a permanent voltage offset that can steer the charge packets in the right direction. The main problem with three-phase registers is that only one third of the silicon can be used to store charge. The other two thirds only provides space to apply the transport voltages between packets.

The proportion of the silicon used for charge storage can be increased to one half with a *four-phase* register. In this type, two electrodes are energized and the other two are at zero during charge accumulation and then the four electrodes are cycled to move the charge. The electrodes must be closer-spaced than in a three-phase register to maintain the same spacing between charge packets and four controllable voltages are needed rather than three, but the capacity to store charge is increased by 50%.

About half of the silicon space can also be used for charge storage in *virtual-phase* registers. These have only one voltage that is changed, relying on a complex set of diffusions in the silicon to provide fields to keep the charge moving in the right direction. This additional device complexity can add noise and increase dark current. Unlike the other register types, virtual-phase registers do not accommodate transporting charge in the reverse direction.

For all transport registers, *charge transport efficiency* (CTE) is a critical parameter. Each time the charge packet is moved from one cell to the next, some of the charge can be left behind. This smears out the signal as the packets of different sizes start to mix together. While a single shift losing, say, 0.1% of its charge might be acceptable, 1,000 shifts, typical of even small sensors, would deliver only $(0.999)^{1000}$ or 36.7% of the original signal to the output. An 0.01% loss per shift would deliver 90.5%, still generally unacceptable, and an 0.001% loss would deliver 99.0%. 0.001% loss corresponds to 0.99999 of the charge delivered per shift, the so-called "five nines" CTE. This is generally considered the minimum practical efficiency. Typical CCD registers now have between five nines and six nines CTE when operated properly.

10.2.4.2 Light-Sensitive Registers

When potential wells were originally linked to form transport registers for use in bucket-brigade analogue memory devices, the electrodes were made of aluminium just like any other silicon device conductors. After the discovery at Bell Laboratories that potential wells could act as sensitive, noiseless photon detectors, replacing the aluminium with something transparent to visible light became important. Currently, two materials are used. In the vast majority of CCD image sensors, the electrodes are polysilicon, selected for its easy fabrication and high conductivity. Unfortunately, polysilicon *is* silicon and absorbs light itself. This absorption has the biggest impact on the blue end of the spectrum because blue is more strongly absorbed than longer-wavelength light. As a result, early CCD image sensors had deficient blue response.

To improve blue response, sensor designers moved from straight-line electrodes to patterns that included areas free from polysilicon to allow blue light to reach the underlying detector. Some detectors now have electrodes made of indium tin oxide (ITO), which is a transparent conductor. ITO is more resistive than polysilicon so the speed and length of the electrodes is more limited but ITO does allow more blue light through. For very high blue response, *backside thinning* is used, which is discussed in detail below.

10.2.4.3 CCD Pixel Boundaries

In order to have a pixel, it is necessary to create a structure that converts photons to charge and then hold the charge at the place it was detected until the signal can be read out. Pixel boundaries are generally created in two ways. To produce isolated photodiodes all of the photodiode boundaries can be created during the semiconductor fabrication process by diffusing or implanting or depositing materials in the silicon that create electronic barriers to keep adjacent charge packets separated. These barriers are usually wrapped closely around the photodiodes so that charge that is generated in any photodiode remains there. To get signals in and out of the photodiodes requires only provision of aluminium conductors near the surface of the silicon because in a simple array of photodiodes the charge never moves.

In CCDs, however, the charge must move. The pixel boundaries in CCDs where the registers are photosensitive are, therefore, defined in one direction by the voltages applied to the electrodes. In the other direction, the pixel boundaries are formed in the silicon using silicon fabrication processes. Thus, in these CCD types, the boundaries between columns of pixels are permanent but the divisions along the column exist only when power is applied. Even in CCDs that use photodiodes for light detection one of the sides of the photodiode is usually defined by an applied voltage from an electrode not from a semiconductor process step.

10.2.4.4 Sampling Node

When a charge packet arrives at the point where it is to be sent off-chip, it must first be converted to a voltage. The simplest way to do this is to place the charge packet on a capacitor where the simple formula

$$V = \frac{Q}{C}$$

shows that the voltage generated (V) is proportional to the size of the charge packet (Q) for a fixed capacitance (C). If Q is set to be the charge of one electron (1.6×10^{-19} Coulombs) then the equation becomes

$$V = \frac{n_e Q_e}{C} \text{ or, rearranging, } \frac{V}{n_e} = \frac{Q_e}{C}$$

where n_e is the number of electrons in the charge packet. The value of the capacitor determines the *conversion factor* in volts per electron, which sets the output voltage range of the image sensor.

Typical CCDs have a conversion factor in the range of 5–10 microvolts per electron. Higher values than this are very difficult to achieve because the sampling node capacitor must be very small. The limit is often just stray capacitance from other nearby structures. The conversion factor may have to be intentionally set lower in image sensors with high full well capacity to avoid generating voltages larger than the output amplifier can handle.

10.2.5 Common CCD Architectures

Four types currently make up almost all CCD area sensors sold – *full frame* (FF), *frame transfer* (FT), *interline transfer* (IT) and *frame interline transfer* (FIT). Almost all of these are *frontside illuminated*, that is built so that the light enters on the same surface of the device that holds the circuitry, and they are almost all *single-tap*, having only one output point. There are some exceptions and a few additional variations on the structures that will be discussed in detail below. In all of the figures in this section, the light grey areas are sensitive to light and the dark grey areas are covered with aluminium. The arrows show the direction of charge motion when the electrodes are operating.

10.2.5.1 Full Frame Architecture

The Full Frame CCD image sensor (➔ *Fig. 10.4*) is the simplest type. It consists of a set of photosensitive registers arranged next to each other in columns with a transport register at the bottom configured so that each well can receive charge from a different column. In a typical operating cycle, light is prevented from reaching the sensor by a shutter and then the charge is cleared from the photosensitive registers while the image sensor is in the dark. The shutter is opened for the desired exposure interval and then closed. In the dark, the charge in all of the columns is shifted down by one row so that the last row moves into the horizontal transport register. A faster clock then moves the charge packets in the horizontal transport register to the sampling node to generate the output voltage. When the row is complete, the next row is moved down for readout. This process is repeated until all rows have been read. The sensor is the ready for another exposure.

FF Advantages

Of all front-illuminated CCD sensor types, the FF type has the largest percentage of the silicon area sensitive to light. The only truly insensitive parts are the diffusions in the silicon that define the boundaries between the columns. The vertical registers generally have three or four phase electrode sets and the horizontal register is often a special form of two phase electrode set that is designed to always move the charge packets in the correct direction. The waveforms used to drive FF sensors are simple – a set of phase drivers for vertical transport and a second set of drivers for horizontal transport, usually connected together and run from a single master clock. Since the vertical transport is always allowed more time that the interval required to read out

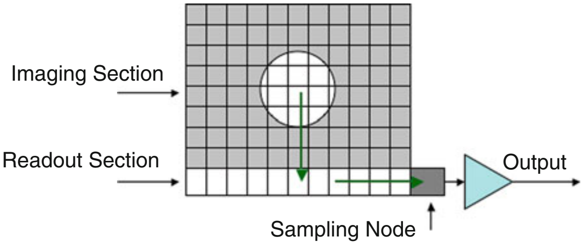


Fig. 10.4
Full-frame CCD image sensor

a row, the vertical electrodes can be designed to allow the maximum amount of light to pass and for maximum charge transfer efficiency even with large charge packets without regard for speed. This simple, slow structure has resulted in adoption of the FF architecture for design of most high-performance scientific devices.

Higher speed can be obtained in FF devices by providing more than one output from the horizontal transport register. The register is simply split into segments – 2, 4, 8 or more – so that these can be read out simultaneously. This reduces the line output time and, therefore, the time to read out a complete frame. In two-output devices, the horizontal registers may be arranged so that half of the image comes out on opposite corners or half comes out on one corner and half in the centre. The opposite-corner scheme is easier to fabricate but causes half to the image to be reversed.

Alternatively, the device can be split horizontally in the centre and read out through both top and bottom horizontal registers. This has the advantage of reducing the number of vertical shifts required but results in half of the image emerging upside down. In devices require very high speed, both vertical and horizontal segmentation may be applied. All segmentation imposes the requirement to provide separate readout electronics for each output.

FF Limitations

The defining requirement for implementing FF sensors is that they must be read out in the dark. Because the pixels remain light-sensitive while entire image is shifted vertically during readout, any illumination falling on the sensor will generate vertical stripes. More rapid vertical shifting can reduce the amount of this striping but cannot eliminate it entirely. Since each line must be completely read out before another vertical shift can occur, faster vertical transfer requires similar increases in horizontal readout rates.

The striping can be different above and below any bright points in the image because the stripe will be generated above the points for exposure during the current readout interval and below the points for exposure during the previous readout interval. This leads slanted stripes with breaks at the bright points when horizontal motion is present. If the pixels can all be reset at the end of each scan, then stripes will appear only above the bright points.

FF sensors can also suffer from vertical stripes that result from light overloads. The boundaries between pixels are defined only by barriers produced by applied voltages. Large exposures can generate enough charge to raise the pixel voltage above the barrier, allowing charge to spill into pixels above and below the overloaded area. In extreme cases, entire columns of pixels can be affected. To avoid this, some sensors are provided with *antiblooming* structures adjacent to each of the columns to drain off excess charge, thereby preventing the pixel voltage from rising above the barrier level. Because any light that falls on the antiblooming structures generates charge that is lost, the *fill factor* of sensors with these structures is reduced.

FF Applications

Full-frame sensors can be used wherever readout in the dark is practical. In most scientific imaging applications, the light can be easily controlled so FF sensors work well. Because FF imagers can be easily made with large pixels and low-noise readout amplifiers and because they are very compatible with cooling and backside illumination, they are very widely used in high-performance scientific cameras. Since scientific devices tend to need large numbers of large pixels and to support both high dynamic range and high resolution, the simple structure of FF sensors maximizes silicon utilization and minimizes fabrication complexity. FF sensors with active areas almost as large as an entire silicon wafer have been fabricated for scientific use.

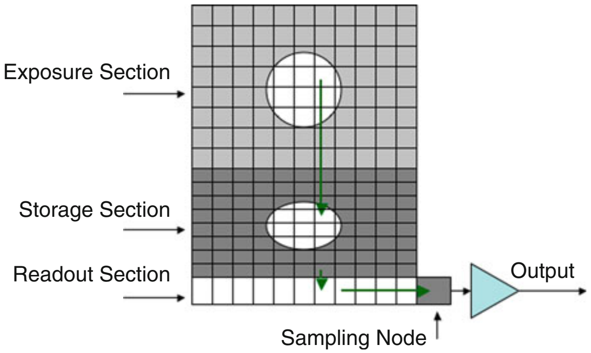
Almost all CCDs used in digital cameras for photography are the FF type because of the high utilization of silicon area and because this type of camera has a shutter to permit readout in the dark. However, FF sensors are not suitable for use in video cameras because the striping would be severe. Striping also prevents use of FF sensors in industrial applications except perhaps where the illumination is pulsed and the stray illumination is very low.

10.2.5.2 Frame Transfer Architecture

The Frame Transfer (FT) architecture (➋ *Fig. 10.5*) was developed as the first CCD type suitable for continuous (video) imaging. In an FT sensor, the exposure and readout functions of the sensor are physically separated with the exposure section built essentially identical the an FF sensor and a storage section almost a copy of the exposure section but covered with an opaque layer (usually aluminium) as shown in ➋ *Fig. 10.5*. In an FT sensor, the exposure and readout can occur almost simultaneously because while the exposure section collects light for a new image, the previous image, held in the storage section, is shifted out.

The FT sensor is slightly more complicated to operate that an FF sensor because the exposure and storage section need separate shift drivers. In a typical cycle for a simple 30 frame-per-second *progressive scan* video camera, the exposure section is collecting light and the charge is not moving while the previous information in the storage section is shifting down to the readout section line by line. After the shifting is complete, the two sections are connected to the same clocks and the entire image is quickly moved down from the exposure section to the storage section. The line shift rate during the transfer is typically several hundred times faster than the readout line shifting rate. Since the charge pattern moves down very rapidly, the vertical striping is reduced relative to the FF striping. After the shift, the scanning is disconnected from the exposure section and the readout resumes at a slower shift rate.

Generally, the CCD cells in the storage section are smaller than the pixels in the exposure section because there is no need to efficiently collect light, no need to provide a square matrix and no need to provide anti-blooming protection. A few extra rows are usually added at the top



■ Fig. 10.5
Frame Transfer (FT) architecture. Data relating to the previous image is shifted out while the imaging section collects light for the next one

of the storage section and covered with the opaque layer to provide a buffer between the edge of the light-collecting pixels and the storage cells. This prevents stray light from getting into the first few rows of the storage area that contain image data where it can generate spurious charge that can produce white background patches in the image.

FT Advantages

FT sensors can accommodate an additional type of antiblooming structure. Anti-blooming structures similar to those in FF sensor can be implemented for good results, however, the separation between exposure and storage sections also allows incorporation of antiblooming structures placed beneath the sensor array, so-called vertical antiblooming (VAB). Because the VAB structure is placed beneath the CCD structure rather than next to it the fill factor is not affected. However, VAB must be implemented carefully or it can introduce non-uniformity in the image and non-linearity in the sensor light response curve.

With a reverse charge dump, FT sensors can approximate simultaneous *electronic shuttering*, the electronic equivalent of a mechanical shutter. The reverse charge dump is a single long channel placed at the top of the exposure area into which charge can be transferred when the lines in the exposure section are shifted up instead of down. Charge moving into the dump is simply carried away and disposed of. If the Exposure array is shifted up rapidly, then any charge generated by illumination is quickly removed. Then, when an exposure is ready to be started, the shifting is stopped. During the reverse shifting, the time for accumulation of charge is longest in the row just about to be dumped so there is some non-uniformity in the background when the exposure starts. This prevents the shuttering from being perfect. Unfortunately, the line at the top of the exposure section that had the longest exposure during the dump process is also the last line to be moved under the opaque layer in the storage section. This effect can be minimized by designing sensors that can scan extremely rapidly in the reverse direction when, fortunately, preservation of image data quality of no importance. The reverse scan frequency in a sensor designed for this operation is typically around 5 MHz. For a sensor with 1,024 lines, this produces a total delay from the top to the bottom of the picture of about 200 μs.

FT Limitations

While the vertical striping is generally manageable, it can still be seen when very bright objects are in the image. With a 30 Hz frame rate and a 1 MHz vertical transfer frequency, a 1k × 1k sensor will have a ratio of frame exposure time to vertical transport time of about 30:1. With a bright spot 10% of the raster height that produces a signal just at saturation for the normal exposure, the vertical striping signal will be about 1/300 of full scale or slightly less than one LSB in an 8-bit system. Smaller or dimmer spots can be regarded as insignificant in most cases but larger or brighter spots will almost always produce visible and measurable artefacts. Vertical striping is worse in high frame rate FT sensors because the ratio of the frame readout time to the vertical transfer time is lower. Vertical striping can be eliminated by the use of a shutter that blocks the light during vertical transport. Synchronized rotating shutters were common when FT devices were used for many years in broadcast television cameras.

With the addition of the storage area, the FT sensors are less space-efficient than FF sensors. Backside illumination is more problematic because light must be prevented from striking the back of the storage section.

FT Applications

Frame transfer sensors are used where high fill factor is needed in situations where the light is on all the time. (Fill factor is the ratio of the area of regions capable of collecting light to the total area of the camera's "retina".) FT sensors offer some additional flexibility in camera design because the exposure and readout functions are disconnected in time. This is especially useful in long-exposure applications where reverse scanning can provide an approximately simultaneous start to an exposure and vertical transport can provide a similar termination. In such situations, the readout timing of the signal from the storage area can be optimized for data quality since it is independent of the exposure timing.

FT sensors are also useful in partial scanning situations. If only a subset of the lines contain useful data, then after vertical transport, the vertical shifting can continue at the high rate until the first useful line is reached, then slowed to the normal readout speed to capture the useful lines. After the last useful line is read, the remaining lines can be left in place to be transported out at the beginning of the next image. To support rapid vertical transport of lines without readout shifting, the readout section be designed to quickly dump the charge from the arriving lines. Failure to provide this capability will result in filling the wells and halting further transport of charge.

Because FT sensors generally have fill factors approaching 100%, they generate a minimum of aliasing. This makes them useful in viewing objects with highly structured surfaces where providing optical filtering is not practical. The aliasing is also comparable in the horizontal and vertical directions so object orientation is not critical.

FT sensors are relatively easy to bin. (Binning is a procedure in which several pixels are grouped into a function unit) Some sensors support vertical binning by providing access to the vertical transport phases in a way that allows double-height pixels to be defined and transported. Some sensors have even supported 1.5-height pixels. Alternatively, two rows can be transported into the readout register before they are shifted out. Many sensors include registers with storage capacity sufficient to allow this. Similarly, horizontal binning can be implemented by shifting two (or more) pixels into the readout node before sampling.

Finally, producing interlaced images with FT sensor only requires selecting phase lines that define the top and bottom of the pixels in alternate fields. Some sensors will actually only produce full vertical resolution in an interlaced mode. In these, the pixel areas overlap between fields so vertical resolution (and aliasing) are reduced.

10.2.5.3 Interline Transfer Architecture

In Interline Transfer sensors, the exposure and storage sections are alternated column by column in the same area of the silicon as shown in ➲ *Fig. 10.6*. Each column of photosensitive elements has next to it a vertical transfer register covered with aluminium. The group of transfer registers makes up the storage area. Light is collected in the photoelements, which may be CCD wells or photodiodes, and then the accumulated charge is moved from all photoelements simultaneously into the neighbouring transfer registers. After this move, the charge is shifted vertically, one line at a time, into the output register for readout. For faster operation, IT devices can incorporate multiple taps and can support split vertical scanning

Many variations on this basic arrangement exist, each with its own characteristics. Although the photoelements appear to be in columns as in FT sensors, the photoelements in IT sensors are not connected together vertically because there is no need to shift vertically in the

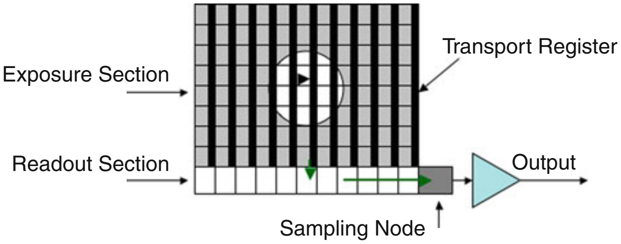


Fig. 10.6
Interline transfer sensor

photosensitive areas. Fundamentally, the photoelement can be either a CCD well or a photodiode. Typically, the CCD well has lower noise because it does not have to be reset but the photodiode has higher quantum efficiency because it does not require phase lines on its input surface. The large variety of configurations for photoelements now available has blurred the lines significantly so when the details of operation are important, thorough investigation of the actual configuration in candidate sensors may be required.

Transferring the charge from the elements to the transport registers may occur at a specified time or continuously until the time to scan has arrived. There may be some advantage to continuous transfer because the possibility of leaving some charge behind is reduced and because continuous transfer may allow the use of *pinned* photodiodes, which can have lower noise because they do not need to be reset [1].

IT Advantages

IT sensors also offer the possibility of electronic exposure control. If the photoelements are equipped so that all of them in the sensor can be simultaneously reset upon application of an external signal, then this signal can be used to define the beginning of an exposure. Since the end of the exposure is defined by end of the time that the charge is transferred to the transport register, the duration of the exposure is set by the difference between these two times. If the scanning is periodic, then the end of the transfer time will be periodic and fixed at a particular point in the exposure-readout cycle. This means that although the duration of the exposure can be varied, the position of the exposure in the cycle cannot. As a result, if the exposure must be synchronized with an external event, it is necessary to time the entire cycle to this event by operating in an *external trigger* mode in which entire images can be produced on demand.

Because the photoelements are all isolated vertically from one another, the possibility of blooming along the photoelement columns is essentially eliminated. Antiblooming drains can be added between each column of photoelements and the transfer register for the column to the left as part of the isolation structure required between them. When the photoelements are isolated on all sides like this, blooming can be kept under very tight control.

Closely-spaced exposure pairs may be taken with IT imagers. Assuming that the vertical registers are cleared, one exposure is taken in the normal way so that the charge ends up stored in the registers. As soon as the transfer to the registers is complete, another image may be taken. This is held in the photodiodes until the readout of the first image is complete. The second image can then be shifted to the registers and read out. With typical transfer times, the two images can be acquired only a few hundred microseconds apart. This process only works correctly with strobed illumination, otherwise the accumulation of light in the photodiodes will cause the second image to be exposed for a time equal to the difference between the length of the first exposure and the total readout time. The first exposure must occur just before the transfer for the images to be spaced closely in time. Of course, the minimum interval between acquisitions of pairs of these images is two complete readout periods.

In the CCD universe, IT sensors offer the most compact arrangement of functions; only CMOS imagers can offer more complexity in a similarly small space.

IT Limitations

Like FT sensors, IT sensors suffer from vertical striping but the cause is slightly different. Because there is no direct illumination of the vertical transport register where the charge is shifted, it would seem that there should be no striping. Unfortunately, the close proximity of the photoelements to the transport registers allows some light to sneak under the opaque aluminium layer. The amount is generally small, but the shifting proceeds slowly so there is ample time for the stray light to produce a signal. In a typical IT sensor, the vertical shifting takes exactly one frame to be completed. This means that the integration time for the stray light is essentially equal to the integration time for the photodiodes acquiring the desired image. Thus, even if the leakage is only 1%, the signal produced in the transport registers by an area just at saturation will be larger than one LSB in an eight-bit system. Oversaturated points can produce much larger stray signals. Confusing matters is the problem that the stray light is acquired from the frame that will be produced after the frame currently in the transport registers. Thus, fast-moving bright objects can cause vertical striping away from the positions where they appear in the real image. This problem is exacerbated by lenses with their last element very close to the sensor because the proportion of light rays arriving at the sensor at low angles – exactly the rays that get under the aluminium – is increased.

A substantial portion of the surface area of IT sensors is taken up with the vertical transport registers resulting in low fill factor and proportionally reduced net quantum efficiency [2]. In sensors with very small pixel geometry, the fill factor can be below 50%. The need to squeeze both the photoelements and the transport registers into a compact area also restricts the well capacity to 50% or less of the capacity of FT sensors with similar pixel spacing imposing a limit on dynamic range.

Because IT imagers typically incorporate junction photodiodes rather than charge wells for detection of incoming light, it is possible for some small amount of this charge to be left behind during the shift to the vertical transport register. This will show up as image retention or lag if observed frame-to-frame. Lag can be eliminated if the photodiodes are reset after each exposure or if pinned photodiodes are used.

IT Applications

IT sensors are very widely used in applications from machine vision and security to camcorders and medical systems due primarily to their small size and electronic shuttering capability. Their widespread use has led to the development of single-chip timing and scan generators to simplify camera design and minimize camera size and cost.

Many special-purpose IT sensors have been designed. IT sensors with photoelements of the UV-sensitive photodiode type are used to image masks and wafers with the same ultraviolet sources used in wafer production. With the non-photosensitive parts of the sensor protected by aluminium, these sensors are relatively immune to damage from UV photons.

Interesting hybrid sensors have been made in which the normal location of the photoelement is replaced by an isolated contact and a separate photosensitive layer is deposited over the entire sensor surface. This provides 100% fill factor and, with appropriate top layers, even infrared or ultraviolet sensitivity. Such devices are not common but the adaptability of the IT architecture continues to support many research efforts.

10.2.5.4 Frame Interline Transfer Architecture

The Frame Interline Transfer (FIT) architecture (➔ *Fig. 10.7*) was developed to provide both very low vertical striping and electronic exposure control. Striping in FT sensors, which have no exposure control, can be reduced only so far as the image can be moved rapidly from the imaging section to the storage section while IT sensors, which do have exposure control, have more striping as the exposure time is reduced. The FIT sensor is a combination of IT and FT in which the exposure section of an FT sensor is replaced by an IT sensor array. After the exposure, the charge can be shifted to the vertical transfer registers under the aluminium shields as in an IT device, but is then immediately shifted at high speed into an FT storage array below. As a result, the time available for accumulation of unwanted charge from light leaking under the shields is reduced to typically less than 1 ms with any exposure setting. The striping is thus reduced by both the effect of the shields and the rapid transfer out of the exposure section.

FIT Advantages

The FIT has an IT imaging section so the advantages of the IT sensor carry over such as low blooming, parallel reset and flexibility in designing the photodiodes.

The exposure in an FIT can end at any time, not just before the transfer to the storage section starts, because the image can be held under the shields until the FT storage section is ready for another image. However, as the image is held longer in the IT section, the time available for light to cause vertical striping increases. In the extreme case, with a short exposure at the beginning of the imaging cycle, the striping will be essentially identical to the striping of an IT device with the same exposure.

Closely-spaced exposure pairs with low striping may be taken with FIT imagers. Assuming that the storage section is cleared, one exposure is taken in the normal way so that the charge ends up stored in the storage section. As soon as the shift to storage is complete, another image

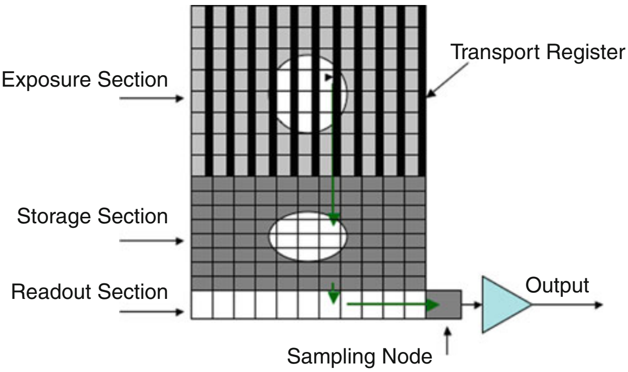


Fig. 10.7
Frame Interline Transfer (FIT) sensor

may be taken. This is the stored under the shields and held until the readout of the storage section is complete. The second image can then be shifted to storage and read out. With typical vertical transfer times, the two images can be acquired only a few hundred microseconds apart. If the illumination is strobed, the image can be closer. Of course, the minimum interval between acquisitions of pairs of these images is two complete readout periods.

FIT imagers can also take exposure triplets using pulsed illumination. When the exposure pair is complete a third exposure can be taken and held in the photodiodes until the first two images are read out. Images two and three can be more closely spaced than images one and two because the wait for the vertical shift does not apply. Readout of complete triplets takes three readout periods.

FIT Limitations

Because the FIT sensor has both FT and IT sections, it is at least as large as the equivalent FT sensor and 1.5–2 times as large as the equivalent IT sensor. In addition, it has the same fill factor restrictions as an IT sensor. FIT sensors cannot be charge binned in the imaging section like FT sensors because the imaging section is of the IT type. However, binning by holding the charge for more than one vertical shift into the output register remains possible.

FIT Applications

FIT imagers are used almost exclusively in professional television, where a rotating shutter is added to block the light during the brief transfer time so the striping is truly reduced to zero. The last bit of improved resistance to vertical striping provided by FIT sensors is generally not worth the extra cost in applications where the lighting can be controlled. The major exception is in microscope imaging using EMCCD sensors (see ➋ section "CCDs with Gain" below), where very bright objects must sometimes be tolerated in images which are otherwise very faint. Any application requiring strong stray light rejection might benefit from the use of FIT imagers.

10.2.6 Special CCD Architectures

The four main CCD architectures share many common features arranged in different ways for different advantages. Typically, these result in rectangular arrays of pixels with image data coming out in one corner. However, by modifying these simple arrangements and applying a few additional electronic features, the performance of CCD sensor can be expanded in several ways.

10.2.6.1 Multiple Outputs

Often, the speed of the sensor is limited by the maximum allowable clock rate of the horizontal transport register, typically around 30 MHz but often lower and occasionally higher. To get around this limitation, multiple transport registers may be used in conjunction with multiple output amplifiers. Several configurations have been implemented:

- Left-Right (2× speed increase) – The sensor is divided vertically so that the left half of the area is read out to the left and the right half is read out to the right through a transport register that lies along the bottom of the sensor. This is relatively simple to design but results in half of the pixel data coming out backwards with respect to the other half. The order of the lines is unaffected. This configuration can easily be applied to any of the architectures.

- Top-Bottom (2×) – The sensor is split into top and bottom halves, each with its own full-length transport register. The top and bottom line order is reversed but the pixel order can be the same for both halves. This configuration is easy to apply to FF imagers more difficult with IT imagers and much more difficult with FT and FIT imagers because the storage area must be split, half on the top and half on the bottom.
- Quadrant (4×) – The sensor is split both horizontally and vertically with readouts at all four corners. The data order comes out in all four possible combinations. Design is similar in complexity to the Top-Bottom configuration.
- Multi-Tap (nx) – The sensor is split vertically into several (n) stripes and each stripe has its own output. This scheme is only rarely used because designing transport registers that nearly touch on the ends, provide paths for the signals in the centre stripes to get out and provide high CTE is quite difficult. Generally, this configuration can only be applied if the pixel spacing is relatively large. The line order is undisturbed and all pixel data comes out in the same order.
- Sub-Sampled (4×) – In sensors with 2 × 2 colour filter arrays, the charge may be routed to separate outputs for each of the four array quadrants. This requires some complexity in the transport area because the four signals must somehow be separated using only charge transfer. This configuration is more likely to be found in CMOS sensors where implementation is easier.
- Column Readout (c×) – Each column (c) is fitted with its own readout amplifier. No transport register at all is used at the output. Due to practical limitation on the number of pins available on CCD packages and the power dissipated by large numbers of amplifiers, this configuration can be applied only to relatively small sensors.

Top-bottom splitting could also be applied to any of the last three configurations.

All tapped sensors share the problem of multiple output amplifiers that are not quite alike. Typically, the camera must include compensation for mismatch of both gain and offset between output channels to avoid striping in the final image.

10.2.6.2 CCDs with Gain

Ionization is an atomic process in which an incoming particle provides enough energy to an orbiting electron to free it. In semiconductor devices, freeing an electron means promoting it from the valence band to the conduction band. The energy to do this can be provided by an electron that has been accelerated through a large enough voltage so that it can strike a valence electron and promote it to the conduction band. The minimum energy for ionization in silicon is 1.18 eV. This sets the minimum electric field required to add that much energy to a conduction electron before it collides with another electron. The required field can be provided in a typical CCD transport register by raising the phase voltages used to move the charge to about 40 V.

Because there are other ways for electrons to lose the imparted energy, only about 1% of the electrons that move through a transport register stage ionize additional electrons but those that are ionized join the charge packet being shifted. This means that if 1,000 electrons enter the stage, about 1,010 electrons leave it. By using multiple stages as in a typical transport register, the number of electrons can be substantially multiplied. One electron will produce 1.01 electrons on average because ionization is a statistical process.

With 1% gain per stage, a gain of 2 requires 70 stages so a gain of 100 requires 463 stages and a gain of 1,000 requires about 700 stages. These are well within the constraints of designing and constructing transport registers. Fortunately, the multiplication process in silicon is nearly noise-free so adding a gain of 1,000 through impact ionization is feasible.

The result of adding the multiplication structure is that the sensitivity of a CCD can be increased by 1,000 with essentially no added noise with only the cost of adding a single 40-volt transport register. With this structure, a CCD can perform in low light levels as well as a sensor coupled to a traditional image intensifier while providing certain advantages.

- Since the intensifier can be eliminated, so can the fibre optic coupling, the high voltage, the image degradation and the attendant size, weight, distortion, noise and magnetic susceptibility.
- The sensor can be run with the multiplication function turned off. In this mode, it will produce the same image quality as it would if the register were not present at all.
- Because the quantum efficiency of silicon is higher than the quantum efficiency of photocathodes, the noise statistics of the incoming image photons are better.
- Silicon responds over a wider spectral region than photocathodes, including the near infrared.
- All of the normal surface treatments, such as colour filter arrays and microlenses can be applied to CCDs with multiplication registers.
- Since charge packets from all of the pixels go through the same register, there is no fixed pattern noise added in multiplication.

Of course, there is some added complexity to the design of the register. Consider, for instance, what happens if the charge packet entering the register is almost large enough to fill it. Some provision must be made to allow the size of this packet to be kept small enough to exceed the charge transport capacity. This is not a problem in normal CCDs. Special design is required to avoid high dark current at the high applied voltages. Many similar problems had to be addressed to make these sensors practical. These have been solved so the sensors are now readily available although they remain comparatively expensive.

The two most common applications for these sensors are night vision and fluorescence imaging in microscopes. They are useful wherever very low light levels are encountered.

10.2.6.3 Backside Illumination

CCDs of all architectures suffer in sensitivity because substantial parts of their areas are either taken up with non-photosensitive structures like transport registers or covered over with light-absorbing materials like polysilicon. To get around this problem, it is possible to illuminate the sensor from the back side, which has no structures. If a standard CCD is flipped over and backside illuminated only the near-infrared can be detected and the images will be fuzzy because most of the light is absorbed by the thick silicon and electron-hole pairs are generated far from the light-sensitive registers. The long distances allow most of the charge to recombine and much of the rest to spread laterally. Overcoming this requires thinning the silicon until it is thick enough to hold together but thin enough to permits capture of the charge in the right locations. This is a tedious process still only performed by a few facilities, mostly for high-performance astronomy applications. Carefully done, backside thinning can produce sensors with 90% peak quantum efficiency and spectral response from the near ultraviolet through the visible.

Aside from being very expensive, backside thinning has other limitations. The most important is that in backside illuminated sensors there is no way to block light from reaching all of the sensor structures. In IT sensors for instance, light must be prevented from reaching the vertical transport registers. Since this is not possible, IT sensors are not suitable for backside imaging. Even if the charge could be cleared from the registers, it would just be thrown away, essentially lowering the fill factor. Some other sensors do not work properly with backside illumination. Sensors with vertical antiblooming will have very low sensitivity because most of the charge will be generated and dumped in the anti-blooming layer. Essentially all backside illuminated devices are FF types without antiblooming. They are intended to be used in applications like astronomy where every photon counts. It is fortunate that the frame rates in astronomy are low because thinning devices increases the substrate resistance to the point where they cannot be scanned very fast.

10.2.6.4 Cooling

The dark current of CCD sensors may be reduced by cooling. The inverse of this is that the dark current of CCD sensors is raised by heating. Typically, the dark current will double for every 8°C rise in temperature. Sensor data sheets generally show the dark current at 25°C (although some show slightly lower temperatures – watch for this) but the temperatures inside cameras are generally warmer than this. The typical rise is about 20°C so the dark current even with a camera in room temperature ambient is already likely 45°C. That is the dark current is already nearly six times higher than the data sheet says. As an example, let us assume that the data sheet dark current is 50 pA/cm^2. In a 1k × 1k 10 μm pitch sensor, this is equivalent to accumulation of about 10 electrons/frame/pixel at 30 fps. In a real camera we multiply this by 6 for the rise from ambient and another 13 to account for the maximum 55°C maximum operating temperature. This results in a dark signal of 780 electrons. While this may not seem like much compared to well capacity, it carries with it a standard deviation of 28 electrons, which is likely larger than the sensor noise. As a result, the dynamic range of the camera may be limited by the noise in its dark current. In addition, non-uniformity in the dark current may mask subtle brightness variations in the image and “hot” pixels (those with unusually high dark current) could become a problem at these temperatures. Cooling, or just control of the temperature, can help these problems.

Note that cooling has little direct effect on the sensor noise. The sensor noise is roughly proportional to the square root of the absolute temperature. A 30°C change in temperature is also a change of 30 Kelvins, perhaps from 273 to 303 K. The increase in the sensor noise in this circumstance is only 5% while the dark current noise increases by 780% and becomes dominant.

10.2.7 CCDs for Line Scanning

Most imaging involves viewing of individual objects or of scenes. For these tasks, area imagers are suitable. However, when the object to be viewed is larger than an area camera can view in one shot at the necessary resolution or when the object to be viewed is in continuous linear motion, line scanning may be more appropriate. Line scanners sample the viewed object in only one dimension, relying on motion of the object to provide samples in the orthogonal direction. Traditionally, the direction along the sensor is "horizontal" and the orthogonal direction is "vertical", probably because of the usual direction of scan lines on displays. Oddly, in many actual inspection setups, it seems more natural to have the object move from left to right on the display because this mimics standing at the side of the object as it moves rather than on a catwalk above it. For this reason, "horizontal" and "vertical" are often replaced by "cross" and "down" respectively in referring to line scan directions.

While linescan sensors are simpler than area image sensors in some ways, certain features used only in linescan sensors merit special consideration.

10.2.7.1 One-Line Sensors

CCD linescan sensors are essentially one line of an FT image sensor consisting of a single row of wells isolated from each other by diffusion and connected by a charge transfer register from a readout register. The simplest current linescan sensors are built exactly this way. In operation, the wells are isolated from the readout register by a voltage (actually a negative voltage if the charge is electrons) to a transfer gate between them to trap the charge in the well during an exposure. When the exposure is over the barrier voltage is removed (and, perhaps, the well voltage is raised) to allow the stored charge to flow into the readout register. After the transfer, the barrier is raised again so the next exposure can start while the charge in the transport register is read out, just as in any CCD image sensor.

Variations abound. Because there is only one row there is no need for vertical charge transport, except to the readout register. As a result, the exposed area can actually be a true photodiode instead of a potential well without any fill factor penalty. This permits the photosensitive area to be tailored to the requirements for different applications. For instance,

- In spectroscopic sensors, the photodiodes can be long and thin to match a spectrometer slit.
- In ultraviolet applications like mask inspection, the diode can be optimized for UV response and protected from UV damage
- When high linearity or very low lag are required, the photodiode can be pinned
- Diodes that collect holes rather than electrons can be made so that the signal polarity can be reversed

The readout registers can be modified as well. As in the area imagers, the registers can be split left-right, top-bottom and into segments. In addition, the registers can be arranged so that one of the top of the sensor reads even pixels and one on the bottom reads odd pixels. This has the advantage of relaxing the design rules on the readout registers to make very high speed readout operation simpler while retaining high charge transfer efficiency. As with areas sensors the gain and offset of the various readout amplifiers on linescan sensors may need compensation. This is especially critical in the even-odd readout schemes because offset errors produce every-other-pixel patterns in images that can be hard to eliminate.

For very high line rates in short bursts, linescan sensors can be provided with a storage area essentially identical to those in area imagers. These permit the transfer of many lines of charge from the photodiodes at speeds much faster than the readout register can handle. The limit, of course, is the number of storage lines provided because at some point, the acquisition process has to be stopped to allow the storage area to be emptied. With some careful masking and adjustments in scan timing, the last line of many area imagers can be used in this way, too.

Access to both sides of the photodiodes allows other functions to be applied that are more difficult, if not impossible in area sensors. For instance, very robust, fully-adjustable anti-blooming can be provided because the anti-blooming voltage can be completely separated from the readout functions. Additionally, adding anti-blooming structures does not reduce fill factor because they can be positioned outside the photosensitive area. Structures to insert known charge packets can also be provided either at the end of the line or along the whole row to provide calibration data for CTE and non-uniformity.

One problem can arise in the design of linescan sensors that is rare in area sensors. Because the height of the photodiodes is essentially unconstrained, there can be a temptation to make them as large as possible and then compensate optically to improve the sensor dynamic range. Up to a point, this works, but two problems can occur if the photodiodes are too long. The first is excess lag. Since all of the diode charge has to move to the readout register while the transfer voltage is lowered, the charge farthest from the transfer gate may not have time to move all the way. This leftover charge will make of part of the next image. The second problem is the limited capacity of readout registers. Even with all of the design tricks applied, the maximum capacity of CCD transport registers is about 750,000 electrons. Beyond that, increasing amounts of charge will be left behind at each transfer, leading to poor CTE. Building diodes that can accumulate more charge than this is easy so any proposed sensor design including large diodes needs to be checked for potential excessive charge packet problems. This is why spectroscopic sensors that require very large charge storage capacity do not use transport registers for readout.

10.2.7.2 Time Delay and Integration

One limit to the usefulness of linescan sensors is their very short exposure times. Even though the exposure time can approach 100% of the total cycle time for acquiring and reading a line, the high line rates still lead to short exposures. With a 40 kHz line rate, common is sensors now, the maximum exposure time is only 25 µs. This leads to the need for very intense lighting – so bright that in some high-speed systems, the motion of the material cannot be stopped while the illumination is on due to the danger of starting a fire. As the need for speed increases, another technique can be applied, subject to certain very important restrictions, to increase the exposure time. This is time delay and integration – TDI.

If the image of a linearly moving object like a conveyor belt is viewed by an FT CCD image sensor, then the result is an image in which all of the points on the conveyer that pass a particular point on the sensor are added together. In other words, the image is smeared vertically. However, if the motion of the conveyer is aligned accurately with the vertical axis of the sensor, then the image will not be smeared horizontally. If the conveyor had a series of stripes aligned in its long direction, then an image of the moving conveyor would look just like an image of the conveyer if it were stopped. This is because the stripes are always aligned with the pixels on the sensor.

If the stripes are across the conveyor, then they will be smeared because they traverse the pixels. In theory, the camera could be moved with the conveyer to keep each point on the conveyer aligned with a particular row of pixels. That would be mechanically inconvenient and it would cut the conveyor image into a series of stills rather than generating one continuous image like a linescan camera.

Fortunately, in CCD imagers the normal scanning process requires that the charge be moved in a vertical direction during readout. Normally, this is done as fast as possible to avoid smearing the usual static scene. If the charge motion is slowed to match the speed of the image of the conveyor, then each point on the conveyer stays aligned with the charge it creates on the sensor as this charge is moved row by row down the sensor. To maintain the alignment, the charge can be moved instead of the camera. In this mode, the charge has the entire time the point traverses the sensor to accumulate. The exposure time is thus lengthened. The process of moving the charge is called time delay and integration because there is a delay between the time an object enters the field of view of the camera and the time it leaves that is exactly equal to the resulting integration time. With a sensor having 100 lines, the exposure time with a 40 kHz line rate would be 2.5 ms, providing substantial relief in the lighting brightness requirements.

It is worth noting that TDI was first applied in surveillance aircraft by pulling wide film across a slit parallel with the aircraft motion. Silicon TDI sensors were first developed as an electronic replacement for the film.

Requirements for TDI

TDI sounds simple enough but there are a number of important constraints that must be recognized if high-quality images are to be produced, mostly relating to keeping the object and the charge aligned.

- Use high-quality optics – Distortion in the optics will cause the image to wander as it traverses the field of view and wandering means blurring. Non-uniform MTF will cause non-uniformities in blurring from centre to edge.
- Assure accurate optical alignment – The image sensor must be parallel to the object and the lens optical axis must be perpendicular to both of these. Deviations result in variations in magnification across the field of view.
- Keep travel parallel – The rows on the sensor must be at right angles to the travel of the object. The rapid onset of smearing when the travel is misaligned can be used as a tool to align the camera accurately. Keep in mind that lenses invert the image when deciding which way to mount the camera.
- Hold a constant distance – The distance from the object to the camera must remain constant. Changes in distance imply changes in magnification so if the distance changes and the conveyor speed is constant, the charge motion will no longer match the conveyer image motion.
- Maintain synchronization – Assuming perfect alignment, correct synchronization between the object and the charge must be maintained. The details of various methods to achieve this are discussed below.
- Avoid vibration – Vibration of any sort will smear the image. There is no equivalent to strobing with TDI so the camera must remain very stable relative to the object at all times.

There are some variations that do not affect the image as they would with either area or single-line scanning. For instance, the illumination in a TDI system does not have to be uniform over the entire field of view. The signal from the camera is essentially an average of the illumination levels and so the output signal is always the same because each sum of lines sees the same total illumination. In addition, the extreme alignment requirements for light sources with line scan cameras are substantially eased. Whereas in line scan installations only a few percent of the light is typically in the right place for imaging, TDI illumination is area illumination and can be well over 50% utilized. However, uniformity across the field of view is still required because the averaging works only in the direction of travel.

Even stray objects between the camera and the object have a reduced effect as long as their motion is not synchronized with TDI scan. The switch from film to electronic imaging in the surveillance aircraft was driven by this fact. Non-uniform objects in the image path (clouds for instance) produce a non-removable imprint on static images, but in TDI images the effect of the clouds is smeared out and the contrast of the final image is improved. This works on film as well as in electronic imagers, but removing the bright offset from the clouds in the images is much easier in the electronic domain. The improvement in signal-to-noise ratio is proportional to the square root of the number of lines in the TDI sensor.

Synchronization Methods

Somehow, the motion of the image generated at the focal plane and the motion of the charge on the sensor must be kept moving together. Several methods have been used.

- Free running – In the surveillance example, the distances are long, the earth is stationary, the platform is stabilized and the aircraft speed can be well controlled. In this case, the speed of the image can be calculated and provided to the sensor clock circuits. Over practical time frames, the scanning will stay synchronized.
- Known speed – In some applications like glass sheet inspection, the object is stationary and the camera is moved. Since the camera speed is under the operator's control, the charge transfer speed can be made to match it.
- Drive the object – In processes where the conveyer or other moving object is controlled by an external clock, the clock can come from the camera to make the object move at a speed determined by the charge motion.

In all of these methods, the speed is constant because it is set by a controlled clock. This means that the integration time for every line is identical and, therefore, the signal generated by each line for a particular object brightness will always be the same. Unfortunately, many real-world applications do not permit external clock control as in the fourth synchronization example.

- Trigger the camera – Real moving objects like paper, plastics textiles, and almost everything else can't be clocked. Instead, they have rollers with rate encoders that can tell a scanning system how fast they are moving. This information can be converted to a train of pulses that are used to tell the charge when to shift.

Because the triggering rate can vary with time in this method, the integration time can vary from line-to-line also. This may require that the image processor keep track of the actual integration time for each line so the signal amplitudes of all lines can be normalized.

Limitations of TDI

While TDI can solve the sensitivity problem in many linescan applications, its tight geometric requirements limit the span of usefulness. Some specific considerations include:

- Parallax – The parallax phenomenon is familiar to anyone using line scan cameras. Parallax simply refers to the variation in viewing angle of the object from the centre of the image to the edge. Parallax will make objects at the edge of the image look different from those at the centre and causing, for instance, variations in the measured positions of edges. TDI imaging has parallax from the top to the bottom of the scanned field as well, but the reported result will be the average of the signals generated over the entire range of viewing angles. In objects that are not perfectly flat, this average will include data from all of the views in the scanned area.
- Speed variation – As noted above, variation in the speed of motion causes variation in the integration time line-by-line. In addition, each TDI imager will have upper and lower limits to scan speeds that can be tolerated. The upper limit is the same as for any line scan imager – it is simply the time required to read out a line. The lower limit may be set by accumulation of dark current or by saturation of the imager as the integration times lengthen.
- Flatness – TDI works best only on truly planar objects. Height variations cause both defocusing and loss of synchronization with charge motion. Telecentric optics can alleviate these problems if the coverage area is small enough.

Practical TDI Sensors

Although nearly any FF or FT CCD imager can be operated in a TDI mode, purpose-built TDI imagers have certain architectural features that optimize TDI performance. FT imagers have the disadvantage that they add a delay between the end of the exposure and the time that the line will be read out. This can be minimized by clocking each arriving line rapidly through the storage area but only if the process is fast enough to finish the traverse before the next line is ready. An advantage of retaining the storage area is that the TDI imager can be used to make images when the motion is stopped. This can facilitate system setup, alignment and troubleshooting.

Vertical transport in TDI imagers generally utilizes a four-phase structure so that each shift of a line is broken into four parts. While this reduces the charge storage capacity of the wells, it produces smoother motion than two or three phase structures and keeps the moving charge better aligned with the moving image.

While the integration time in TDI sensor could be increased without limit by adding lines, the practicalities of maintaining the geometric relationships and the diminishing returns of adding more line generally cross over in the neighbourhood of 100 lines. As a result, most TDI imagers have line counts in that range. Line counts of 96 or 128 are common. TDI imagers can be made faster by adding taps, but this can only be done on one side because the charge has to travel in a specific direction. Taps added to the other side can allow bidirectional operation. To provide limited exposure control, some TDI sensors have controls that can disconnect groups of lines from the scan signals. Thus, a 96-line sensor might also be capable of operation with 64, 48 or 32 active lines. The sensitivity adjustment is not smooth but may solve problems where the speed or lighting vary substantially. The active lines are those nearest the output so the centre of the image shifts as the number of active lines is changed.

10.2.8 CMOS Structures

CCDs are limited in the complexity of circuitry that can be included on-chip because they are essentially either NMOS or PMOS devices. That, is, they can support only one polarity of MOS transistor. While this structure is sufficient for gating and signal following functions, it cannot support fast logic gates or voltage amplification. For these, both transistor polarities are required. The circuit structure that has them both is called CMOS. Of course, CMOS processes have been around for a long time so it might be expected that high-performance imagers using them might also be that old. They reason that they have only recently emerged as competitors to CCDs in imaging lies in the restrictions of CMOS process.

One disadvantage of CCDs is that they must be made using special semiconductor fabrication processes tailored to producing the particular structures needed in CCDs for good performance. With CMOS, the early expectation was that CMOS processes already in use for digital logic could be directly applied to fabrication of imagers. Unfortunately, this was not the case. CMOS sensors require changes in the processes to support fabrication of high-quality photodiodes, a device not present in CMOS logic. In addition, geometric factors come into play.

10.2.8.1 Optical Structures

In CCD's the only large structures in the photosensitive area of the devices are the light conversion areas and, in interline devices, the transport registers. Some area might be taken up by antiblooming features but the majority of the area can be devoted to light sensing. In CMOS sensors, however, each pixel must be supplied with control and readout circuits consisting of multiple transistors. These take up significant space at the surface of the silicon, in the same plane as the photodiode. As a result, the fill factor in CMOS devices can be quite low, in the range of 30% in devices with small pixel pitch. Further, the metal interconnect lines must be built above the silicon surface with the result that each photodiode usually sits at the bottom of a hole. ➲ *Figure 10.8* shows the typical situation. As the pixel pitch is made smaller, a larger portion of the available area is taken up by the circuits and so the width of the hole relative to its depth gets smaller and smaller.

Microlenses

To provide enough light in small-pitch devices, a plastic layer is added to the surface with the upper side formed to make a small lens above each photodiode. Such structures are called microlens arrays. ➲ *Figure 10.9* illustrates these. (Also see ➲ Chap. 6) As the two sample rays show, much of the light that would have missed the photodiode will be directed to it by the microlens, increasing the effective collection efficiency by as much as a factor of 3. Microlenses

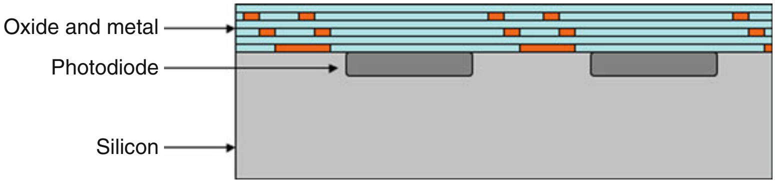


Fig. 10.8
CMOS sensor. Each pixel is supplied with control and readout circuits consisting of multiple transistors. These reduce the fill factor. Metal interconnect lines are placed above the silicon surface. Hence, the photodiodes are located below these

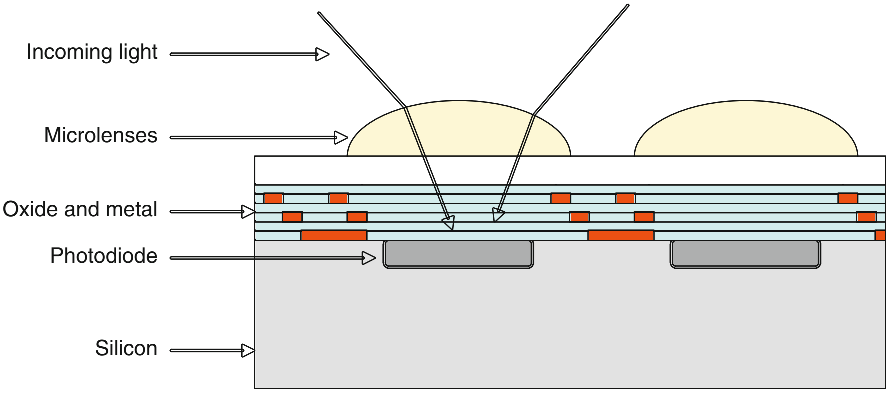


Fig. 10.9
Microlens dedicated to each individual protodiode. The microlens focusses light onto the photosensors, which improves the fill factor and increase the angle of acceptance for incoming photons

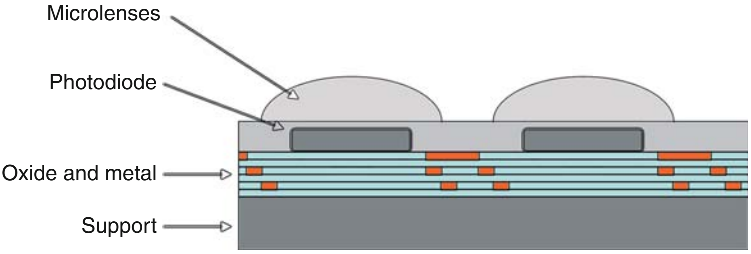


Fig. 10.10
Support wafer bonded to a thinned sensor array with microlenses

are most effective on pixels in the 2–5 μm pitch range. Larger pixels generally have sufficient collection efficiency and smaller ones have acceptance holes that become too deep and narrow to be helped much.

Backside Thinning

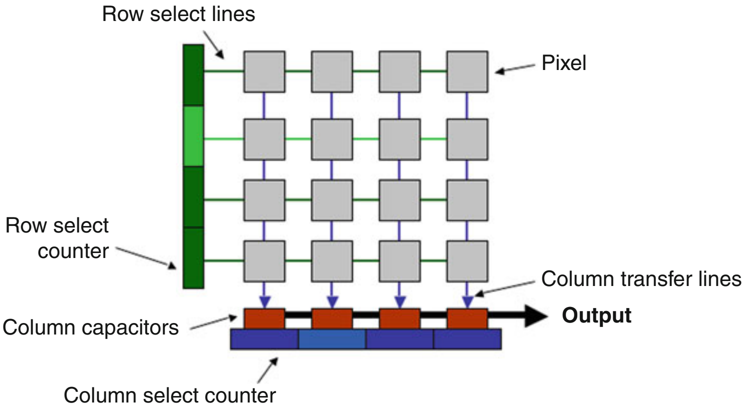
One solution to photon collection in very small pixels is backside thinning. Backside thinning of CMOS imagers is more difficult than for CCDs because of the large amount of structure in CMOS circuits. However, the improvements in quantum efficiency are similar. There are several methods to provide backside access for light to the photodiodes but all of them are similar in that they require the finished array to be thinned. A support wafer must be somehow bonded to the sensor array because after thinning the array silicon is typically less than 10 μm thick. Backside illumination eliminates the need for the light to enter a hole, but it does not improve the fill factor because the photodiodes and the circuitry remain in the same plane as shown in ➜ *Fig. 10.10*. Microlenses can still improve the collection efficiency.

10.2.8.2 Basic CMOS Operation

The baseline for all CMOS designs is the architecture shown in ➋ *Fig. 10.11*. The CMOS sensor is an array of photodiodes each supplied with sampling and switching circuits. In the simplest sensors, the entire array is reset by reverse biasing all of the photodiodes then disconnecting them from the reset voltage. The voltage across the photodiodes will drop as the exposure to light accumulates so that at some time, the voltages can be samples to produce image data. In the simplest CMOS sensors, this sampling is accomplished by transferring the voltages from a row of photodiodes selected by a row counter to a bank of capacitors at the bottom of the sensor so that these can be sequentially read out of the device by a column counter. Unlike the readout process in a CCD, CMOS readout does not move the actual charge generated by the incoming light. Only the voltage this charge generates is moved – actually, copied – from the pixel to the readout capacitor to the output. Significantly, this is a non-destructive process. After readout, the voltages are still in the pixel circuits so before the next image can be taken, the photodiodes have to be reset. The methods of reset and readout offer the option of a variety of pixel circuits. All of the configurations described here are of the active pixel sensor (APS) variety. Passive pixel sensor (PPS) configurations are slightly simpler but suffer from numerous performance deficiencies and so are rarely encountered.

10.2.8.3 Pixel Structures

CMOS allows a wide variety of sampling and amplification circuits to be incorporated into the pixel area so the actual variety of these circuits is quite broad. The complexity of these circuits is generally limited not by the lack of ideas for functions to be incorporated but by the lack of space to put them. As a result, the design emphasis usually centres on developing ways to maximize function while minimizing the circuitry needed for implementation. Much of the effort in these designs involves optimization of the structures and processes in the silicon itself.



◘ Fig. 10.11

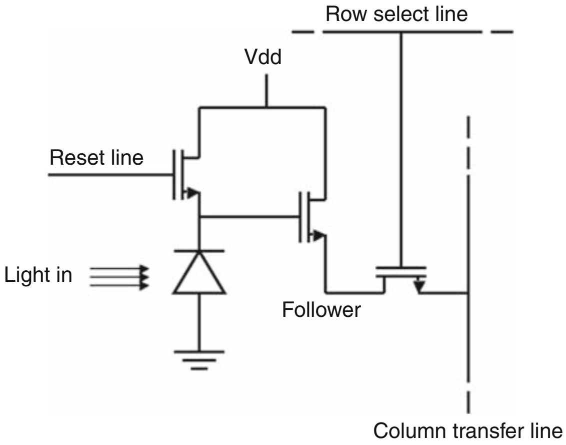
Basic CMOS sensor consisting of an array of photodiodes, each supplied with sampling and switching circuits

This level of examination is beyond the scope of this book but references are provided for those who are interested in the details of silicon device design and fabrication.

3T Pixel

The most common pixel structure is the three-transistor (3T) readout. A typical example is shown in ➋ *Fig. 10.12*. To initiate an exposure, the reset gate is turned on by a signal applied to the reset line. This connects the diode to the V_{DD} supply to fully reverse bias it. Reset may be applied to all photodiodes simultaneously or, in some sensors, to a row at a time. After reset, the voltage on the diode will slowly fall as the light arrives. The diode voltage is presented to the gate of the follower transistor. When it is time to read out a row, all of the column transfer transistors in that row are simultaneously turned on and the diode voltage is transferred through the column lines to the row of storage capacitors at the bottom of the array. Immediately after this transfer is complete (typically a few microseconds), the diode can be reset and the next exposure cycle can begin.

While this structure is simple, it suffers from two important problems. First, it has a high noise floor because the reset noise from the photodiode is included in the signal sent to the output. Since this noise is never independently measured, it cannot be subtracted as is done with CCDs. Second, since there is only a single row of output capacitors, only one line can be transferred from the array at a time. The next cannot be transferred until the readout is complete. This means that if all of the photodiodes are reset at the same time, the exposure of the photodiodes at the end of the readout will be longer than for the photodiodes at the beginning, producing top-to-bottom shading in the image. Three solutions to terminate the exposure of the whole sensor simultaneously are commonly applied. A mechanical shutter can be closed to allow readout in the dark. Similarly, the imaged area can be kept in the dark except during a short flash of light from an illuminator. If an all-electronic exposure control method that produces equivalent but not simultaneous exposure of the rows is acceptable, each line can be independently reset at a fixed time before readout. This is the "rolling shutter" to be discussed in more detail below. For simultaneous electronic exposure, more transistors are required.



■ **Fig. 10.12**
CMOS three-transistor pixel control and readout circuit

4T Pixels

Adding a fourth transistor permits separation of the exposure and readout functions thus enabling simultaneous or global electronic shuttering. The fourth transistor is usually added as shown in *Fig. 10.13* between the photodiode and the gate of the buffer transistor. In operation, the photodiodes are reset together while the transfer transistors are conductive and then at some later time the transfer transistors are closed, fixing the voltage on the gate of the follower. This voltage is then held until the row is transferred. In this way both the start and end of the exposure are electronically controlled.

While the transfer circuit is represented as a transistor switch and a capacitor to hold the voltage, in fact the capacitor is usually another charge storage well on the device and the photodiode is pinned, thus forcing the charge into the storage well as it is created. This arrangement has the advantage that is reduces the reset noise and improves linearity. As is often the case with image sensors, the schematics alone do not provide sufficient detail to describe the actual circuit or to permit analysis of its operation or performance. If the real details of sensor operation and performance are important, some understanding of device construction is essential.

It is often suggested that another 4T arrangement might provide for subtraction of reset noise. The proposal is to add a second row transfer transistor to permit transfer of the reset noise to a second row of readout capacitors so that the reset noise can be subtracted from the signal using a differential amplifier at the readout. Unfortunately, this works only if each row is separately exposed because saving the reset noise requires a separate capacitor row for each line in the sensor. Even if this were provided, the rows would need to be transferred one at a time so all rows but the first would have some exposure added to the noise signal. Subtraction of the reset noise has to be done in the pixel and this requires even more transistors.

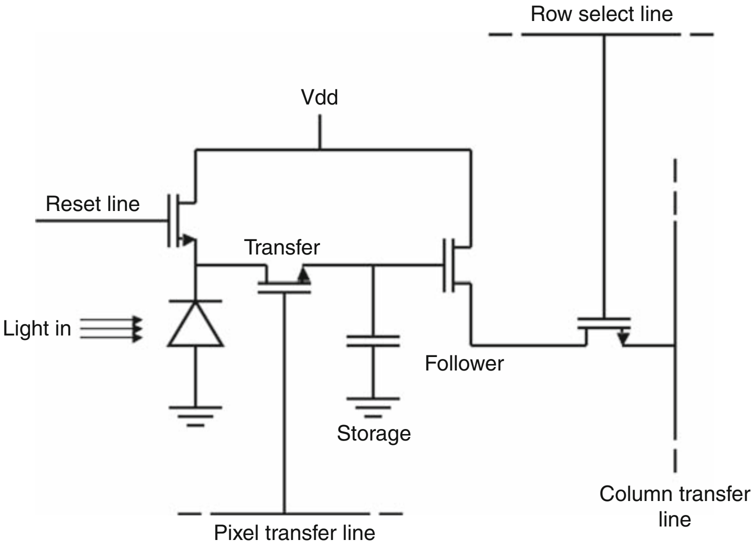


Fig. 10.13

By adding a fourth transistor to provide global electronic shuttering. This is achieved by separating the exposure and readout functions

5T and More

With five transistors, it is possible to configure a pixel circuit that stores the reset noise and the signal both. With six transistors, it is possible to do the actual noise subtraction in the pixel. With six or seven transistors, the dynamic range of the sensor can be increased by multiple sampling. More transistors can be added to allow collection of many images very close together in time. A sensor has been built that has 16 sample nodes per pixel that can be sampled only 1 μs apart, the equivalent of one million frames per second for a short time.

Unfortunately, all of these transistors and the lines to control them take space on the silicon die so adding functions either reduced fill factor or forces an increase in the pitch of the pixels. That is why CMOS sensor with a lot of functions tend towards large pixels. Large pixels drive increases in cost.

2T and Less

One the other end of the scale, where cell phones reside, the push is for tinier pixels to reduce both the cost of the silicon and the thickness of the lens-sensor assemblies while continually increasing resolution. In these sensors, configurations have been developed that allow sharing of some of the transistors between photodiodes. For example, if speed is not an issue, it should be possible to use the same follower and row select transistors for multiple photodiodes if the photodiodes have transfer transistors as in ➋ *Fig. 10.13*. These transistors could be turned off until each of the pixels is to be read out. If, for example, the readout circuit is shared by four photodiodes, the then the result is seven transistors for four photodiodes or 1.75 transistors per pixel. Many arrangements of this type have been designed and implemented for applications where space is at a premium.

Other Configurations

Many CMOS sensor engineers have designed other circuit configurations to provide specific advantages in particular performance measures. The active column sensor (ACS) architecture, for instance is intended to improve fixed pattern noise while reducing the space taken by transistors in each pixel. Sensors have been built with JFET switches instead of MOSFET to reduce noise. Photodiodes have been stacked one above the other to provide multiple spectral channels from the same area on the sensor. Pairs of photodiodes in two sizes have been provided to increase dynamic range. Comparators have been put in each pixel for direct digital output. The variety of configurations continues to expand to meet the diverse performance requirements of the many imaging market segments.

10.2.9 On-Chip Functions

While the transistors that must support each pixel are problematic because of the space they require, transistors outside the photosensitive area can be used to provide a wide variety of support functions, culminating in the complete system on a chip. Because essentially any analogue or digital circuit function can be implemented in CMOS, the capability of the circuitry that can be placed around the photosensitive area is essentially unlimited encompassing analogue-to-digital converters, microcontrollers, high-speed serial links and many other functions. In practical devices, four broad types of functions are commonly implemented –control, data conversion, processing and transmission.

10.2.9.1 Control Functions

One major advantage CMOS sensor offer in comparison to CCDs is the capability to include on-chip control functions for scanning and exposure. With CCDs, essentially all control must be provided externally to a set of pins that accept only very simple signals like phase drivers and tap selection. CMOS sensors, on the other hand, can include counters, registers and even microprocessor cores and so can execute complex sequences of operations at very high speed on the receipt of external high-level commands. Examination of any current CMOS sensor data sheet will reveal how far this capability has been taken. Many sensors now have hundreds of control registers to set internal clocking, scan configurations, power-saving modes, readout configurations bias voltages and almost anything else that could be controlled in a complex circuit. Most of these controls are useful only in the camera design process but some of them translate into very useful camera functions that can be set by users.

Scan Control

In the most common case, a camera is operated with a periodic scan of the entire area of the sensor. For general viewing or producing television shows, this may be all that is ever necessary. In many industrial, scientific and even viewing applications, changing the scan in some way from simple periodic repetition can provide useful benefits. Even in television, all of the cameras must scan together so that their signals can be easily mixed and switched. In television, though, the cameras all run at the same rate so that the timing for the next picture can be figured out from the last one.

Instrumentation situations are not so predictable so it may be necessary for the camera to produce images on receipt of random external commands. This is called triggering. There is, of course, nothing inherent in sensors that prevent them from starting the generation of images at any time. However, any imager will accumulate charge in its photodiodes when it is waiting so the black level of randomly triggered images will have black levels that vary as well. These could, in principle, be subtracted out if they were pure offsets but, unfortunately, sensor dark charge will contribute non-uniformities and shot noise and reduce the dynamic range available for the real image.

To avoid this, it is useful to be able to reset the imager rapidly upon receipt of a trigger so that the dark charge is zero when the exposure begins. While various techniques can be applied to CCDs to continuously sweep dark charge from the photosensitive areas and the transport registers, the problem remains that any stray light reaching the sensor will generate ghost patterns that are essentially impossible to remove because the entire array cannot be cleared at once. IT sensors are somewhat less susceptible but still cannot avoid the effects entirely. In CMOS sensors, on the other hand, all of the photodiodes can be reset rapidly and simultaneously to provide a uniform, near-zero starting point at any time. Many CMOS sensors can also reset specifically-selected rows. Other reset patterns can be readily implemented by providing separate sets of reset lines for the desired photodiode groups.

Scanning only part of the active area is sometimes desirable to concentrate data gathering in a particular part of the image or to speed up the frame rate. This is called region of interest (ROI) scanning. While rudimentary ROI scanning is possible with CCDs (where it is typically called partial scan) the complete independence of each pixel from the others in CMOS sensors allows very sophisticated selection of areas to be scanned. The typical CMOS sensor has as in its controls methods of setting the number of the first row to be scanned and often the number of rows to be scanned at once and the number of rows to be skipped before the next scan. The end of scanning is generally controlled by simply stopping the counters that select the rows for scanning. Similar controls can be applied to the row of output capacitors so that only the columns of interest are passed to the output stage. Using these controls together allows great flexibility in matching the scanned area and its configuration to the data requirements of the current scan. As an example, consider the following:

► Let us say we wish to make a camera that will rapidly sample selected areas of the scene to provide exposure and focus information. On command, the camera will then take two pictures, one at full resolution for use as the primary image and one at reduced illumination to display on a small LCD panel. To do this, we first command the sensor to scan a series of small areas in defined positions. This can be done rapidly because the areas are small. We have to design the camera to quickly change the row and column start settings so that the scanned area can be repositioned after each small scan. When the trigger is received, the entire sensor is reset and then scanned twice. First it is scanned with a raster in which we read only every fifth line and column (this is for the display) and then it is scanned again, without resetting, and all of the pixels are captured. All of the setting changes can be easily implemented with appropriately capable CMOS sensors.

Additional options are available. If the scan counters are reversible, then the sensor can reverse scan directions on command. If multiple start points can be set, then multiple ROI selections can be scanned at the same time. The options can be increased further if 4T or more pixels are used because these provide for storage of images in the array itself. It can be safely said that few, if any, image sensor manufacturers explicitly identify all of the possible scan combinations available on each sensor and it is certain that no camera manufacturers accommodate all of them in any existing design so engineers needing special operations are well-advised to raise them with the sensor or camera manufacturers to determine if some sensor is actually available that might meet the requirements.

Exposure Control

Since most CMOS sensors are 3T or some equivalent or, if they are 4T, have severe constraints on the use of the photodiode independently from the storage well, the availability of global shuttering is severely limited. What this means in practice is that the initiation of the exposure can be controlled electronically – it is just the end of reset – but the termination of the exposure cannot. The exposure for a particular row ends only when its contents are sampled and transferred to the row of readout capacitors. Therefore, the exposure time of each row as the scan moves down the sensor is one row longer than the previous row. If the exposure is set to match the duration of the scan, for instance, then the last row will get twice the exposure of the first row. There are some ways to mitigate this problem.

The simplest mitigation in instrumentation settings is to control the illumination so that the sensor is in the dark after exposure. A mechanical shutter can provide this darkness if it blocks the light from the entire sensor simultaneously. Such a condition is not necessarily easy to provide because any shutter near the sensor will obscure different parts of the sensor at different times in a circular or some sort of sweeping pattern. This lack of simultaneity may not be a problem if the time to close is short compared to the exposure time. Real simultaneous exposure can only be accomplished if the shutter is in the position of the iris in the lens or in some similar optical plane. In situations where the object to be viewed can be enclosed, the exposure can be controlled entirely by turning an illuminator off and on at the right times. Care must be taken to exclude stray light, especially if the exposures are short compared to the scan time. It is important to note that the onset of illumination, either by a shutter opening or a light turning on, does not have to be the start of the exposure because the sensor can be held in reset and used as the starting time. This can be very useful when an object is anticipated to arrive for capture but its exact time is not known. The illumination can be initiated slightly in advance while the exposure start can be held until the object is actually detected. Ending reset is generally much faster and more accurate than changing the illumination state.

CMOS can provide a fully electronic exposure method if the sensor is provided with two independent row counters. If one of the counters is used to command reset of the selected row and the other counter controls reading, then simply by establishing a fixed delay (usually measured in multiples of the row period) between reset and read, all rows can have the same exposure time. For example, if row 25 is reset as row 1 is read, then, in the next cycle, row 26 would be reset and row 2 read. After 25 repetitions of this, row 25 would be read after having accumulated exposure for 25 row periods. This spaced arrangement is moved down the array so that all rows see an exposure of 25 rows. The start has to be handled to be sure the rows there get the right exposure but this requires only a simple logical control. The effect, though, is of providing a variable, uniform exposure simply be setting the number of rows between reset and read. A little additional logic can stop the scanning to provide exposure of any length, not limited by the number of actual rows in the sensor. This process is called rolling shutter scanning.

The difficulty with the rolling shutter is that exposure is not simultaneous. Each row has an exposure time (in the absolute sense) that is slightly delayed from the exposure of the previous row. The total difference in starting times will be exactly the time needed to scan the sensor once. While the delay is not important if static objects are to be imaged, moving objects will suffer geometric distortion. Objects moving perpendicular to the rows will be stretched or compressed depending on their direction relative to the scanning and objects moving along the lines will be tilted one way or the other. The significance of this depends on relative speeds and the requirements for geometric accuracy in the images. Oddly enough, the digital still camera with a mechanical shutter is often offered as the ideal way to use a sensor that does not have electronic exposure control. However, the shutters in such cameras are of the curtain type, consisting of two metal leaves that move across the image – one to start the exposure and one to end it. At speeds above about 1/125 sec, the terminating blade starts moving before the initiating blade has completed its travel. By the time the shutter gets to 1/1,000 sec, the aperture the leaves form is a vertical slit that traverses the image, providing an exact mechanical equivalent to the electronic rolling shutter and producing the same geometric distortions.

To provide electronic exposure control at both initiation and termination, an extra storage node is needed in each pixel. This implies at least a 4T pixel and perhaps more. The extra transistors take space and will reduce fill factor at least until sensors can be built with transistors buried under the photodiodes.

Data Conversion

Even though the control and scanning of CMOS sensors is all digital, the output signal is still analogue. It begins as a charge packet, is converted to a voltage and is then passed along from one storage capacitor to another until it emerges from the sensor. There are a few exceptions to this signal flow but they are reserved for special applications. However, building an analogue to digital converter (ADC) in CMOS technology is not difficult so sometimes these are added to the sensor chips itself to provide a digital output. Usually, the sensor designer does not have to design the ADC because these are available under license as silicon intellectual property (IP) modules designed to be dropped into new designs. Several types of ADC are useful for on-chip incorporation

The simplest type is the common flash video ADC that is applied at the output of the image data path. Instead of being sent off-chip, the analogue signal is directed to the input of an ADC where it is converted to digital data which is then made available on package pins in either serial or parallel form. This is a direct match of the typical configuration in which a flash ADC is connected to the video output of the sensor. The requirements to use an ADC inside the package properly are essentially the same as those for using an external ADC so some additional on-chip circuitry must be provided. To accommodate the various ranges of signal level, a programmable analogue gain stage is needed before the ADC input to assure that the signal levels are matched to the full-scale input of the ADC. If this is not done, then the ADC input range has to be set to accommodate the saturation signal of the sensor. This is acceptable as long as the typical use of the sensor produces highlights that are near saturation. However, if the application only produces maximum signals that are, say, one half or one quarter of saturation then the full range of the ADC cannot be used. An 8-bits ADC with a quarter signal maximum is then functionally only a 6-bit ADC. To use the full 8 bits with this low signal, a gain of four is needed before conversion. A programmable gain amplifier will accommodate this. In addition, level-shifters, buffers, differential amplifiers or other circuits may be needed to match the analogue signal to the ADC input and to maintain stability over temperature and power supply variations.

In a very fast sensor, a single flash ADC may be insufficient to handle the required data rates so the sensor can be partitioned, typically into groups of columns, with each partition having its own ADC. This arrangement has three problems. First, flash ADCs take a lot of power, perhaps defeating one of the purposes of selecting a CMOS sensor in the first place. In addition, they generate a lot of heat. This is undesirable because heat raises dark current. Further, in CMOS sensors, multiple ADCs would have to all be positioned along one side of the sensor so the heating of the sensor would be uneven probably resulting in shading in the image. Finally, flash ADCs take up a lot of space and, in sensors as in all silicon devices, space is money.

The common solution for high-speed sensors is to use slow ADCS at the bottom of each column, then combine the outputs digitally. Since these ADCs can be of the compact, low-power delta-sigma type, they fit more easily and have much less heating effect than flash converters. Of course, there are many more of them. A 1k × 1k sensor would have 1,000 ADCs on-chip. These column ADCs have some performance advantages because they have such low sample rates. They are much easier to design because their clocks are so much slower than flash clocks. This leads to easier selection of a data sampling point and simpler calibration. Column ADCs can also more easily provide data with more bits because increasing the bit depth only requires increasing the ADC clock rate. At the highest frame rates, the column ADCs may need to include low-bit flash sections to keep the clock rates manageable.

Processing

CMOS sensors can include both analogue and digital image processing on-chip. The analogue functions are generally simple. As previously described, these might include reset noise subtraction, or adjustment for gain and offset. Digital functions can be more complex and are often aimed at reducing the data rate from the sensor or reducing the number of pins required. Of course, there is a tradeoff, more processing functions take up more silicon area and so the silicon area and the packages get bigger, increasing cost. Some of the more common on-chip functions are:

- Compression – Generally, the simple types like run-length encoding or delta coding are used. When these are operating, the camera designer must be prepared to reconstruct the original data before applying further processing.
- Automatic gain control – Rather than depend on the camera for exposure control, a sensor might include a circuit to measure the illumination in selected areas and adjust the exposure to keep the signal level optimized. Data from such a sensor does not provide any absolute measure of scene brightness unless the gain control signal is also made available as an output. A similar control might be applied to black level.
- Exposure balancing – Under different light sources, the relative amount of energy received by the colour channels can vary substantially. To optimize dynamic range, a senor could adjust the exposure time of the photodiodes in the three channels separately to produce outputs of similar amplitude. The sensor might even be made capable of remembering relative levels of the channels for white objects to simplify colour balance.
- Post-processing – If desirable, any digital operation can be included such as noise smoothing, gamma correction, edge enhancement or even operations as complex as retinal simulation or feature detection.

Fundamentally, the reason for incorporating processing on-chip is to provide adjustments at the earliest possible point in the data stream to minimize data loss. This is not quite the same as the camera-on-a-chip concept in which many functions, including some not directly involving the sensor, are included on one piece of silicon to minimize camera size and assembly cost. All additions to sensor functionality come at a penalty of yields, size and heating at the chip level and of flexibility at the system level. Once a function is added to a chip, its maximum capability becomes the capability limit of the system.

Data Transmission

Data transmission complexity is directly related to speed. In sensors with analogue outputs, higher speed means more taps an, as a result, more pins. The pins for multiple signals in analogue sensors must be carefully spaced so that there is no inter-tap crosstalk. Digital sensors also need more pains as speeds increase but the relationship is not so direct because of the variety of digital channel choices available.

Simple digital output sensors have an ADC that puts out data in parallel form, which is connected directly to a set of pins. For one tap, this is a good arrangement because it requires minimal circuitry to receive the data. Adding taps makes the pin count rise proportionally and adding bits to the data has a similar effect. While packages with hundreds of pins are available, the size of these would often be far larger than the sensor die so a better scheme is needed. Multiplexing can be employed. In the parallel multiplex scheme, the data from several taps (or from many column ADCs) is sequenced into a single set of output pins. This raises the data rate but digital transmission can be much faster than the typical maximum clock rate of a single tap. With this scheme, care must be taken to assure that skew in the data timing between outputs does not become so large that the data gets scrambled.

To avoid the skew problem, the data from one or more ADCs can be sent through a serializer and transmitted on a single pair of pins as a balanced high-speed (often LVDS) signal. Since the entire data stream passes through a single pin pair, the skew problem is eliminated. Even if multiple data streams are needed to handle all of the data, deskewing codes can be added to the serial data stream to support resynchronization. Serialized data sent through LVDS or other drivers can be sent much longer distances than can parallel data providing additional flexibility for the camera designer. If this technique is to be used, the selection of data format and driver type should be matched to some sort of commercially available receiver/deserializer to minimize the need for custom design.

10.2.10 Other Sensor Types

While the vast majority of sensors are of the raster-scanned, rectangular, monolithic, silicon type in either CCD or CMOS process producing a voltage that represents light converted to charge in depleted silicon, there are other types of devices made, mostly to server special purposes. Thousands of types have been made, many in only a few samples in sensor design labs but some have found real commercial applications. A selection of these is included to suggest the diversity of what can actually be made. One caveat – typically, sensors with special capabilities will be available from only one source.

10.2.10.1 Charge Injection Devices

The charge injection device (CID) is another form of MOS device. Unlike most sensors, it does not have photodiodes. Instead, it consists of a silicon layer overlaid with an insulating oxide with thin metal lines in a row and column arrangement as shown in ➋ *Fig. 10.14*. Voltages are applied to the lines to produce a charge storage well that can then accept incoming photons and convert them to charge. The odd thing about CID sensors is that since the row and column lines create a potential well around their intersection, the centre of each pixel is obscured. The boundaries of the pixels, however, are unobscured, just the reverse of essentially every other sensor geometry. By changing the voltages applied to the electrodes, the charge can be made to

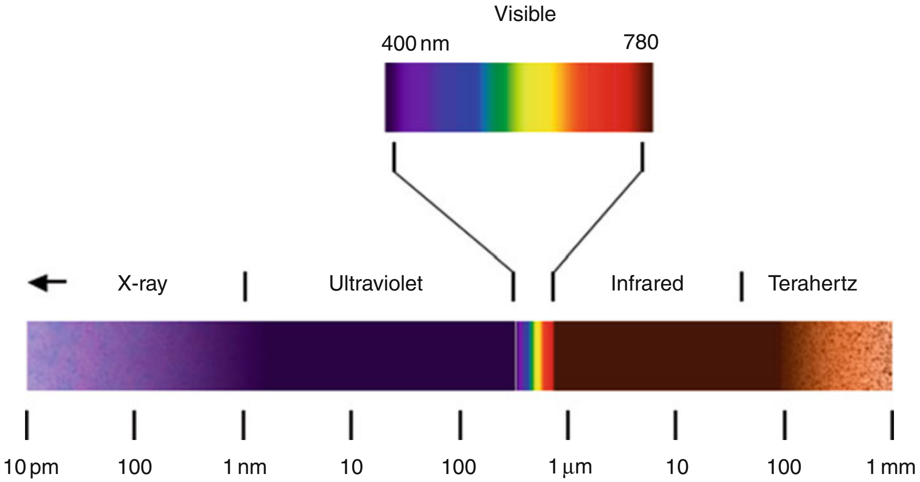


Fig. 10.14
The visible spectrum is shown relative to the near-infrared (NIR) and near-ultraviolet (NUV) bands. This suggests that these wavebands may be imaged using techniques similar to those in the visible part of the spectrum (VIS)

move and generate a voltage that can be read out at the ends of the row electrodes. The motion is non-destructive so it can be repeated to provide images showing the image as it builds up. When the exposure is over, the charge is injected into the silicon substrate, hence their name. The injection can be read as a signal as well if the pixel crossovers are injected separately in a raster sequential fashion for either the full array or one or more ROIs.

Because CID sensors have no junctions and no transistors in the imaging area, they are very resistant to radiation exposure. This makes them suitable for use in monitoring cameras in nuclear reactors and other high-radiation environment areas. In the CID structure, there are no communication paths from pixel to pixel even in heavy saturation so blooming is essentially non-existent. In addition, CID pixels can be made to store relatively large amounts of charge, making them suitable for spectroscopic applications. Unfortunately, the high capacitance of the row and column lines makes the conversion factor for CID arrays very low and contributes substantial noise. Nevertheless, the CID continues to serve several important niches.

10.2.10.2 X-Y Arrays

CID sensors are X-Y arrays because they have pixels that can be randomly selected. Pixels from any sensor can be randomly selected after the sensor is read out but in X-Y arrays, the scanning process itself can be random. In theory, any CMOS sensor could be operated as an X-Y array by allowing rapid setting of the row and column decoders to select individual pixels. In practice, this setting process is usually so slow that random readout doesn't produce enough data in a reasonable time. Real X-Y arrays tend to be relatively small simply because the applications for them do not require high resolution.

A typical X-Y array has either individual pins for each row and column (small arrays) or separate ports to load the coordinates of the next pixel rapidly. Most X-Y arrays built to date run in the current mode rather than the charge accumulation mode of both CCD and CMOS area arrays. To run in current mode generally requires that the light levels be quite high so that sufficient photocurrent is generated to produce an output signal with reasonable signal-to-noise ratio. Since the maximum signal level in current-mode operation is not constrained by well capacity, X-Y photodiode arrays can have extremely large dynamic range. One million to one is possible. An interesting advantage of the X-Y array is that the photodiodes can be of any type – not only simple silicon pn junctions but also p-i-n diodes, avalanche diodes and diodes made of other materials.

10.2.10.3 Non-Rectangular Arrays

Not everything to be viewed maps well into a rectangular array of squares with constant spatial resolution. Although nearly every scene is treated as if the rectangular geometry is good enough, there are situations where non-rectangular or non-periodic arrays are more suitable. The most obvious of these is the acquisition of round objects. Clearly, the rectangular array will produce widely varying angular sample rates from centre to edge of a round object and often fail to accurately image the object centre. In addition, the constantly changing angle of the pixel array relative to the slope of a round surface leads to very inconsistent estimates of edge position. One solution is to use more pixel so that the sampling is sufficiently fine to reduce the errors to insignificance.

Another solution is to use a sensor with the pixels arranged as either concentric circles or radial lines. The difference between these two geometries is that the concentric circle type has pixels of constant size at all diameters arranged so that the pixel edges are ether radial or tangential. The pixels do not overlap in the radial direction so each circumferential ring acts as a row would in a rectangular array. Such arrays are good for determining centration and deviations from roundness but they cannot measure individual radii accurately because the centres of the pixels do not form straight lines in the radial direction. In the alternative geometry, the pixels are arranged in radial rows. The distance from the centre of the pixel boundaries can be made uniform along all radii. These arrays will have gaps because the circumferential spacing increases with increasing radius and pixels can only be used to fill in when the space between adjacent radial lines is sufficient to allow insertion of another row. Sometimes the uncovered area can be minimized by allowing the diodes closer to the centre to be smaller than the nominal size but this generates shading that has to be compensated. Radial arrays can be designed with four or six lines beginning at the centre depending on the symmetries requires in the measurements. Reading out circular arrays is straightforward in CMOS types because the rows and columns can be easily mapped to increments in radius and angle

In addition to the reasonably obvious rectangular and polar geometries, many special configurations have been built for specific purposes. Some of these are arrays of arrays. Examples of these can be found is the instrumentation for particle accelerators where small sensor may be combines on curved surfaces to match a mathematical function of a focal plane. Arrays can be stacked for detection of coincidence or for suppression of background events. In production, the two most common special devices are both made for spectroscopic applications. In optical spectroscopy, light is spread by wavelength using a prism or grating so individual wavelengths can be detected. The light to be detected always consists of a spectrum spread in one direction and a long slit perpendicular to the spread to provide enough light for robust detection. To accommodate this configuration, spectroscopic sensors are built as single lines with pixels that are close together in the spread direction and very long, up to a few mm, in the slit direction. This not only matches the optical geometry but provides very large charge storage capacity, often in the hundreds of millions of electrons. These are always MOS arrays because shifting charge packets that large by CCD transport registers is not possible.

The other spectroscopic array is much more odd. In atomic emission spectroscopy, very specific emission lines, some very close together in wavelength, must be detected. Simple prisms and gratings cannot spread these lines out enough so a two dimensional grating called an Echelle is used. This spreads the emission lines into several rows rather than just one and leaves great unused spaces between many of them. Since the entire area to be covered can be large but very sparse, the economical sensor to build consists of a set of detection rows each arranged to cover exactly one emission line. Essentially, this is a two-dimensional array of small line arrays all connected to a central readout point. Most of the silicon is unused.

Many other non-rectangular arrays have been built to address one or more limitations of photodiodes at image detectors. In some arrays, dynamic range is expanded by providing two diodes per pixel. One arrangement for this is to use large octagonal pixels and fill the square holes left between them with smaller square pixels. The centres of all of the photodiodes form a rectangular array but the centres of the octagonal and square parts taken separately make two offset diagonal arrays. The mathematics for decoding such a structure, especially when colour filters are added, is complex. Another useful configuration is the hexagonal array. This has the advantage of representing both rectangular and circular objects reasonably well and it can provide a smoother representation of the image than rectangular arrays, especially when the fill factor is low but the output almost always must be mapped to a rectangular display or at least rectangular coordinates to be used so the advantages vanish quickly. In addition, designing a colour filter array for hexagonal arrangements is problematic because there is no minimal repeating cell.

10.2.10.4 Stacked Sensor Arrays

While two dimensional arrays of photodiodes are enough for most imaging applications, more and more three-dimensional arrays are being developed to support detection of scene features in addition to spatial variation in brightness. Some of these arrays are literally three-dimensional and others encode the third dimension by partitioning the two-dimensional surface much like colour filter arrays are used to produce colour (really two more dimensions) from a single plane of identical photodiodes. Taking colour as a starting point, consider the arrays of stacked photodiodes in which colour separation is accomplished by use of the inherent wavelength filtering behaviour of silicon. Three diode junctions stacked at appropriate depths in the silicon can separate visible wavelengths sufficiently to produce colour images. A device of this type must have complete sets of scan and readout electronics for each of the three layers in the diode stack so the pixel areas tend to become crowded with transistors. This structure can only be implemented in CMOS because the CCD process does not allow enough stacked wells to be defined for three layers. Two layers have been built in CCD structures but never commercialized.

Substantial work has been done in stacking arrays for infrared imaging as well. This is covered in detail below. As an introduction consider that every semiconductor material becomes transparent to photons with energies below the bandgap energy. In silicon, this occurs around 1,100 nm. A dual-band sensor can be readily made that has a long-wavelength detection array placed behind a silicon detector array. If the silicon is packaged so that radiation passing through it is not blocked, then the longer wavelengths can be detected by the second array. The problem here is generally optical – how to design a lens that will focus the two wavelength bands at the correct distances and transmit both bands.

Instead of supplying spectral information, arrays can be built to indicate distance. Several types of these have been built that rely on the demodulation of time-varying active illumination. In all cases so far, the structures required to detect the phase of the illumination use pixels that include at least two detection elements. Thus distance is detected using a partitioned planar array. The variety of special arrays that can be built is endless, limited only by the funding available.

10.2.10.5 Intensified Arrays

As described above, CCDs can be fitted with special high-voltage shift registers to provide on-chip gain for low-light-level imaging. Unfortunately, there is no equivalent for this in the CMOS structure. Nevertheless, methods of providing imaging with very limited illumination using CMOS do exist.

First, a small refresher on vacuum image intensification devices. Certain materials will emit electrons when placed in a vacuum and bombarded with light. This is the photoelectric effect that led Einstein to conclude that light could work as particles. In practical devices, the emitter (called a photocathode) is deposited on glass in a vacuum and fabricated so that photons arriving on the glass side of the photocathode cause electrons to be emitted from the opposite surface into the vacuum. For imaging, the photocathode may have a diameter of 18 or 25 mm. In a camera, a lens is placed in front of the photocathode so that the optical image is converted to a pattern of emitted electrons. By application of appropriate electric fields, the electron pattern can be focused on some material that will multiply the electronic charge and hold the charge for readout. Thus, the final signal is stronger than the original photon image could have produced if just focused on a silicon image sensor.

Silicon has an advantage as a receptor of electrons because for each 3.6 eV of energy carried by an electron arriving at the silicon surface one hole-electron pair will be produced. Silicon used in this fashion is called electron-bombarded. In typical operation, the voltage is set so that the electron gain is about 300. This multiplication process is essentially noise-free. With modern photocathodes made of semiconductor materials like gallium arsenide, nearly all the incoming photons in the visible can be converted to electrons so overall gain remains near 300. Of course, this device is not monolithic and may be susceptible to external effects due to it reliance on vacuum and electrostatic fields.

Monolithic CMOS sensors with gain have been built in small arrays. The silicon equivalent of the vacuum intensifier is the avalanche photodiode. Development of CMOS avalanche arrays is still in the early stages but arrays of 256 × 256 pixels have been produced. This remains a technology to watch for low light imaging.

10.2.10.6 Very Large Arrays

As machine vision imaging tasks become more complex and as larger fields of view and higher resolution are needed, the image sensors used need to have more and more pixels. Even arrays up to 4k × 4k can now be found in vision systems. Since the optics often limit how small the pixels can be, high-resolution often means very large areas of silicon. The size of individual sensors is usually limited by the maximum area that can be exposed in one shot in the photolithography systems used to lay down the patterns in silicon. In a typical fabrication facility, this size is on the order of 25 × 30 mm, more or less. While that may seem large for typical inspection applications using even megapixel sensors, it is not as large as a 35 mm film frame and it is much smaller than the latest line scan sensors used in LCD glass inspection, which are up to 90 mm long.

Three fabrication techniques are used to move beyond the constraints of the projection lithography aligners. The first is simply to use a different type of lithography system in which the masks containing the patterns are placed in contact with the silicon. Since these masks can be made as big as the silicon wafer, the sensors can be this large, too. In fact, many devices using wafers up to 125 mm in diameter have been fabricated this way. Unfortunately, this technique is completely incompatible with almost all device fabrication facilities so it is essentially relegated to specialty production.

A second technique uses small sensors designed to have their active areas placed in close proximity to each other to simulate a larger single device. This technique is called butting. Butting can be accomplished by either designing sensors with pixels very close to the edge of the cut silicon or by providing fibre optic bundles that connect slightly separated sensors together using a taper in the fibres. The fibre optic method supports easier sensor design but is subject to some distortion in the image and is very expensive. Area sensor designed for one, two or three sided butting have been made but are found only in specialty situations. Line scan sensors designed for butting, however, are found in every documents scanner and fax machine using a contact image sensor. The critical factor in successful butting is minimizing the dead space between sensors without damaging the outer rows of photodiodes. Typically, the gaps can be many tens of microns but some butted line scan assemblies have gaps as low as 15 microns.

For large monolithic sensors, a technique called stitching is used. Instead of being exposed in one shot, stitched sensors are exposed in many shots carefully aligned at the edges to produce a continuous array that has no gaps. To design an array for stitching, it is necessary to break the layout of the sensor into small parts that can be repeated on the silicon to build up the desired full array. For instance, if an 8k × 8k CMOS image sensor with 10 micron pixels is needed, the basic array might be just 1k × 1k and the registers and drivers and readout structures around the active part of the array are broken into centre and end and corner pieces that are put back together on the silicon. Because silicon exposure alignment can be done with sub-micron accuracy, the boundaries between exposures have insignificant effects on operation and performance when stitching is properly done. However, CMOS imagers are much easier to stitch than CCD imagers because CMOS does not require building charge transport registers that work across the stitching boundaries. Essentially in large CMOS imagers only the aluminium interconnecting wires need to be stitched.

10.3 Imaging Outside the Visible

Without optical or electronic aids, humans can see only a small portion of the electromagnetic spectrum, that part with wavelengths between 400 and 780 nm known self-referentially as the visible spectrum. In many situations, the use of wavelengths outside the visible band can reveal invisible features useful in measurement or monitoring. In ➋ *Fig. 10.14*, the visible spectrum is shown relative to it nearest neighbours, infrared and ultraviolet, which may be imaged using techniques similar to those in the visible and relative to its more distant relations, which require very different imaging techniques. A survey of these bands and their associated techniques will explain their usefulness.

10.3.1 Infrared

Infrared is that portion of the spectrum adjacent to the red end of the visible spectrum and extending to about 30 μm. In ➋ *Fig. 10.15*, the infrared region is expanded to show its various named components. The imaging requirements and detector materials for each of these bands is discussed below. Infrared imaging comes in two types – active and passive. Active infrared imaging is much like visible imaging in which an object is externally illuminated and the results of this illumination are imaged. In passive imaging, the object itself provides the illumination, in the form of emission of electromagnetic radiation. Of course, passive imaging of very hot objects is possible in the visible band but passive imaging of lower-temperature objects is much more common and more widely useful. Since detectors in each of the infrared bands is always

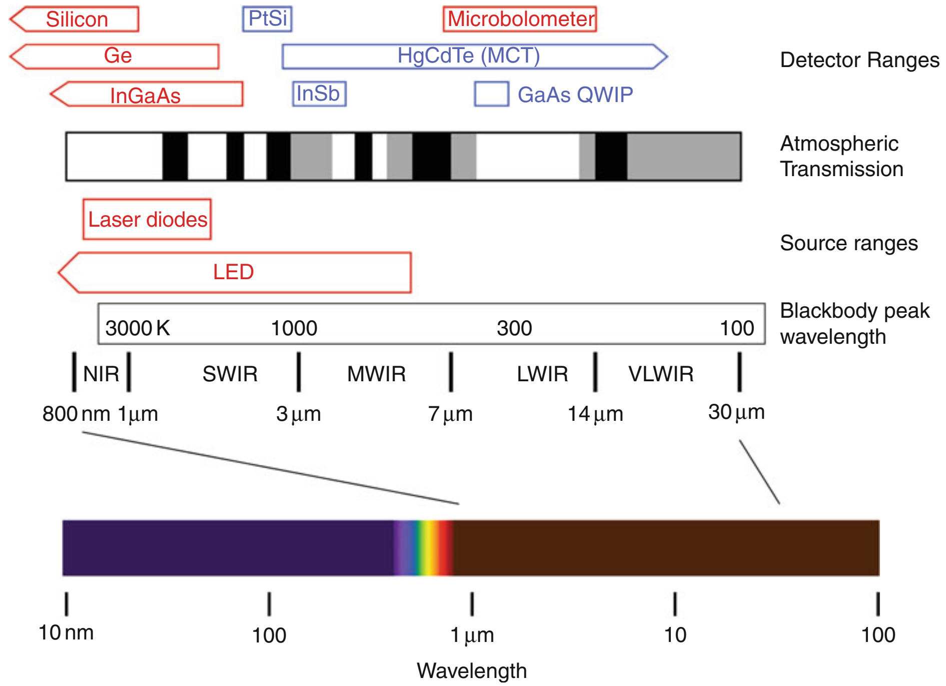


Fig. 10.15
Expanded view of the near-infra-red part of the spectrum

sensitive to emitted energy, careful design of cameras to minimize the effect of stray emissions, even from the camera itself, is essential.

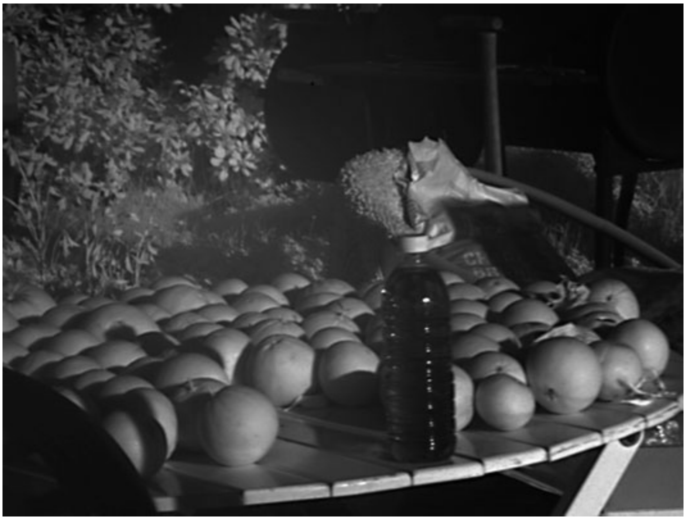
The optical absorption characteristics of the atmosphere have led to the separation of the infrared spectrum into bands. Near infrared is composed of wavelengths adjacent to the visible while the rest of the bands fall farther and farther away. Each band has its own set of useful detection and optical materials that are roughly as exotic and expensive as their relative distance from the visible band. One practical consideration in the implementation of infrared imaging is that essentially all sensors and cameras operating beyond the near infrared are export-controlled by the U.S. government.

10.3.1.1 Near Infrared

In spite of the first atmospheric transmission limit around 1,200 nm, the extent of the near-infrared band is essentially defined by the long-wavelength photon absorption limit of silicon – about 1,100 nm. Because the transparency of silicon increases rapidly beyond 800 nm, thick image sensors are needed to capture a significant fraction of photons beyond 800 nm. In most CCD and CMOS sensors, even though the silicon may appear to be several hundred micrometers thick, the actual photon collection depth is defined largely by the depleted region, which may only include a few micrometers at the input surface. Even those sensors constructed to collect photons through the full thickness of the silicon still only have a few percent efficiency at 1,000 nm.

The advantage of using the NIR band is that the same sensor, and sometimes the same optics, can also be used for visible imaging. This provides additional flexibility in selecting illumination or filters to produce contrast where different wavelengths the visible alone might not be sufficient. For example, bruises can be identified in fruit by NIR imaging. The photograph in ➲ *Fig. 10.16* taken using a CMOS sensor in sunlight through a 1,064 nm filter, shows several potentially useful NIR effects. Most significantly, the opacity of water at 1,064 nm is apparent. Even a few centimeters of water appears nearly black. In addition, the high NIR reflectivity of foliage is demonstrated as it the transparency of the PETE plastic used to make the water bottle. Another common use for NIR is imaging through dyes. Most coloured dyes are transparent beyond about 800 nm. Under NIR illumination all of the dyes disappear so that the underlying material can be seen. This is the principle used in film digitizing to remove scratches from the scanned image. Three colour channels detect the RGB components while a fourth channel with only NIR response sees through the colour dye layers and views only the film substrate material. When this channel is subtracted from each of the colour channels, the scratches disappear. Dye transparency in the NIR also lies behind the capacity for image sensors with colour filter arrays to see at night using NIR illuminators and imaging of lower paint layers in works of art.

NIR imaging allows imaging hot items by their own emissions, but temperatures of at least 800–900 K are required to produce enough energy in the NIR band. This is typical of soldering irons, some industrial processes and some invisible flames. Heated objects can be imaged by sensors extending to 1,000 nm at temperature below those where the eye will see a dull red glow. When the eye can see the glow, then any visible band sensor will do.



◘ Fig. 10.16
Near infrared image taken using a CMOS sensor in sunlight through a 1,064 nm filter

10.3.1.2 Short-Wave Infrared

Short-wave infrared (SWIR) overlaps with the boundary of the visible, starting around 900 nm, and extends to the atmospheric (and glass) transmission cutoff around 2.5 μm. While SWIR imaging has been in use for temperature profiling of boilers and other equipment and for some aerial surveillance for over 30 years, only the ascendance of infrared laser diodes in telecom applications provided sufficient funding of the technology, especially sensors, to advance the use of this band significantly. For many years, the detector of choice for SWIR was a lead sulfide (PbS) photoconductor in a vidicon-style image tube, but now the pre-eminent detector material for SWIR is the solid-state indium gallium arsenide (InGaAs – pronounced in-gas) array even though most InGaAs detectors respond only to 1.7 μm. The complicating factor in the use of InGaAs arrays for imaging is that InGaAs itself cannot serve as the substrate for the readout circuitry so that the detector array must be bump-bonded to a matched readout array made of silicon. This leads to relatively large pixels compared to those in silicon imagers. The typical range for InGaAs sensors is 25–50 μm but pixels as small as 12 μm have been put into production. The large pixels force larger sensor sizes and raise costs for both the sensors and their optics but they also allow the inclusion of complex silicon circuitry for each pixel to support pixel-rate correction of sensitivity, dark current and fixed-pattern noise and for complex control circuits similar to those found in sophisticated silicon CMOS imagers. The fill factor of InGaAs sensors can be quite high because the circuitry is not in the same plane as the detector. In addition to area arrays, line scan sensors are also readily available. These were originally designed to be used as channel power monitors in telecom systems but also serve well in line-scan cameras. Low-light-level SWIR sensors have also been built, using an InGaAS layer as a photocathode in an electron-bombarded intensifier structure.

InGaAs is actually responsive from the visible through the SWIR range but the visible is blocked from detection because the InGaAs layer is grown on an indium phosphide (InP) wafer substrate. When the InGaAs is bonded to the silicon readout circuit, the InGaAs layer faces the silicon so incoming photons must pass through the InP to reach the InGaAs. Unfortunately, InP blocks everything below about 900 nm so the visible response is lost. Devices have been made in which the InP is etched off after bonding of the InGaAs to the silicon so that the response can reach below 500 nm. These devices are difficult to make and expensive but can provide reasonable response from visible through SWIR, minus the wavelengths blocked by the atmosphere, of course. InGaAs is generally run at room temperature for industrial and surveillance applications but cooling can improve image quality for scientific use. Cooled cameras must include a cold reference plane that can be placed in front of the sensor on command to provide calibration.

Germanium photodiodes are responsive over the full visible-SWIR range but have not found use in imaging because the lack of an insulating germanium oxide prevents fabrication of MOS transistors on the germanium wafer. Germanium could be bonded to silicon for readout like InGaAs but the assembly would have to be cooled due to the high dark current of germanium diodes at room temperature. Another possible way to use germanium is to deposit it on silicon so it can function only to generate charge while the silicon provides the diode and readout functions. Devices of this type can be expected to become more available as the process problems are worked out.

One difficulty in using any sensor in the visible and SWIR simultaneously is the extreme burden put on the optics. Ideally, lenses for this application should focus over the entire

400–1,700 nm band simultaneously. Unfortunately, such a design cannot be realized with any combination of known glasses so techniques such as diffractive elements would have to be employed. At this writing, no such lenses exist. Even if only the 900–1,700 nm band must be imaged, the design options are limited. The large pixels in InGaAs imagers relieve the design requirements somewhat but large sensors mean long focal lengths, leading to large lenses. In addition, the curvatures of the surfaces must be constrained so that the required wide-band coatings function properly. All of this leads to relatively high cost for lenses specifically designed for SWIR use, especially in applications requiring operation all the way to the 2.5 μm band edge. Lenses intended for visible use can be applied in relatively undemanding SWIR applications, especially where only a narrow band must be imaged as in telecom testing.

Objects at 400 K and above produce enough emission in the SWIR band to be detected in contrast to the background. This makes SWIR imaging useful for detecting heating due to thermal insulation defects, electrical connection problems, and even passage of heated fluids. This temperature range is common in industrial processing applications so SWIR imaging has found wide use in both on-line and off-line inspection and in examining machines and facilities for incipient failures. SWIR imaging in surveillance applications is enabled by nightglow from the sky, which is concentrated in the 1.3–1.8 μm band. Nightglow is sufficiently bright, even on overcast nights, to permit passive imaging.

10.3.1.3 Mid-Wave Infrared

The next atmospheric window, extending over the range of roughly 3–5 μm, is home for the mid-wave infrared (MWIR) band. Three detector types are used in MWIR cameras, indium antimonide (InSb – pronounced ins-bee), mercury cadmium telluride (HgCdTe, but usually abbreviated as MCT) and microbolometer. The first two are narrow-bandgap semiconductors operated while cryogenically cooled and are built as bonded assemblies like InGaAs but the third can be operated at room temperature and is built in silicon using a MEMS process. The performance differences for industrial temperature measurement applications are rarely significant. While cooled detectors provide better temperature resolution in thermal imaging applications, they are much larger, more expensive and more difficult to operate. Even so, microbolometers are found less often in MWIR applications simply because the manufacturers who could provide them concentrate instead on LWIR sensors and because the MWIR band is used mostly in defence and research applications where temperature resolution is paramount. In laboratory applications, the choice between InSb and MCT often depends on the total range of wavelengths required rather than solely on MWIR performance. InSb can detect photons to below 1 μm with appropriate optics but MCT is limited on the short wavelength end to about 2 μm. On the other hand, MCT can detect out to beyond 12 μm, while InSb extends only to slightly beyond 5 μm.

In addition to temperature measurement applications, MWIR cameras are very useful in narrowband imaging intended for detection of specific substances. This has led to their wide use in hyperspectral imaging systems in which area images are taken of many narrow spectral bands simultaneously to allow real-time detection of the presence of several target substances. In spectroscopic terms, the range of detection for the MWIR band is about 2,000–3,300 wavenumbers (cm^{-1}), which covers the detection of CH_x, various C double and triple bonds, Nitrogen bonds and several other important identifiers for organic compounds. No other band capable of imaging through the atmosphere covers as many individual bond types.

As a result, MWIR can be used for imaging uniformity, purity and distribution of a wide variety of organic compounds.

Lenses for the MWIR band are generally made of germanium and silicon and protective windows are made of sapphire (aluminium oxide). All of the usual lens types can be made – fixed focal length, switchable field of view and zoom lenses. Since the ratio of the high and low wavelengths to be covered is relatively small, correction of aberration can be fairly accurate. Diffraction-limited designs are possible. However, lenses containing germanium must be used with care because germanium can experience thermal runaway. As germanium gets hotter, its transmission drops so if it is used in situations where energy arrives at a rate faster than it can be dissipated, then the lens can overheat and suffer permanent damage. In industrial situations, it is important to avoid allowing radiation in the range of 1.5 μm and beyond to fall on a germanium lens even if the radiation is not used to form an image.

10.3.1.4 Long-Wave Infrared

Beyond the next atmospheric absorption band, from 7 to 14 μm, lies the long wave infrared (LWIR) band. This is the band traditionally associated with thermal imaging because it is the band providing the best temperature discrimination for objects around 300 k. The first thermal imagers, built back in the 1960s, used MCT cooled with liquid helium. As MCT quality and readout circuits improved, the operating temperature could be allowed to increase until now liquid nitrogen is sufficient. MCT still produces the best temperature resolution in this band, down to 15 milliKelvins, and it is still the detector of choice in the most critical applications but is relatively complicated to support and expensive. As a result, MCT has been supplanted in all but the most critical applications by microbolometer arrays.

A bolometer was originally pair of metal strips, one blackened and exposed to the radiation to be measured, and the other shielded. Incoming radiation heated one strip and the difference in resistance was measured using a Wheatstone bridge. Microbolometer image sensors are essentially of the same construction except that the resistance measurement is typically single-ended – there is no reference strip. Commonly, the absorbing material, which must also have a substantial temperature coefficient of resistance, is amorphous silicon or vanadium oxide prepared to present a black absorbing surface. The strip is silicon etched by chemical micromachining so that it is suspended over an open space. This construction prevents rapid conduction of heat away from the resistive element and improves sensitivity at the expense of lengthening the time constant of the measurement. The tradeoff choice for most imaging applications is to match the time constant to the imaging frame rate, which results in temperature resolution on the order of 50 milliKelvins. The primary advantage of microbolometers is their ability to operate over a wide range of temperatures up to 60°C. This simplifies cameras design substantially. However, to avoid serious temperature-induced drift in image performance, microbolometers are often stabilized at some selected operating temperature with closed-loop thermoelectric plates.

LWIR imaging is primarily used for viewing temperature differences of a object only a few tens of degrees different from the background. The widely-known night-vision applications involving detection of people, animals, vehicles and shelters by their thermal signatures are still the most common use for LWIR cameras. However, the temperature discrimination of these devices is sufficient to reveal small differences in heat transmission by insulation or variations in industrial process temperatures or heating due to friction in machinery and vehicles. These tasks all involve detection of emitted radiation but active illumination is also involved in some uses. In industrial settings, LWIR cameras are used to monitor CO_2 laser radiation, especially for its presence in places where it is not desired because stray invisible laser beams, especially at the typical power levels used in industrial processes, can be extremely dangerous. LWIR radiation penetrates smoke well because its wavelengths are not scattered as much as is visible light. This has led to the use of LWIR imaging in monitoring processes that generate smoke and in firefighting.

Because the detector elements in LWIR sensors are essentially mechanical, they are typically larger that the silicon photodiodes in silicon visible sensors. Pixel pitches are commonly 25–50 μm although pixels down to 10 μm have been produced. The minimum pixel pitch is also constrained, as it is in InGaAs sensors, by the need to bond the detector array to an underlying readout device, which, for microbolometers, can be just a multiplexer since the signal is always available on the detector elements. LWIR line scan sensors have also been made for thermal monitoring of moving objects and panoramic thermal imaging.

Optics for LWIR incorporate primarily germanium. Because of the well-developed set of applications for this band, the selection of optics is quite broad, including fixed, multiple-field and zoom optics, in addition to reflective optics with long focal lengths.

10.3.1.5 Very Long-Wave Infrared

Beyond 14 μm the goal remains object detection and measurement but the objects are those colder than 300 K. Since most radiation in this band is absorbed by carbon dioxide in the atmosphere (the "greenhouse effect"), observation typically occurs in space where the targets are discarded shrouds, fragments of old satellites and other objects either in orbit or on space trajectories. MCT works as a detector over the shorter-wavelength part of this band, which extends to some indeterminate wavelength one side or the other until the radiation is more often termed "terahertz". At the longer-wavelength end, more exotics materials like microstructured quantum wells, QWIPs and even quantum dots are being explored for use in improved detectors. Outside space or the laboratory, little is seen of VLWIR imaging systems. Wider application will need to wait until significant industrial operations are undertaken in atmosphere-free locations.

10.3.1.6 Two-Colour Infrared Imaging

Accurate temperature measurement requires detection of radiation at two wavelengths. If only one wavelength is used, some assumption must be made about the emissivity of the material being measured. For many common materials, the assumptions are good enough to produce temperature estimates that serve the intended purpose or to produce useful images of temperature differences. Viewing leaks of heat around doors in a house, for instance, does not require an absolute measurement, only an indication that some places on the surface of the house are warmer than others. For industrial process control, though, accurate, non-contact, temperature measurements are needed. This requires measurements at two wavelengths, so a ratio of signals can be taken that cancels out the emissivity factor. Of course, if emissivity changes with wavelength then errors will occur but for most materials emissivity varies little within the range of wavelengths typically used.

While two-colour optical thermometers are common, two-colour line and area sensors are not. Various attempts have been made to supply two colours using essentially the same techniques applied to colour imaging, which is considered in detail below. Two-colour imaging is more difficult than true colour imaging with silicon detectors because infrared detectors require silicon readout circuits. Attempts to stack two imagers of different materials have been only marginally successful due to the need for the longer-wave radiation to pass through the first detector and its readout circuit and due to the large separation between the positions of the two focal planes. The recent refinement of this is the development of a single device that has two stacked layers of QWIP materials tuned to different wavelengths. This design allows use of germanium optics with simpler designs and eliminates the need to operate the two focal planes separately. Cryogenic cooling is required with these sensors but the performance is promising and manufacturing at reasonable cost is feasible. Quantum dots have also been used in prototype sensors. If these can be made to work commercially, they can offer improved selectivity and some relaxation of the cooling requirements.

The simplest way to make two-colour detectors ought to be to use HCT with different fractions of mercury and cadmium in two stacked layers. Theoretically, the results should be excellent. Unfortunately, no method to reliably fabricate devices of sufficient quality has yet been discovered. Also under development are LWIR microbolometers layered with InGaAs, arrays with alternating pixels that respond to two different wavelengths and numerous other schemes, all concentrated around defence and security requirements.

10.3.2 Terahertz Imaging

What was previously known at the sub-millimeter microwave band (and considered part of the radio spectrum) has been moved to the imaging camp and renamed the terahertz band. This term roughly identified the band starting at the high end VLWIR and extending to down to a few hundred GHz – about 0.3–6 THz, corresponding to wavelengths of from 50 μm to 1 mm. Imaging at frequencies from around 100 GHz up has a long history. 94 GHz, for instance, has been used for passive imaging through fog because atmospheric attenuation is low at that frequency. The optics used are spherical structures of graded refractive index polyethylene foam called Luneberg lenses. However, the molecular bonds in materials have no signatures at this frequency so the images tend to show featureless shapes of objects. While this may be sufficient to monitor movement of aircraft around airport concourses, it is insufficient to support identification of smaller objects.

At higher frequencies, especially above 1 THz, the composition of objects begins to become visible in images. Unfortunately, the self-radiation of the objects has insufficient power to provide these images so active sources are required. Unlike 94 GHz, where high-powered sources using standard microwave components and techniques are readily available, terahertz frequencies have no easily-implemented generators. Quantum cascade lasers and other devices are used but better sources are the continuing subject of research. In addition, atmospheric absorption is very high in the terahertz band, so the applicability of terahertz imaging is limited to short-range situations.

Terahertz radiation can be detected with a variety of small-geometry antenna arrays or by wavelength-independent detectors. Pyroelectrics and thermopiles, for instance can detect radiation at any wavelength. Unfortunately, the sensitivity of these is too low for practical imaging systems. Microbolometer arrays can be used as can more esoteric detectors such as

Pockels cells, which convert electric field strength to polarization, essentially converting the terahertz image into a visible one. Research on better, faster detectors continues.

The varying signatures of organic materials have made terahertz imaging valuable in security screening. Clothing is largely transparent while metal objects are opaque and many plastics, including those used in explosives, have unique terahertz signatures. In addition, the relatively long wavelengths of terahertz radiation make detection of phase feasible. This may enable measure thickness or other properties not easily detected just by transmission amplitude.

Although many potential industrial inspection applications have been suggested, little implementation is yet in evidence. Aside from the cost and size of the detectors necessary for terahertz imaging, the relatively low resolution, fractions of a millimeter, stands in the way of qualifying this technique for detailed inspection tasks. Perhaps, there will be important applications in detecting materials below other materials, such as in laminates or in detecting organic contaminants in surface treatments. If an application seems intractable in the visible or infrared, considering the contrast the might be generated with terahertz illumination is worth considering.

10.3.3 Ultraviolet Imaging

Imaging in the ultraviolet is full of complications – more and more optical materials stop transmitting as the wavelengths get shorter until, finally, only reflectance works. But, because more and more image sensors are capable of UV imaging and because UV imaging reveals many interesting industrial, medical and scientific phenomena, the need for high-performance, cost-effective UV camera designs in steadily increasing. Successful execution of these designs requires careful consideration of the characteristics of optics, sensors and the radiation itself.

10.3.3.1 What is UV?

Ultraviolet is the range of electromagnetic radiation occupying the wavelength range from just shorter than the visible (about 400 nm) to about 10 nm, where the photons begin to penetrate materials enough to be called soft x-rays. The UV range is further subdivided by certain practical effects into the five bands in ➲ *Fig. 10.17*. The wavelength boundaries between these bands are based on photobiological activity and may vary in different references.

- UVA is the tanning and blacklight ultraviolet. It passes through most optical materials like visible wavelengths and can generally be detected by visible light detectors. Although it is invisible to humans, many insects use UVA images to identify certain plant materials, which are highly reflective in this band. UVA above 340 nm is commonly used to stimulate visible fluorescence in paints and the like.
- UVB is the sunburn and “solar blind” ultraviolet. Most UVB is blocked by ozone in the earth’s atmosphere. In small does, it produces vitamin D but more exposure causes sunburn and other skin damage. Any intense UVB must come from local sources. Solar blind imagers, which respond only to the UVB band, permit monitoring of local sources like hydrogen flames and electrical corona in daylight. Some local sources, like quartz halogen bulbs, can emit enough UVB to pose significant UV skin hazards.

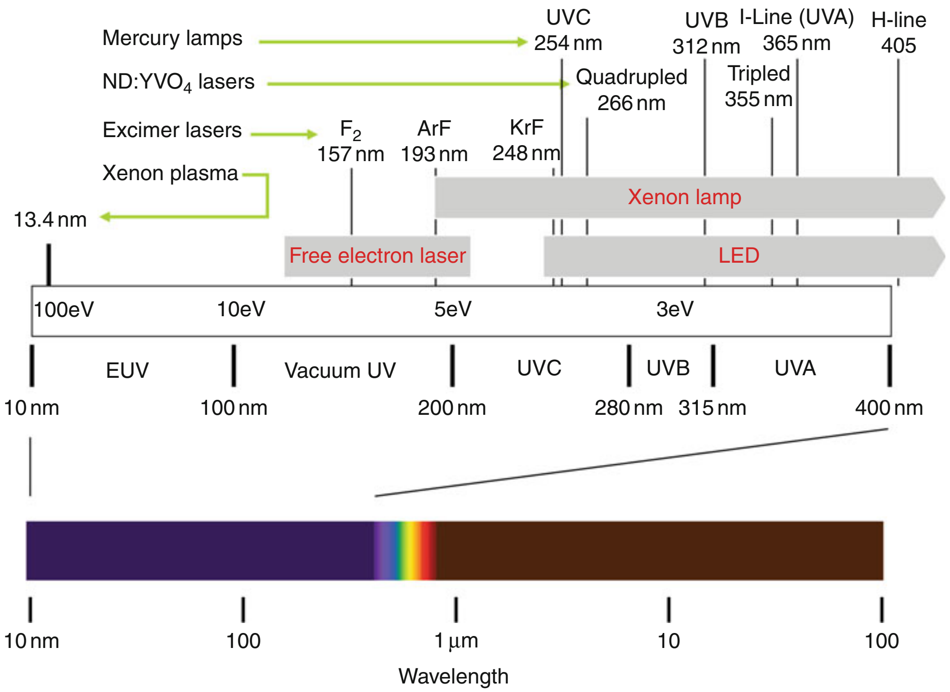


Fig. 10.17
Expanded view of the near-ultraviolet part of the spectrum

- UVC is the industrial and germicidal ultraviolet. Solar UVC is almost entirely blocked by ozone but large amounts of UVC are produced artificially, primarily by mercury lamps at emitting at 253.7 nm. This UV is used extensively for curing, sterilization and erasing EPROMS. Glasses largely block UVC so quartz or other materials are needed in optics for this band.
- Vacuum UV begins below 200 nm, where water vapour and oxygen begin to absorb ultraviolet strongly, relegating use of such wavelengths to evacuated systems. Absorption of similar magnitude continues down to about 10 nm by one or more atmospheric gases so this entire band is called vacuum UV.
- Extreme UV is the designation of the below-100 nm band to indicate that there are no transmissive optical materials useful below 100 nm.

Mirroring the infrared, as UV wavelengths move farther from the visible, fewer and fewer optical materials are available and managing the radiation becomes increasingly difficult and expensive.

10.3.3.2 Sensing UV Images

Fortunately, silicon can be made to work well as an ultraviolet detector across all of the UV bands. However, as the wavelength drops, so does the characteristic penetration distance of the radiation into the silicon before absorption occurs. Ultimately, the distance gets so short that surface effects on the silicon can draw a substantial proportion of the charge towards the surface instead of allowing it to travel inward to the collection point. The minimum distance, about 25 nm, occurs around the 254 nm wavelength. Treatments have been developed to provide an electric field in a favourable direction at the surface of the silicon to force the charge away from the surface where it is generated to a junction where it can be collected. These treatments can use very thin silver layers or chemisorption techniques but only work where the unoxidized surface of the silicon can be presented for treatment. As a result, they can generally only be applied to backside thinned scientific imagers.

Careful design of photodiodes can extend useful response below 200 nm for applications in spectroscopy but these design techniques are used primarily in line scan devices where the process constraints are generally more relaxed. Frontside area imagers with useful UV response below 350 nm are rare. In typical CCD imagers, the photosites are covered with polysilicon electrodes needed to move the charge. The polysilicon is thick enough to absorb essentially all radiation below 400 nm before it reaches the photodiode. In frontside CMOS imagers, the photodiodes are buried under multiple layers of silicon dioxide and plastics that severely attenuate the UV. Techniques have been devised to provide electrodes without covering the entire photosite or to replace polysilicon with less UV-absorptive materials to improve UV response. With these, the response can be pushed below 400 nm.

One time-tested technique for detecting UV radiation that cannot reach the silicon surface directly has been to coat the sensor with a transparent phosphor film that converts incoming UV into visible radiation that the sensor can readily detect. This technique, illustrated in ➔ *Fig. 10.18*, can extend the response of the typical sensor to below 200 nm. In general, however, these coatings will absorb some incident visible radiation, reducing overall quantum efficiency slightly and increasing response nonuniformity.

Because the reflectance of uncoated silicon can rise to more than 60% in the 200–300 nm range, the maximum possible external QE can be quite low. ➔ *Figure 10.19* illustrates the problem. Generally, the surface of the silicon is covered by silicon dioxide, but that helps little

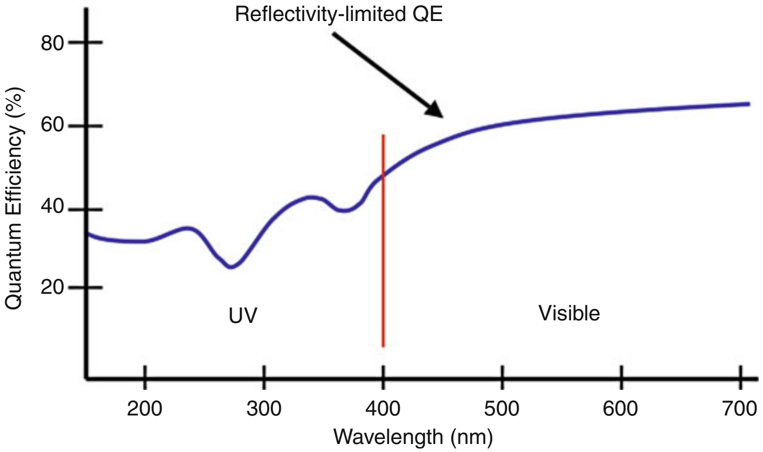


Fig. 10.18
Quantum efficiency of a silicon photo-sensor collecting charge all the way to the silicon surface as a function of wavelength

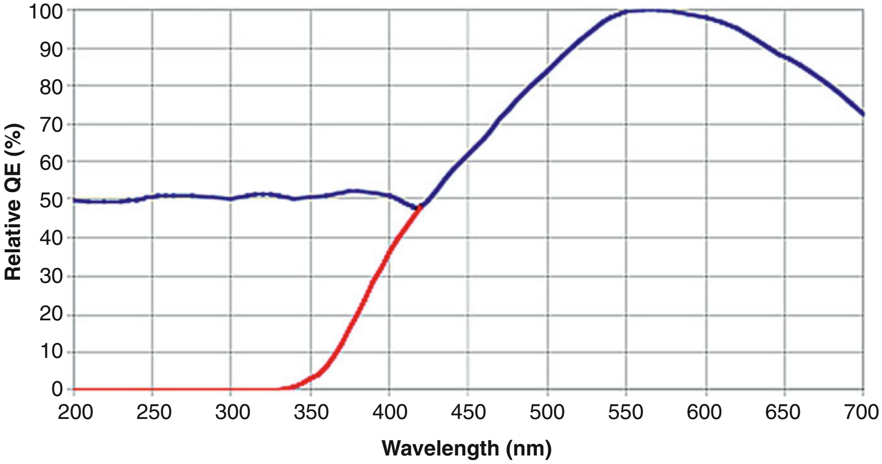


Fig. 10.19
Quantum efficiency of silicon sensors (which do not collect charge well at the surface) can be increased in the UV waveband by applying a UV-fluorescent phosphor (*Red curve*: uncoated silicon)

because it has an index of refraction too low to form a matching layer. Hafnium oxide (HfO_2) is a higher-index material that works well but is practical only in large area applications to backside-thinned sensors. With both field generation and antireflection applied, sensors with external QE in excess of 50% at 300 nm can be routinely accomplished.

Around 340 nm the photon energy equals the pair creation energy in silicon so that at shorter wavelengths two electron-hole pairs are generated and the internal QE exceeds 1. A second extra carrier pair is generated at 170 nm and an additional pair for each incremental 3.7 eV of photon energy. Since this extra charge does not improve the shot noise statistics, it can use up well capacity in sensors without delivering the S/N expected of strong signals. As a result, well capacity must be carefully considered for imaging in the VUV.

10.3.3.3 UV Optical Materials

Many transmissive materials used in visible optics will also transmit in the UVA band but exhibit a higher index of refraction. This displaces the focal plane forward in most commercial lenses, perhaps a tolerable accommodation. However, most currently manufactured commercial lenses do not transmit well in the UVA band because their antireflection coatings reject wavelengths below 400 nm quite strongly. This is intentional, intended to avoid fogging of film or bleaching of colour separation filters in still and video cameras. For best performance in the UV, optics must be carefully selected or specially built.

Lenses of many standard glasses can be used down to 300 nm if the AR coatings are designed for UV transmission. Below 300 nm, the choices become much more limited. The primary material used below 300 nm is fused silica, which has useful transmission to below 190 nm, which includes the ArF laser emission at 193 nm used extensively in semiconductor photolithography. Fluoride-doping can extend transmission of fused silica to nearly 150 nm. All of the other useful materials are crystalline and birefringent to varying degrees.

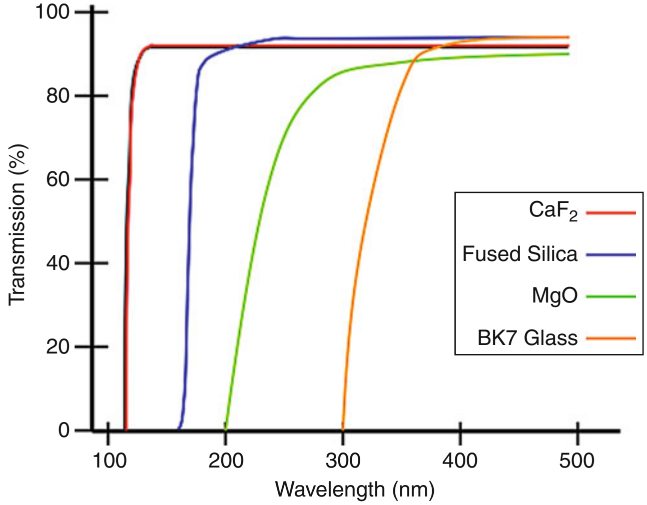


Fig. 10.20
Transmission through materials used in UV optical devices as a function of wavelength

Figure 10.20 shows some common materials that can be used below 200 nm. Other fluorides, chlorides and bromides are also useful. Since most of these are soft and hygroscopic, they must be handled very carefully and shaped using special techniques. Optical designs using these materials must include compensation for birefringence.

Reflective optics are also useful in the UV. Below 100 nm, they are mandatory, but even at longer wavelengths, they have a place. Two of the common reflecting materials used in the visible and infrared, gold and silver, perform poorly in the UV. Aluminium is the material of choice. Typical aluminized mirrors used for astronomy will reflect about 85% of UV light as long as they have not been coated with protective layers that absorb UV. Objectives for UV microscopy incorporating two aluminized surfaces have long been available. High-performance reflecting lenses can also be built in catadioptric designs with fused silica corrector plates but the lower wavelength limit on these is about 180 nm.

10.3.3.4 UV Imaging Applications

The most widespread applications for UV in industry are in the production and inspection of semiconductor masks and devices. Essentially all exposure of photolithographic resists used in defining patterns on semiconductor wafers use ultraviolet illumination. In the past, the 254 nm mercury arc lines was used, followed by KrF laser output at 248 nm for higher intensity. Currently, the vast majority of semiconductor production lines use 193 nm ArF laser sources. A substantial effort was made to implement systems using the 157 nm F_2 laser for finer geometries but the optical problems proved overwhelming. Even though the optical problems are much worse for even shorter wavelengths, it is still believed that a switch to the 13.4 nm wavelength from Xenon plasma must be accomplished to allow continuing reductions in device geometry. These same sources are also used for the production and inspection of the exposure masks and for inspection of fabricated wafers for proper geometries.

More generally, UV is used to image materials that fluoresce. Of course, in such applications the UV serves only as an illumination source and the actual images are made using the fluorescent output, usually in the visible, so no special optics are needed except perhaps a UV blocking filter. Images of UV emissions are valuable primarily in monitoring of dangerous situations such as the presence of hydrogen flames and the occurrence of electrical corona discharge from high-voltage transmission lines. Special-purpose cameras that provide UV images of the hazards and overlaid visual images for context have been developed for these applications.

10.3.4 X-rays and Gamma Rays

At the extreme short wavelength end of the spectrum lie the overlapping X-ray and gamma-ray bands. (➋ Chap. 12) Both x-ray and gamma rays are electromagnetic radiation but x-rays are generally produced by electrons falling to lower energy levels and gamma rays are produced by events in atomic nuclei. Perhaps the most important practical difference between these two types of radiation is that x-ray sources can generally be turned off with a power switch but gamma ray sources, radioactive nuclei, can't. X-rays cover the range of wavelengths from just below the extreme ultraviolet, about 10 nm, where the photon energy is about 100 electron Volts (eV) down to about 10 femtometers (100 MeV). This is a much wider range of wavelengths and energies than is covered by any other band. The gamma band extends from about 500 keV to beyond 500 GeV, also about six orders of magnitude. Radiation of these energies is generally referred to by the photon energy rather than wavelength, a practice that is followed here. However, in some circumstances, especially those involving diffraction imaging, the wavelength designation is more appropriate. The x-ray band is sometimes further split into two bands, with the lower energies designated as "soft" and the higher energies as "hard". The boundary is roughly 10 keV, corresponding to the highest energy photons that will penetrate only the soft tissues in the body.

Because of the extremely broad range of energies in these bands, a wide variety of techniques and materials must be employed to enable imaging. At the lowest energies, direct detection is possible but as the energies increase and the radiation becomes more penetrating, more complex assemblies of converters are required to stop enough radiation to create usable images in reasonable times. It is important to always keep in mind that the noise statistics of x-ray images, like any other images formed by photons, are limited by the number of photons initially absorbed. This is not confusing when the ratio of electrons produced to photons absorbed is less than one, as in visible imaging, but in x-ray imaging this ratio is often greater than one, perhaps up to several hundred. These large signals are the equivalent of adding gain, but they do not have statistics better than the original absorbed photons and, in many cases, actually have worse statistics because multiplication can be a noisy process. Thus, it is essential to assure that the photon flux is sufficient to provide the statistics needed to support the measurement required in each application. The simple amplitude of the detector output signal is not sufficient to assure that this requirement is met.

10.3.4.1 X-ray Absorption

To detect x-rays requires that arriving x-ray photons interact with some material capable of initiating a process that ends up with charge being stored in an image sensor. In all cases, detection requires absorption of incoming x-ray photons, a process that is strongly dependent on the atomic number (Z) of the material. X-ray absorption curves are generally expressed in terms of μ/ρ, the mass attenuation coefficient. This form is used to allow the curves of various materials to be compared on the basis of the mass per unit area of material intercepting the x-ray beam, a formulation convenient for both detecting and shielding calculations. If the density of the material is known, then a useful plot can be made of the attenuation length of the material. This is the distance an incoming beam will travel in a material as its intensity is reduced to 1/*e*, 37%, of the incoming intensity. The two examples of attenuation lengths, for carbon and lead, shown in *Fig. 10.21* convey the wide distance ranges involved.

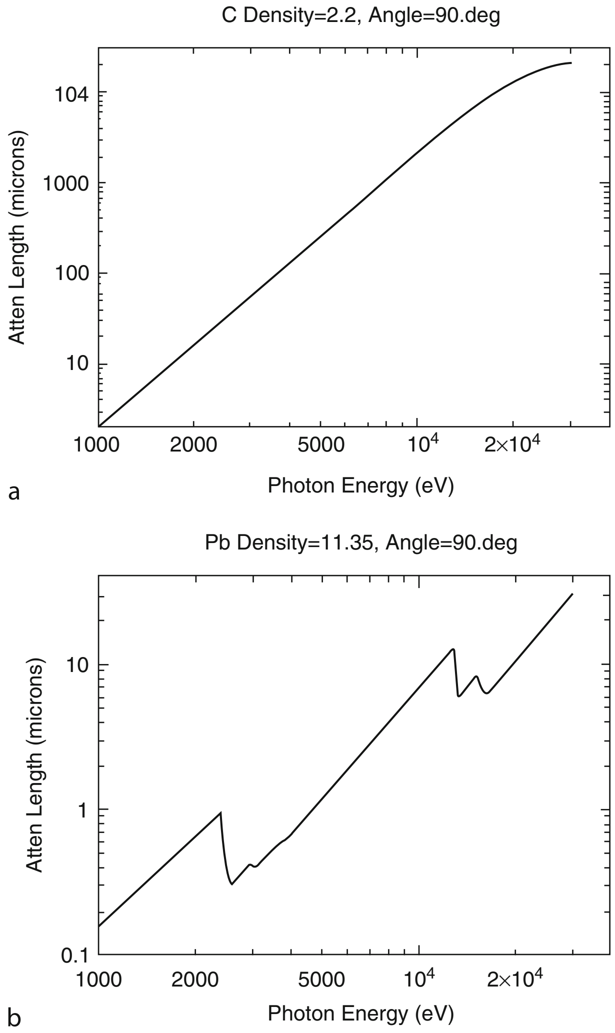


Fig. 10.21
Attenuation of x-rays plotted against photon energy (a) Copper (b) Lead. [3]

In order to produce an x-ray image, enough x-rays need to penetrate the material under examination for collection by an image converter or sensor. Generating and filtering an x-ray beam with the right mix of energies to produce sufficient contrast in the image while minimizing the dose absorbed by the material under examination is an art and science unto itself. A clear understanding of the materials to be examined and the methods by which x-rays are absorbed and scattered is essential to production of good image quality. Five interactions between x-rays and materials are possible as described below.

The Photoelectric Effect

When the energy of an x-ray photon is transferred to an atom, it dissociates by emitting an electron with a kinetic energy equal to the excess of the photon energy over the binding energy of the electron. The probability of this occurring is highest if the incoming photon energy is only slightly larger than the binding energy. This leads to the characteristic sawtooth shape of x-ray absorption curves as seen in ➲ *Fig. 10.22*, the curve for cesium iodide, a common x-ray to visible light converter. The absorption falls as the photon energy approaches the binding energy of electrons in a particular atomic shell and then rises rapidly at the binding energy only to slowly fall again. The fine-structured peaks occur where many several electron binding energy levels exist close together. The highest binding energy (resulting from electrons pulled from the innermost K shell) for all elements is 142 keV for Fermium 254 but most elements commonly used in x-ray imaging have K transitions in the range of 55–70 keV. The iodine K transitions

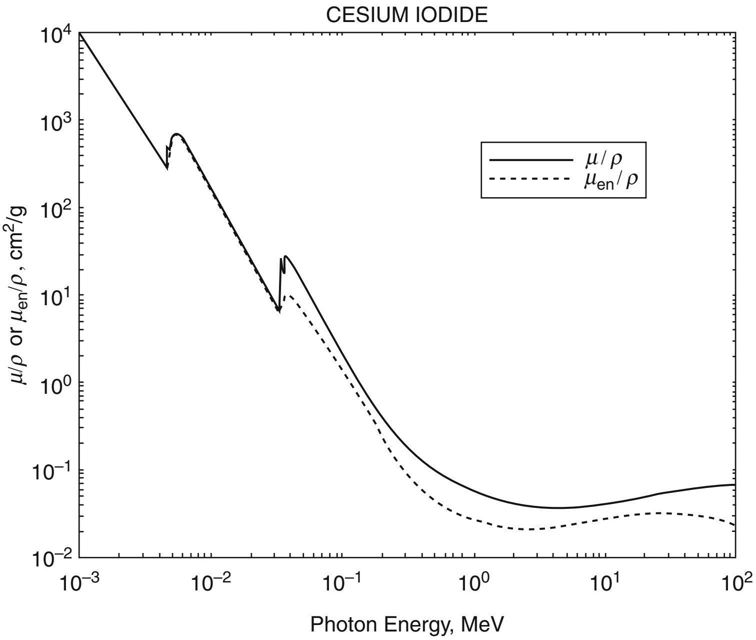


Fig. 10.22
x-ray Absorption curves for cesium iodide showing the characteristic sawtooth shape (See ➲ Chap. 12 for more examples) [4]

around 33 keV and the cesium K transitions around 35 keV are visible on the solid curve in ➲ *Fig. 10.21* as are the large number of L shell transitions in both elements around 3–5 keV. When x-ray photons knock electrons from their shells, other electrons fall into these, emitting secondary x-rays at the characteristic energy and generally these will be reabsorbed before leaving the material. The dashed curve represents the net effect of the reabsorption. The photoelectric effect is dominant in all materials at lower energies but provides a rapidly diminishing contribution to the total absorption above the K shell range.

Compton Scattering
In which the x-ray photon is not completely absorbed by an atom but glances off it at an angle, thereby transferring some of its energy to an ejected electron. After deflection, the x-ray photon energy is reduced by the amount of energy transferred to the ejected electron, which depends only on the deflection angle, not the type of material or the incident photon energy. The deflected photons are absorbed more readily than the incoming photons so most will not escape the scattering material. Compton scattering dominates in the part of the absorption curve where the slope transitions from steeply falling to flattening out where its contribution roughly equals the contribution of the next component.

Pair Production
X-ray photons with an energy of at least 1.022 MeV can produce electron-positron pairs if they strike the atomic nucleus. The positron is very short-lived and soon annihilates with an electron, producing two 511 keV secondary photons, which may then interact in other ways with the material or escape. Pair production can cause difficulties in imaging systems because the secondary photons can emerge from the material in any direction, causing a general background fog. Pair production dominates with x-ray energies above about 10 MeV.

Coherent Scattering
In coherent (or Rayleigh or Thomson) scattering, the incident photon escapes the atom without giving up energy but may emerge in any direction. This phenomenon contributes only a small fraction of the absorption and is confined to the energies of the photoelectric effect. Like pair production, coherent scattering adds to the overall background haze.

Activation
At very high energies, photons absorbed by an atomic nucleus may provide sufficient energy to activate a decay event. Since decay events may not occur immediately, the result of activation is that the nucleus becomes radioactive. The energies required are generally above 15 MeV – copper, for instance, can be activated at 16 MeV – and even at those energies, the efficiency is very low due to the small cross-section of the nucleus compared to the diameter of atoms. However, once a material is activated by x-ray exposure, the resulting radioactivity cannot be neutralized so energies this high should be avoided in industrial applications.

10.3.4.2 X-ray Image Converters

X-rays may be converted to charge for collection by electronic means using several conversion methods. These fall broadly into categories of direct, in which the electrons generated by the photoelectric effect are collected directly and indirect, in which the x-rays are first converted to visible photons and then to charge. At megavolt energies, more complex methods using build-up layers are used to assure that a sufficient number of x-ray photons are collected to assure statistical significance while providing properly-sized signals for detection but only the simpler methods used at lower energies are described here.

Phosphor Screens

The same screens used for film intensification can also be viewed by a video camera. Light leaving the screen is captured by a lens and focused on an image sensor. Since the collection of the light by a lens is much less efficient that direct film exposure by a screen, higher x-ray doses are generally required for electronic imaging with this arrangement. Nonetheless, this rather simple screen-camera combination has been used for many years in industrial imaging where dose rate is not a concern. To maximize the light collection, very fast optics (i.e. with low f-numbers) are commonly used and intensified imaging devices may also be employed.

One additional material, cesium iodide (CsI), is used for imaging with a camera. CsI has the advantage that it can be grown in a columnar structure, called acicular, much like a group of soda straws. This structure keeps any light generated inside its original column, directing the light to the end. As a result, CsI screens can be much thicker than granular phosphor screens resulting in an increase in x-ray absorption efficiency without loss of resolution. CsI also has a higher density than the phosphor materials, further adding to its x-ray stopping power.

Direct Converters

The photoelectric effect releases electrons into the material absorbing x-rays. These electrons will reattach themselves to an atom in the material sooner or later. However, if an electric field is applied across the material, then many of the electrons can be forced to leave the material before they recombine. If those electrons can be collected, they can form an image of the received x-rays. The material most commonly used for this purpose is selenium (Se) because it is a reasonably good x-ray absorber, especially at lower energies and because it is sufficiently conductive to allow collection of the liberated electrons. Substantial work has gone into the preparation of selenium for x-ray imaging so the image quality it can produce is quite high. Several other materials, most notably mercuric iodide (HgI_2) and lead iodide (PbI) have higher x-ray absorption than Se but both of these are more difficult to prepare in the large-area film forms useful for imaging.

X-ray Image Intensifiers

For most real-time (fluoroscopic) applications, screens optically coupled to cameras will not work because the demands on the x-ray generator tubes would be too high. In these applications, image intensifier tubes are most often used. In these tubes, the x-rays enter a vacuum envelope through a thin aluminium window coated with a CsI scintillator having a photocathode on its back side. Incoming x-rays make the CsI emit light, which is captured by the photocathode that then emits electrons into the vacuum. These electrons are focused by an electrostatic lens on an output phosphor which emits green light suitable for viewing by a camera. The output image is typically several thousand times brighter than a directly irradiated phosphor screen would be. This allows operation with much lower x-ray flux but, of course, produces noisier images because of the smaller number of arriving x-ray photons. Image intensifiers have round input surfaces with diameters in the range of 150–400 mm. The output phosphors are most often 25 mm circles.

10.3.4.3 Coated Sensors

Image sensors can be devised to detect light and convert it to charge or to collect arriving charge directly. Both of these techniques are widely applied in the detection of x-rays due to the need to adapt to the wide variety of converter materials available. When phosphors are used to convert x-rays to light, the direct deposition of the phosphor on the sensor array is optional. A very small separation, such as that resulting from a plastic protective layer, may still allow the phosphor to be close enough to the array to preserve the resolution. However, direct converters must be in electrical contact with the array so the interfaces between the converter and the silicon and the possibility of contamination, poor bonding and other deleterious effects must be very carefully controlled.

In addition, placing the converter directly in front of the sensor almost always results in the direct exposure of the array to x-rays because the converters rarely absorb all of the incoming radiation. In some cases, this can be alleviated by using a fibre optic coupler between the phosphor and the sensor. Unfortunately, couplers are limited to use in low-energy exposures and relatively short coupling distances with areas exceeding a 125 mm image diagonal only in extremely expensive equipment. As a result, the lifetime of the sensor under the expected exposure needs to be considered. The typical maximum dose before significant effect for CCD and CMOS imagers is in the range of 30,000–70,000 kRad. CID imagers are more radiation resistant, working often beyond 1 MRad.

Silicon Image Sensors

Standard CCD and CMOS sensors can be phosphor-coated and used for x-ray detection since they have SiO_2 passivation layers on top to protect the silicon from the phosphor materials. Sensors without microlenses are preferred because the microlenses have little effect on collection efficiency when the light source is in contact with them and the binder in the phosphor may react with the lens plastics. Of course, since x-rays travel in divergent paths, the sensor must be larger than the object to be viewed. Tapered fibre optic bundles can increase the input area somewhat but in any case the coverage for a single sensor is not likely to be more than about 125 mm. In very expensive cameras intended for x-ray crystallography, four or nine reducers may be bonded together to increase the input area but this produces extremely expensive assemblies.

The coverage area can also be increased through the use of buttable arrays designed with photodiodes that go right to the edge of the silicon on one or more sides. One, two and even three side buttable arrays have been produced to permit assembly of large x-ray imagers. Four-side buttable arrays have also been produced but these must be mounted on a second silicon wafer to provide paths for power and signal connections. By designing the support wafers so they are also buttable, very large imagers, with uses confined mostly to nuclear physics experiments, have been built and put in operation. Stitched arrays can also be made up to the size of full silicon wafers. Arrays up to 200 mm square have been made in this fashion using whole 300 mm wafers. X-ray sensitive line scan arrays of any length can be made using the same assembly techniques common in the production of contact image sensors.

Amorphous Silicon Panels

For large-area, radiation-resistant imaging arrays, hydrogenated amorphous silicon (aSi:H) is often the most suitable material. Amorphous silicon is the material used to make the switching transistor arrays in LCD panels. The transistor array alone, with the addition only of metal storage capacitors, is sufficient to provide the backplane functions for direct converters like Se. In panels coated with Se, some additional protection is required because the applied voltage used to drive the charge to the capacitors is 5 kV, however, these problems have been long solved. Se panels are most useful for radiographic applications because of the need for a recharge cycle after exposure. Selenium is most useful at lower x-ray energies, up to about 30 keV, due to its low atomic number, but it is in common use with sources up to 100 keV or so.

To use phosphors with amorphous silicon panels requires the addition of photodiodes to convert the light to charge. These are more complex than the panels for direct converters but are also in regular production. Both common phosphor screens and acicular CsI are used with these photodiode panels. Since the CsI can be made quite thick, up to 500 μm, the absorption of x-rays up to 70 keV or so can be high. CsI and the other phosphor screens can be readily used, with reduced efficiency, to image with industrial x-ray sources of up to several hundred kV. With added build-up layers, these panels are also usable with megavoltage sources for medical portal imaging and inspection of very thick materials. Unlike panels for selenium, phosphor panels can be readily operated as fluoroscopic imagers.

Amorphous silicon panels are much more compact than either optically-coupled or intensifier systems and have zero geometric distortion. However, the prime advantage of amorphous silicon panels is their radiation resistance, which extends to doses beyond 5 MRad.

10.3.4.4 Other Radiation Imaging

While radiation generated by x-ray tubes or, in rare circumstances, by gamma radiation from radioactive sources dominate the applications for imaging with penetrating radiation, other radiation modes and particles are useful in special cases. In nuclear imaging, for instance, an object is examined using its own gamma emissions through the use of a radiation collimator and a detector array. The collimator essentially restricts the radiation emerging from the object to angles that line up with the collimator holes, producing a one-to-one correlation between the position of the radiating source and a point on the sensor plane. A single exposure of this kind can only produce two-dimensional data but multiple exposures from different angles allow mapping the distribution of emitted radiation over a volume. Since emitted gamma rays will always have an energy characteristic of the emitting atoms, filtering out background radiation and secondary radiation from the object is straightforward. The most common detector for large areas is a scintillating sodium iodide (NaI) crystal monitored by an array of photomultipliers. This combination works well because the rate of arrival of gamma rays at the detector is usually quite low, generally no more than a few thousand per second. Cadmium zinc telluride (CZT) is also used as a detector for smaller areas. Because the radiation levels are low, the time required to accumulate an image containing enough photons to define the areas of interest can be many minutes, even hours in some situations.

The only particle used regularly for imaging is the neutron. The neutron is useful primarily because the cross-section for its absorption is so different, element by element, than for x-rays. With x-ray, the cross-section is roughly proportional to atomic number but for neutrons, cross-section is determined by the much more complex nuclear potential, which varies substantially depending on the energy of the incoming neutron. The scattering cross-section is especially high for hydrogen and low for metals, making neutron radiography especially useful for examining organic materials inside metal casings. Neutrons are always detected as a result of the decay products (gamma rays and alpha particles primarily) resulting from their absorption into an atomic nucleus. This means that as neutrons are detected, the detection material wears out. At some point so much of the detector will have been converted to the element resulting from radioactive decay that the original neutron-stopping element will be too low in concentration to be useful. Materials with large cross-sections that also emit light include lithium, gadolinium, boron and a large number of both solid and liquid organic scintillators. The most commonly used detector in industrial applications is a layer of gadolinium oxysulfide used as a film intensifying screen, a convertor for silicon detectors or as the input phosphor in image intensifiers.

Proton radiography, after languishing for decades after its initial development in the early 1970s, has again raised interest. For protons, the advantages are that the deposited dose for image quality comparable to x-ray images is much smaller than the x-ray dose and that the absorption length of protons can be accurately tuned by changing the proton energy. Thus, lower-energy protons (<1 GeV) are suitable for examining soft tissue and other organics, while high energy protons (up to 50 GeV) produce high-quality images of thick metal objects. Of course, since protons are charged particles, they can be focused allowing small objects to be examined at high resolution. Proton imaging detectors use phosphor screens of CsI and rare-earth orthosilicates.

10.4 Sensor Performance Measures

The capability of an imaging system to provide the information needed to a computer or a viewer depends fundamentally on the quality of the data that the image sensor can produce. In some applications, reproduction of fine detail is paramount. In others, accurate detection of colour is critical, in still others, precise rendering of small changes in brightness. Success in most applications requires some minimal level in a variety of performance measures so that the cumulative effect of limitations in each measure still permits extraction of the necessary image features. Although the task of converting a flat optical image to an equivalent electronic representation seems straightforward, implementation of this function involves a large number of choices and limitations that should be made with careful regard to the application.

Unfortunately, there is no standard yet for the acquisition or presentation of sensor performance data. Much effort has been and continues to be expended on such standards development but the results, while useful, still fall far short of facilitating universal direct comparison of the performance of one sensor against another. As a result, those faced with selecting sensors must understand sensor performance in terms of individual factors as they relate to specific sensors to support calculations that can lead to satisfactory sensor selection.

10.4.1 Spatial Resolution

Essentially all image sensors now in use break the image plane up into discrete areas, generally approximating rectangles, often squares. As a result, the signal from the sensor is not continuous, but rather sampled. While sampling has only a small effect on the signal when intensity variations in the image occur only over many samples, the effect on the signal when the signal changes by large amount from one sampling location to the next can be substantial, producing results to do not resemble the original intensity variations. Understanding these effects requires a brief discussion of sampling theory.

If a pattern of intensity is projected on the sensor which is a series of parallel black and white lines of equal width and spacing as shown in ➲ *Fig. 10.23*, then the distance between the

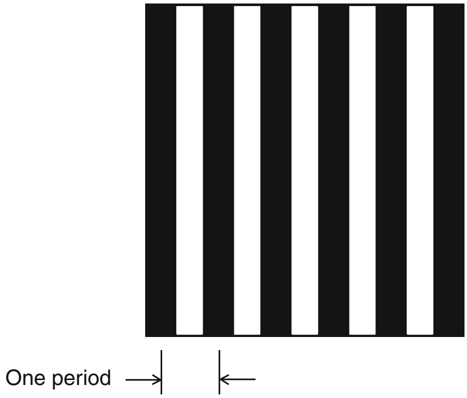


Fig. 10.23
Pattern consisting of parallel *black* and *white* lines of equal width and spacing

centre of one black line and the next (or one white line and the next) is called the *spatial period.* The combination of one black line and one white line is called a *line pair.* Comparing the distance across one line pair (equal to the spatial period) to some standard linear measure, typically 1 mm, produces the ratio of line pairs per millimeter, abbreviated lp/mm and called the *spatial frequency.* Using the same calculation, the maximum spatial frequency that can be detected by an image sensor is just the extent of two sampling locations in the sensor matrix compared to 1 mm. For instance, if the sampling locations are 10 μm apart, the two of them cover 20 μm or 0.02 mm and 50 of these pairs will fit in 1 mm. Thus, the *limiting resolution* of the sensor is 50 lp/mm.

10.4.1.1 Aliasing

In sampling theory, the spatial frequency corresponding to the limiting resolution is called the *Nyquist limit.* Any spatial frequency presented to the image sensor above the Nyquist limit will produce a signal, but it will be an *alias*, that is, a difference frequency between the spatial frequency of the illumination pattern and the Nyquist limit of the sensor. For spatial frequencies in the image just above the Nyquist limit, that alias will have a low spatial frequency that appears in the sampled data as if it were real information at that low frequency. Since the alias has a frequency between zero and the Nyquist limit, it overlays the true image information and cannot be removed by computation once it is generated. It is very important then to minimize the generation of aliases by restricting the spatial frequencies in the image to those below the Nyquist limit. Controlling lens resolution or adding spatial filters, called anti-aliasing (AA) filters, are typical methods of restricting alias generation.

10.4.1.2 CTF and MTF

When a black and white set of bars (a square-wave pattern) is imaged on the sensor, the amplitude of the signal generated will depend on the spatial frequency of the pattern. Plotting the signal amplitude as a function of the pattern spatial frequency produces a curve called the *contrast transfer function* or CTF. If the pattern image on the sensor retains full black and white at all spatial frequencies, then the shape of the CTF curve will be the sinc (sinx/x) function as shown in ➋ *Fig. 10.24*. The horizontal axis is the number of cycles (black/white line pairs) that is imaged on each sampling location. The one (full amplitude) occurs at low spatial frequencies where it is assured that each sample is either black or white so that the difference between these is 100%. The zero occurs when exactly one pair fits in each sample location because the signal from all samples will be the same, the average between black and white, so that the difference between the largest and smallest signals from the various samples will be zero. Between one and zero, the situation is more complicated.

Consider the situation when the pattern bars are exactly the same width as the sample spacing. As ➋ *Fig. 10.25* shows, there are many possible resulting signals, from a full-amplitude square wave to a flat 50% grey level, depending on the relative position of the bars and the sampling locations. Although the signal amplitude is a linear function of the relative phase, the calculated theoretical amplitude when the bars and sample spacing just match (0.5 cycles per sample on in ➋ *Fig. 10.24*) is about 63%, not 50%.

In practice, the plot of signal amplitude vs. spatial frequency is often called the *modulation transfer function* or MTF. True MTF measurements require that the test target bars be sinusoidal in transmission, not square wave. In most practical systems, the distinction is not significant because of the effect of the camera lens on a square wave target. The MTF of the lens is typically limiting compared to the theoretical CTF of a sampling device so MTF and CTF can be used interchangeably with little risk of significant error. In fact, in real systems it is important that the lens or some other optical element limit the MTF to minimize aliasing in the sampled image.

➋ *Figure 10.26* illustrates the situation when the lens is not limiting. In the figure, the entire curve to the first zero is shown. The top curve (blue solid and dashed) shows the natural curve

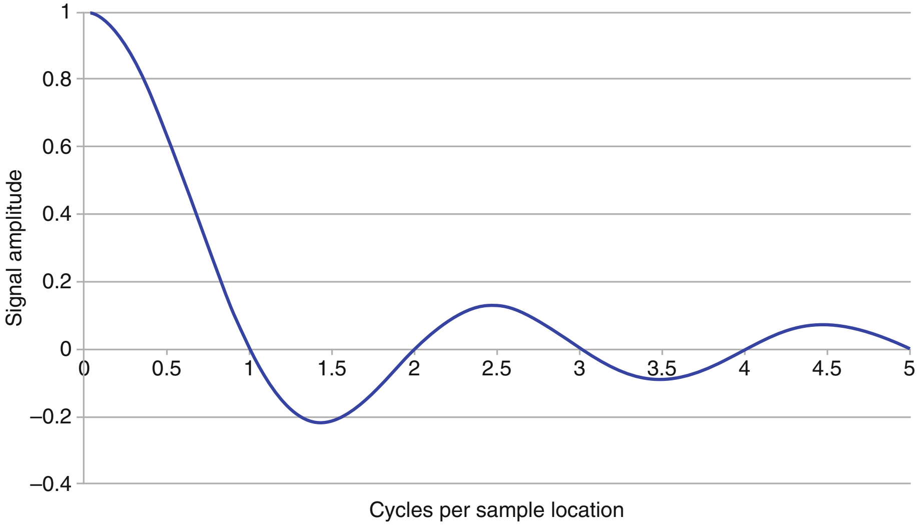
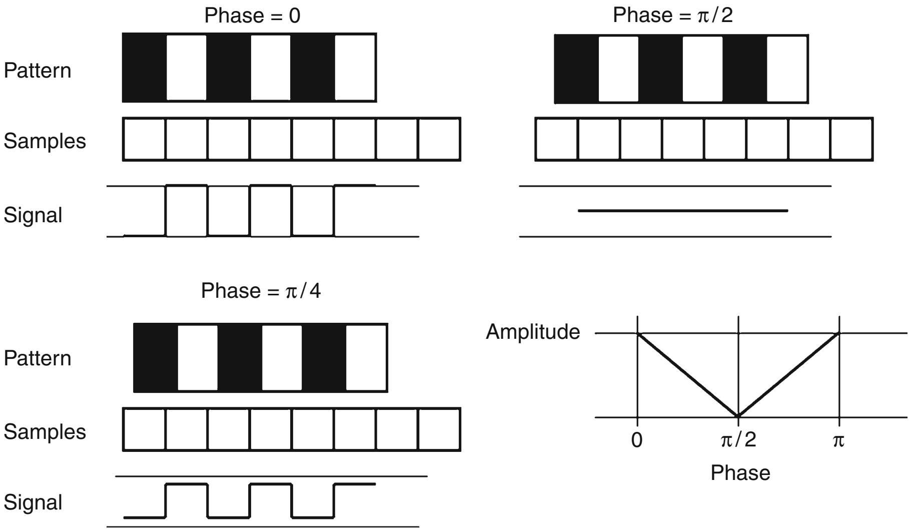
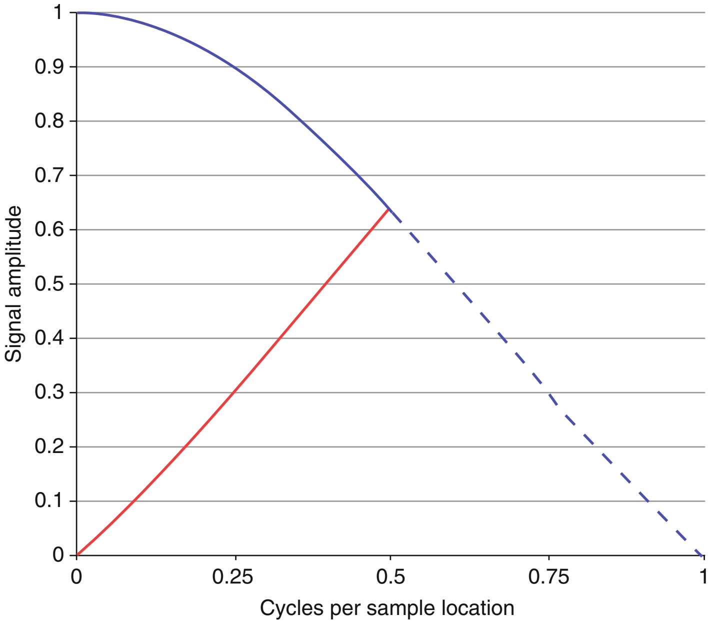


Fig. 10.24
The function sinc(X) (= sin(X)/X) for positive values of X



■ **Fig. 10.25**

Pattern bars of exactly the same width as the sample spacing. The resulting signal depends on the phase (i.e. relative position of the bars and the sampling locations)



■ **Fig. 10.26**

Aliasing when sampling a bar pattern. The solid blue curve indicates the amplitude when the sampling frequency is above the Nyquist limit. When under-sampling occurs, the curve is effectively "folded" about a vertical line (cycles per sample = 0.5); the dashed blue maps into *red curve*)

and red curve shows how the alias from bars with spatial frequencies above the sampling cutoff fold back into the real frequencies of the sampled signal. If the bars are permitted to arrive at the sensor with full amplitudes beyond the sampling cutoff, the amount of aliased signal is a significant portion of the signal as a whole. The image signal becomes even more confused when frequencies above twice the cutoff (1 sample per cycle) are allowed through because the aliases generated there are negative, that is, reversed in phase, from the bars that generate them.

10.4.1.3 Aperture Effects

The curves in previous section apply only when the sampling areas are contiguous, that is, when the fill factor in an image sensor is 100%. When the fill factor is lower or the aperture is smaller than the sample interval, the aliasing situation gets worse. Essentially, the x-axis scales with the reduction in fill factor. So, for instance, if the fill factor is 50%, the bar width equals the aperture at the 0.25 point on the spatial frequency axis. This increases the amplitude response to about 83% at cutoff but the curve will fold three times before reaching the first zero rather than once. Controlling the spatial frequencies reaching the sensor becomes even more important when the fill factor of the sensor is low.

10.4.1.4 Resolution Nomenclature

Resolution can be expressed using several absolute and relative terms. The choice depends on the application and situation. The absolute term typically used is line pairs per millimeter (lp/mm) in the object plane. Measuring resolution in this way allows direct relation of the measured resolution to feature sizes in object to be viewed. The most widely used test chart intended for object-plane use is the US Air Force 1951 Resolving Power Test Target shown in ➲ *Fig. 10.27*. The original target has square-wave bars (white on black) at various spacings. The spacings change by a factor of two for each group (large numbers) and by the sixth root of two for each element in the group. Group 0, element 1 is 1.0 lp/mm. The original chart covered 0.25–645 lp/mm but larger, lower resolution charts have been made for use in outdoor settings.

Resolution may also be expressed in lp/mm either on the image plane or on the object plane (field-of-view). Since resolution is more often useful when related to field of view, it is common to see an expression of resolution relative to an image dimension. Originally, the universal expression was TV lines per picture height (TVL/PH), now often abbreviated just TV Lines (TVL). Since this is a relative measurement, the test charts can be any size and the lines can be any width as long as reference marks on the chart are fitted to the height of the image. A typical chart of this type, the EIA 1956 Resolution Chart, is shown in ➲ *Fig. 10.28*. This type of chart has now also been adopted for characterization of digital still cameras where the resolution is expressed as Lines per Height (LPH). These figures can be converted easily to lp/mm at the image plane by dividing the resolution number by twice the height of the sensor in mm.

10.4.2 QE & Sensitivity

The purpose of an image sensor is to convert incoming photons into an electrical signal. The first step in this process is conversion of photons into charge. This is generally accomplished by

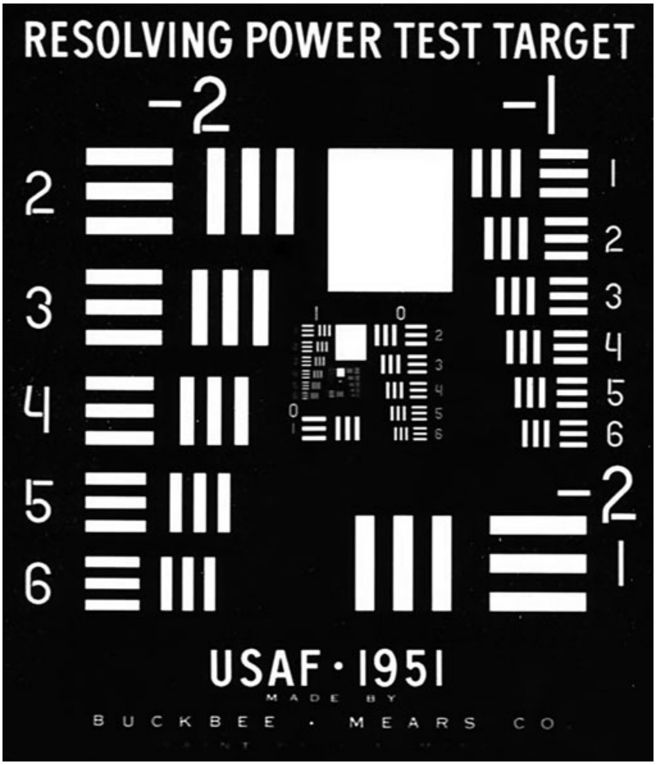


Fig. 10.27
US Air Force 1951 resolving power test target [5]

capture of photons in a photodiode in which electrons or holes are liberated in proportion to the arriving photons and collected for readout. The ratio of the number of collected electrons (or holes) to the number of incoming photons is called the *quantum efficiency* (QE). If a sensor can convert every incoming photon into one collected charge, then the QE is 100%. Unfortunately, some photons are lost because they are reflected from the surface of the sensor or absorbed outside the photodiode or cancelled by effects caused by the wave nature of light. Some charge is also lost, by recombination before it can be collected or by moving to an isolated area under the influence of electric fields or even by arriving at the collection point too late to be counted. The net result is that the QE is always less than 100%.

Due to variations in the photon absorptance of the sensor material and the performance of various layers on top of the photodiode with photon energy, QE is usually expressed as a function of wavelength. Each detector material has an underlying characteristic QE curve that is modified by the thickness of the device, the characteristics of overlayers and various enhancement processes that may be applied. Using silicon as an example, *Fig. 10.29* shows the typical internal QE of silicon in its top curve. This is the QE that would be measured if all of the incoming photons could be delivered to the photodiode in a place where they could generate collectable charge. This curve is essentially the absorbance of a 1 mm thick layer of silicon.

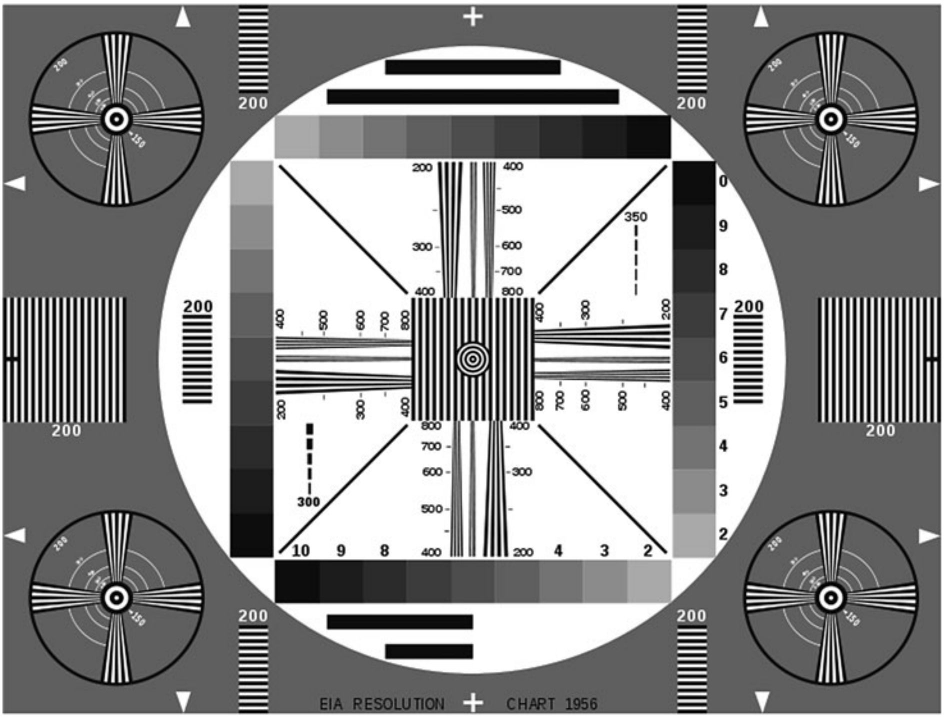


Fig. 10.28
EIA 1956 resolution chart [6]

In a typical sensor, many factors intervene to reduce the QE. *Figure 10.29* shows typical effects in an interline CCD sensor but similar effects are present in other sensors as well. Losses occur due to the reflectance of silicon. In devices with thin enough top oxide to permit use of an anti-reflectance coating, ripples in the QE curve will generally be present as a result of optical interference in the various overcoat layers. The QE is further reduced by loss of photons that strike the aluminium shields and conductors on the sensor. This is the effect of fill factor. Some charge may be lost by conduction to structures intended to provide isolation or to prevent overloads. Finally, the spectral extent at shorter wavelengths is typically reduced by absorption in the overlayers and recombination of charge generated too close to the surface to be collected. At the long wavelength end, the limited thickness of the photodiode will prevent a larger and larger proportion of the photons from being absorbed as silicon becomes increasingly transparent with wavelength. Of course, each of these problems suggests it own solution and many of these have been implemented in sensors where higher QE is required.

Some sensors report QE numbers greater than 100%. Two ways this can occur are by detection of very energetic photons or provision of charge gain on the sensor. Energetic photons, those with more than about 3.6 eV of energy in silicon, will generate more than one electron per photon due to ionization. Charge gain can be provided through the use of impact ionization shift registers. However, although these techniques can increase the signal amplitude, they do not improve the statistical noise in the signal itself so even in such devices converting every photon into collectable charge remains important.

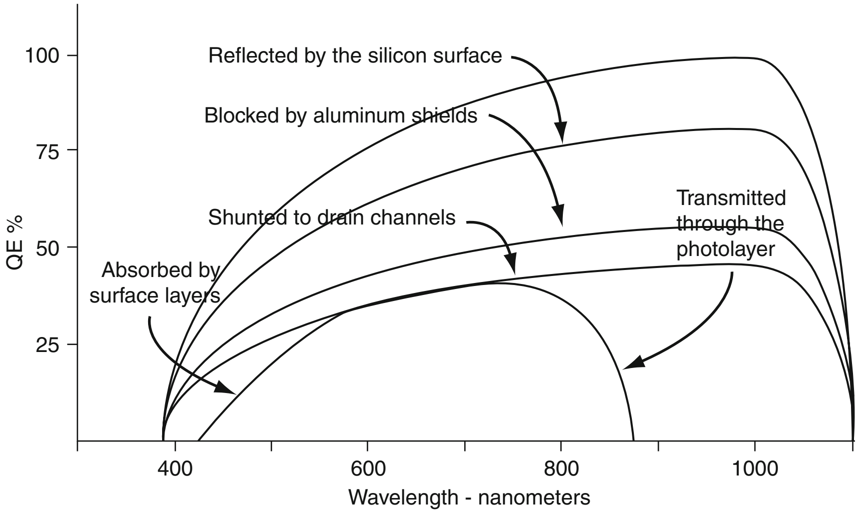


Fig. 10.29

Quantum efficiency of silicon (*top curve*). This is the QE that would be measured if all of the incoming photons could generate collectable charge in a photodiode. Other curves show the performance of various layers above the photodiode

The effectiveness of the photon to charge conversion may also be expressed by *sensitivity*, the ratio of the generated photocurrent to the incoming optical energy expressed in Amperes per Watt (A/W). Equivalently, this measure may be expressed statically in Coulombs per Joule (C/J). The significant difference between the QE expression and the sensitivity expression is that in sensitivity the photons are each weighted according to the energy they carry using the expression e=hc/λ. This means that if the QE is the same at two different wavelengths, the sensitivity will be higher at the longer wavelength. Equivalently, if the sensitivity is equivalent at two different wavelengths, the quantum efficiency will lower at the longer wavelength.

10.4.3 SNR and Dynamic Range

Signal-to-noise ratio (SNR) and dynamic range (DR) are related but must be carefully distinguished to be used and compared properly. SNR is a spot measurement, that is, it is taken with a specific signal level while dynamic range is a multi-point measurement that has meaning only when taken as the ratio of signals at two different levels. To see the difference the makeup of the output signal of a sensor must be understood at different signal levels. For simplicity, a sensor that responds linearly to input photons from zero to some maximum (the saturation exposure) is assumed.

When the input photon level is zero, the sensor signal will consist solely of the sum of noise components either generated within the sensor, such as reset or transistor noise or shot noise from dark current, or received from the outside, such as clock, conducted power supply or radio frequency noise. This total is the dark noise as shown by the line labelled "Sensor noise" in ❯ *Fig. 10.30*. Since the signal from the photon input is zero, the SNR is also zero. Note that, strictly speaking, the SNR with no signal cannot be expressed in decibels. As the exposure rises,

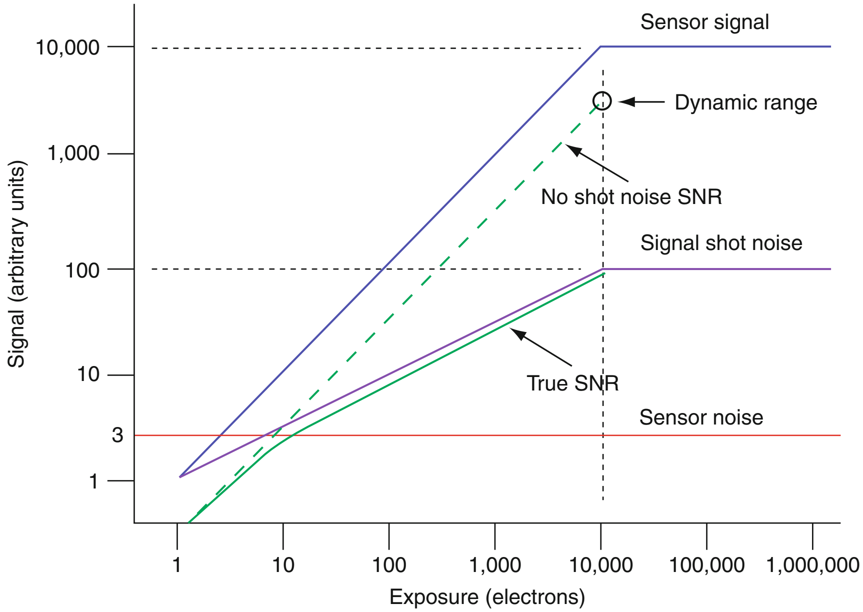


Fig. 10.30
Noise and signal components which taken together define the sensor output

the signal rises with it until the maximum signal handling capacity (usually sensor full well) is reached.

It is often assumed that the SNR of a sensor is just the signal divided by the sensor noise. This is unfortunately not true, since the signal consists of discrete packets, photons or electrons. This characteristic gives rise to shot noise, a statistical variation in the actual number of photons or electrons captured when some average exposure is measured. Since photon arrival follows a Poisson distribution, the shot noise is simply the square root of the signal . That is, if the average number of photons arriving at the sensor is 10,000, then the standard deviation in the arrival number is 100. Thus, at exposures above the square of the sensor noise, shot noise predominates. Below this point, the SNR curve parallels the exposure curve while above this point, it parallels the square root of the exposure curve as does the shot noise itself.

Shot noise is often the limiting factor in detecting low contrast objects. Generally, two signal amplitudes must be separated by at least three standard deviations of the noise to be reliably discriminated. If a signal 90% of maximum is generated using the sensor of *Fig. 10.30*, then the signal is 9,000 electrons and the standard deviation is 95 electrons. Three times this is 285 and 9,000/285 is about 32. This means that that maximum number of separate levels that can be reliably discriminated with this sensor is only 32. This implies that if 256 levels were needed (8 × 32) the maximum signal would need to be at least 8 × 8 × 9,000 or 576,000 electrons. Of course, these signals and the noise are measured on a per sample basis so detection can be readily improved by averaging over a number of samples either spatially or in time.

Dynamic range is a much simpler calculation because the denominator in the calculation has no signal and therefore no signal shot noise. In the sensor in *Fig. 10.30*, the maximum signal is 10,000 and the noise is 3. The DR is 10,000/3 or 3,333. This is much higher than the

SNR but the utility of a signal equal to the noise at the low end, where the SNR is only 1, must be considered in light of the application. A requirement for ten times the noise is not uncommon, reducing the usable DR to more like 300:1.

10.4.4 Speed

Just as the number of sampling elements in an imager determines its spatial resolution, the frame rate of the imager determines its temporal resolution. Frame rate is important whether a single object is to be viewed multiple times for monitoring of changes of individual objects are imaged only once as in high speed industrial inspection. Calculating frame rate is rarely as simple as it seems. In the simplest case, it is only necessary to multiply the number of sample elements in an image times the time per sample period and add any dead time in the scan cycle to determine the frame period. The frame rate is then the inverse of the frame period. However, complications arise even in the simplest situations so the details of the scan cycle must be knows to assure a propoer calculation.

10.4.4.1 Line Scan Rates

In the earliest line sensors, the photodiodes were active all the time and no charge integration took place in the sensor. Readout simply consisted of sequentially connecting to each of the photodiodes and reading the instantaneous current. Although, this configuration requires a lot of illumination, readout is very simple and the speed is determined simply by how long each photodiode must be interrogated to produce an integrated signal with the necessary S/N. In addition, the exposure of the photodiodes in imagers of this type is not simultaneous. Each pixel is sampled in a different time interval, so moving objects appear as a series of tilted lines. Whether this is significant or not depends on the application.

When the integration function was moved to the sensor, the timing rules became more complicated. The important distinction between the two types of line scan imagers is the capability for storing an integrated signal in a location separate from the photodiodes so that a new image acquisition can be started during the readout of the previous image. Those without separate readout will produce tilted lines similar to those from the sampled photodiode scanners except that the exposure periods are overlapped rather than sequential. In this case, the line scan interval is determined solely by the time required to read out all of the storage locations unless the exposure time is set longer than the minimum scan time. In that case, the readout time will be almost the same as the exposure time. To avoid tilted lines, a strobe may be used which illuminates all the photodiodes simultaneously and then permits scanning in the dark. The scan time is then the strobe time plus the calculated time to read out all of the photodiodes.

In sensors with storage structures separate from the photodiodes, the integrated photodiode signals are all transferred at once into the storage structure and then the contents of the storage structure are read out while the next exposure takes place. This arrangement eliminates tilted lines even with continuous illumination. The scan time is still some minimum set by the number of photodiodes and the clock rate that will be increased if the exposure used is longer than the minimum.

In all cases, repetitive scanning will guarantee the shortest periods. Individual triggered scans often add setup time to the cycle. Attention to determining all elements required to complete a timing cycle for a particular sensor or camera are critical to understanding the actual speed that may be attained.

10.4.4.2 Area Scan Rates

Fundamentally, an area scan is just a sequence of line scans with some time added between lines and some time added before or after the complete scan. As with line imagers, some imagers are still arrays of photodiodes without integration. Similarly high illumination levels are required and images of moving objects suffer similar tilting. Most imagers use one of the schemes described in earlier sections so appropriate precautions should be taken to assure that the image content is as expected. As with line scanners, the scan period in area scanners has some minimum that may be increased by long exposures or by the necessity to operate in an individually triggered mode. Full knowledge of these effects is similarly important.

Many area scanners offer options for region of interest (ROI) scanning or binning to provide tradeoffs of field of view or resolution for speed. Rarely is there a direct inverse relationship between the reduction in elements scanned and speed because of the fixed times that have to be added to lines and frames regardless of the ROI or binning settings. In some sensors, these added times are even longer in the reduced area or resolution modes than in the full modes so, again, a complete understanding of the timing requirements of each needed mode is essential to avoid erroneous assumptions about the efficacy of the tradeoffs.

10.4.4.3 Temporal Limitations

While complete isolation between sequential frames is desirable, some possibilities for interframe correlations need to be considered. The most common interframe effect is *lag* or image persistence. While the amount of one image left behind to become part of the next is small in modern imagers, it is not necessarily zero. In some imagers, it is still on the order of 1% so if large variations in brightness are expected from one frame to the next great care should be taken, perhaps even by scanning dark frames, to avoid leftover signals. Lag can be especially severe in some imagers if the sensor is allowed to saturate because saturation can drive charge into areas not used for normal imaging. Upsets caused by this stray charge can persist long after one frame. If the charge is forced into traps (undesirable energy levels in the bandgap of semiconductors), typically by very large multiples of the saturation level, it can emerge seconds or even hours later.

Interframe problems are not limited to simple retention of image signal. Charge in the wrong places can change the operation of transistors in the signal chain, modulating gain or offsets or even affecting noise performance. In thinned sensors, bright spots can cause localized heating, raising dark current. In some sensors, even moderately oversaturated spots can interfere with charge transfer in later frames or pull down power buses. Many of these effects are small but only testing under real conditions can assure that they are insignificant in a particular situation.

10.4.4.4 The Effect of Taps

Taps are additional outputs on image sensors intended to permit increases in readout speed by providing parallel readout of sections of the sensors. In a simple case, the sensor might be split into left and right halves with each half provided with an output. To a first approximation, this will double the readout speed. Three taps triple the speed, four taps quadruple it and so forth. However, suppose that a sensor with left-right taps is to scan an ROI of half the sensor height and width centred on the sensor as in ➋ *Fig. 10.31*. It is clear that the speed can be roughly doubled by reading out only those lines in the ROI. However, along the rows, there is no gain in speed because the scanning of the taps runs in the same sequence on all taps. That is, columns 1 and 5 must be read at the same time. So too, 2 and 6, and so forth. As a result, all columns must be accessed at the normal clock rate, the same situation that would occur if no ROI had been designated in the horizontal direction. Only if the ROI is completely contained within one tap (and is smaller than the number of columns serviced by the tap) is any speed improvement realized. Of course, if the taps can be separately addressed, a very rare feature, then a speed advantage may be available.

10.4.5 Causes of Degradation

The possible ways that the transformation from an optical pattern of photons arriving at the sensor into an electrical signal can be adversely affected number in the hundreds. Fortunately, most of these factors are subject to control or improvement by good engineering. A few are not – shot noise, for instance – but those which are should be carefully considered lest they degrade the signal sufficiently to render it useless. What is important is to understand where in the build-up of a system each of these factors can be influenced.

Some may occur inside the sensor and be uncorrectable outside it. Others might also occur inside the sensor but be correctable by later processing. Usually, the later in the chain that a negative effect can be corrected the better because the resources required to implement correction are generally more accessible and flexible downstream. For example, reducing

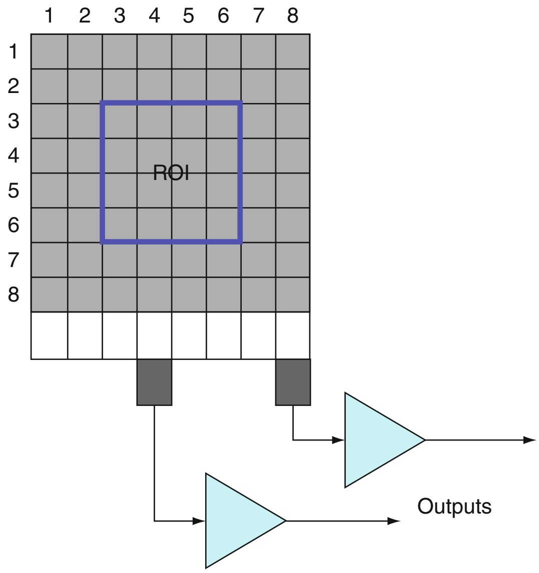


Fig. 10.31
Tapping the sensor array to sample a region of interest (ROI)

noise in a sensor requires modifications of design in silicon and perhaps even process development. Reducing noise for visual display proposes can be implemented in post-processing software.

Some effects are irreversible or, at best, only approximately reversible. Reversing such effects often demands large amounts of processing power and very close matching of the reversing algorithms to the original effects to avoid generation of artefacts that can be more troublesome than the original effect. Better to avoid as many of these effects as possible.

10.4.5.1 Optical Interactions

Interactions between the sensor and its optical environment fall in the category of difficult to reverse without sever artefact generation. In a simple case, lens vignetting can be reversed by careful application of variable gain across the image. Unfortunately, varying the gain also varies the noise, introducing noise gradients in the image that are often harder to deal with than the original shading. Better to just use an optical system designed for low vignetting. Other interactions may be both more difficult to correct and easier to avoid.

Oversize Image

With the large variety of sensor sizes and lens image diameters available matching these is not always possible. An image diameter smaller than the sensor causes an obvious problem – dark edges on the image – but image diameters much larger than the sensor can lead to a different type of nonuniformity. In most cameras, there are reflective surfaces around the active area of the sensor, some, like bond wires, inside the sensor itself and others, like solder pads on the sensor pc board, in the camera. Care must be taken to assure that light does not reflect from these structures and ultimately end up as scattered illumination in the image. It is generally good practice to surround the active part of the sensor with an optically-absorbing material minimize the possibility of scatter.

When large-format lenses, like those for 35 mm film cameras, are used on adapters – c-mount being the most common – additional optical baffling may be needed to assure that the substantial amount of light that can arrive outside the sensor active area is absorbed. One further precaution needed with such lenses is careful monitoring of bright illumination that may be present outside the sensor field of view. A bright light out at the edge of the lens acceptance angle can cause substantial glare in the image even though the source may not be imaged by the sensor.

Interference Patterns

As mentioned above, the various overlayers on a sensor can act as interference filters and produce ripples in the QE as a function of wavelength. In applications where absolute illumination levels must be measured, these ripples can make reliable measurements with narrow band sources difficult due to variations in the ripple positions from sensor to sensor and even in various positions on a single sensor. Variation in the interference patterns across the sensors can also generate significant non-uniformity when narrow-band illumination is used. Interference effects can be very strong at longer wavelengths where, as silicon becomes more transparent, reflections from the back of the silicon die interfere with incoming light. The effect is to produce broad light and dark contours in the images that map variations in the thickness of the die.

Backside Reflection

Also at longer wavelengths, the entire image can be reflected back through the die and detected a second time. Such a reflected image will be out of focus and generally slightly larger than the incoming image. This effect can be avoided by filtering out longer wavelength light before it reaches the sensor. If a particular wavelength that falls in a low QE part of the response curve is essential to the desired image, watch carefully for spreading of small bright spots due to backside reflection.

Crosstalk

In sensors with smaller pixels, the pixel spacing begins to fall inside the typical absorption length for visible photons in silicon. This can allow photons arriving at large angles from the normal to pass through the first pixel area where they arrive only to be absorbed in a neighbouring pixel. Microlenses can ameliorate this effect somewhat by forcing more photons to arrive near the normal. Users can control the effect by selecting lenses with longer back focal distances to minimize the larger angle rays in the arriving image.

Angular Dependency

The high index of refraction of silicon makes all the QE of all sensors somewhat angle sensitive but this is a minor effect and can be minimized through the use of lenses that cover an image area slightly larger than the sensor. Microlenses make the angle dependence worse because they become less and less effective in directing incoming light to the photodiode as the angle of the light deviates further and further from the normal. In many cases, the angular limitation is different in the horizontal and vertical directions because the photodiodes are not square. While the typical acceptance angle (where 50% of the incoming rays hit the photodiode) is about 20°, half on each side of the normal, the range for various sensors can be 10–30° even reaching these extremes on the same sensor for the two axes.

In an attempt to accommodate lenses with short back focal distances, some sensors now have radially offset microlenses, aligned in the centre of the imager and progressively more offset toward the centre as the radial distance of the pixel increases. Such sensors can produce mild vignetting when used with lenses having large back focal distances because of the loss of rays near the normal in outer pixels.

10.4.5.2 Charge Spreading

After photons have been converted to electrons the charge must be controlled to assure that each carrier remains associated with the photodiode where its generating photon was absorbed. Charge spreading through overflow, called *blooming*, can be minimized through the use of photodiode isolation techniques and provisions for rapid removal of excess charge. A variety of structures have been developed for CDD an CMOS imagers to manage blooming so this has becomes less of a problem. Often now, sensor blooming is not at fault when bright objects appear larger than they are because the real problem is reflections in the optics or one of the optical interactions described above.

Even without overloads, charge diffusion in the sensor can put charge carriers in the wrong location. The likelihood of diffusion is very small in most sensors because of the isolation techniques now almost universally employed. However, diffusion remains a consideration in those sensor types which use the bulk silicon for charge storage.

10.4.5.3 Spurious Signals

The large number of potential spurious signals fall into two rough categories, those that affect the offset of the signal by raising or lowering it level independent of the signal amplitude and those that modulate the signal, effectively changing the gain on a pixel by pixels basis.

Offset Noise

Overall variation in offset is typically stated as Dark Signal Non-Uniformity (DSNU). Here are some typical types of offset noise and their effects:

- Reset noise – As discussed previously, this is an uncertainty in the reset level of the sample node. It can raise or lower the signal from its true value and changes with each new sample.
- Circuit noise – this can come from any of the components in the sensor signal path. It has the same effect as reset noise but is usually much smaller in amplitude.
- Telegraph noise – This is due to individual charge carriers moving in and out of traps in the semiconductor bandgap. It is especially significant in sensors with small well capacities or at very low signal levels. The offset can be positive or negative depending on the direction of the charge jump and shows up as transient whiter or darker points in the image.
- Dark current – this is charge accumulation in photodiodes due to reverse leakage current. The major effect is a unidirectional offset in the signals of all pixels. Dark current has shot noise, however, which causes pixel to pixel variations in the dark offsets.
- Column noise – Found only in CMOS imagers, column noise is caused by variations in the threshold voltages in the column storage transistors. Since the signal from all rows in a column pass through the same transistor, column noise appears as vertical stripes. Column noise does not typically vary from frame to frame so it can usually be removed through subtraction of a dark image which contains the column noise signal. Temperature changes can sometimes upset the accuracy of this subtraction. In the extreme, columns can appear completely white or black. Column noise generally makes up the bulk of the measurement called Fixed Pattern Noise (FPN) but other static sources may contribute.
- Power supply noise – This is noise generated in power supplies and conducted to the sensor through internal power buses. The appearance depends on the nature of the noise. Low-frequency noise from feedthrough of AC power can show up as horizontal bars. Square waves from switching supplies can appear as periodic bright or dark points.
- Thermal noise is indistinguishable from similar noise generated in the sensor and can be hard to identify.
- Clock noise – Clock noise that is not synchronous with the scan timing can appear very similar to switching power supply noise. Synchronous clock noise will generally have either the same effect on every sample or inverse effects on alternate samples. The polarity of the offset depends on the details of clock phasing. While clock noise may be considered benign if it affects every sample identically, any instability in the clock timing or amplitude will directly affect the sample offset so precautions should be taken to minimize the presence of clock signals in the image data at the time of sampling. If the image data is not digitized but rather displayed as an analogue signal, clock noise can appear as very strong vertical lines in the image.

Modulation Noise

Overall variation in response is typically stated as Photo Response Non-Uniformity (PRNU). Here are some typical types of modulation noise and their effects:

- Geometry variations – Image sensors are produced by semiconductor manufacturing processes which overall are quite uniform but still contain variations that can affect the QE of individual photodiodes. These variations can take place in essentially any process step resulting in variations in the sizes of the photodiodes, the depth of the junctions, the shape of microlenses, the spectral characteristics of colour filters or in a large variety of misalignments. In the extreme cases, photodiodes may be completely inactive, appearing as dark spots, or shorted, appearing as white spots.
- Transistor variations – Primarily affecting CMOS sensors, variations in the gain of the various transistors in the signal chain can affect the signal amplitudes generated by the charge packets from individual photodiodes. If these differences result from localized problems like particles only small numbers of pixels may be affected. Variations across a sensor can also be present. With small sensors, these variations are more likely to be seen as gain differences from one sensor to the next.
- Temperature changes – The absorption of silicon to various wavelengths decreases with increasing temperature. As a result, the gain of an entire sensor may be noticeably affected by temperature shifts. This shift can be large enough to produce chromaticity errors in colour cameras.
- Nonlinearity – This is a secondary effect caused by the effective change in gain at different signal levels in sensor with nonlinear transfer curves. The effects of nonlinearity can be difficult to sort out because noise sources introduced before the nonlinear element will have amplitudes that are dependent on the signal level while noise sources introduced farther along will be signal amplitude independent. Nonlinearity can make cancellation of other noise effects quite difficult. Correcting the nonlinearity with processing will approximately reverse the effects of the early vs. later noise sources.

Other noise problems can arise from the digitization process. These are considered below.

10.4.5.4 Reciprocity Failure

Contrary to expectations, image sensors do not necessarily respond to a total integrated flux independent of the integration time. That is, a very short exposure from, say, a laser, will not necessarily produce the same signal as the same total flux from a continuous source collected over a longer time. Generally, the shorter pulses will produce smaller signals due to local exhaustion of carriers, and the inability of generated carriers to get out of each other's way. The minimum pulse width at which this effect begins to occur varies widely among sensors. In addition, some sensor may not produce the same signal level even with continuous illumination as the exposure time and illumination intensity are varied inversely.

10.4.6 X-ray Camera Performance

X-ray cameras have one additional performance parameter to consider. Since the quality of x-ray images is often defined by their S/N due to the limitations in available x-ray photons, a measure of the degradation of the S/N by x-ray imaging systems called Detective Quantum Efficiency (DQE) has been devised. DQE is the ratio of the S/N of the signal from the x-ray imager divided by the S/N of the incoming x-ray beam (just the shot noise in the x-ray flux). DQE is generally expressed as a percentage so the maximum DQE is 100% when the imaging device adds no noise. Since the noise spectrum of the x-ray beam and the imaging device are generally not flat, DQE can also be plotted as a function of spatial frequency. Note that DQE can be calculated for any part of an imaging system beginning with the image detector. Electronic conversion is not even needed so the DQE of x-ray image intensifiers with optical outputs can be calculated. DQE typically varies with x-ray dose rate and x-ray energy spectrum for any imaging device so full characterization can require a family of curves.

10.5 Colour Imaging

As image processors become increasingly fast and powerful, one of the increasingly complex tasks they are asked to undertake is detection and measurement of colour. Colour has been a small part of machine vision for many years, applied to sort fruit or detect wire colours, but as the quality of the colour camera equipment has improved and the prices have come down, the tasks now can include colour matching and control that were formerly possible only with off-line instruments. While early colour applications could be successfully implemented with just a little common sense and experimentation the newer applications require some familiarity with the formalities and terminology of colour science and some basis on which to select appropriate camera equipment. Herewith, a brief introduction.

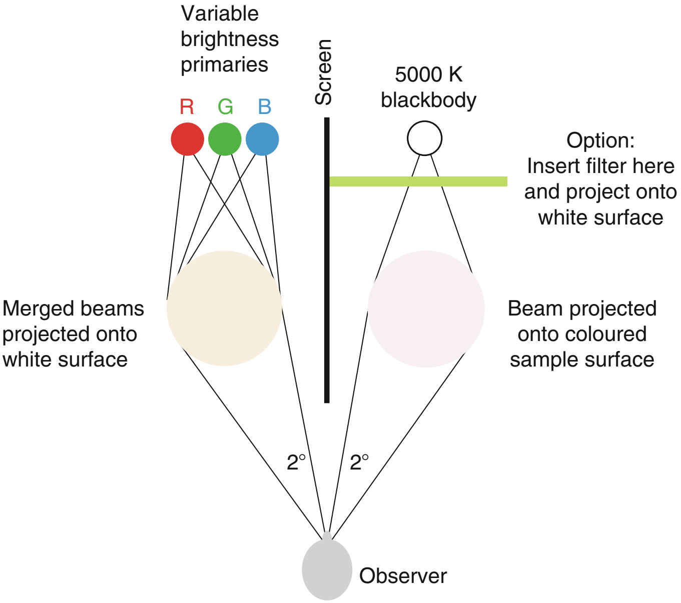
10.5.1 Requirements

The fundamental requirement for production of colour images, that is, images that can be presented in such a way that the eye perceives them as having colour characteristics reasonably approximating the colours in the original, is that at least three spectrally-separated samples of the original must be taken. The reason for this requirement can be seen in ➜ *Fig. 10.31*. In the left diagram is shown a sample task, sorting oranges. If a detector is set up that measures the amount of red light reflected from the object, then it can distinguish from unripe oranges – which reflect only green – and ripe oranges, which reflect both red and green. The red signal can be represented as a grey scale, which can be used to automatically select ripe oranges. Unfortunately, this single colour method cannot distinguish ripe oranges from dead ones, which are yellow. To distinguish orange from yellow requires measurement of green reflected light so this simple task already requires two detectors. Similarly for the task in the illustration on the right, two detectors are required. If we want to measure the brightness of blue emission from an LED, we need one detector sensitive in the blue range. However, this detector tells us nothing about the emission out of the blue range so another detector, which measures everything except blue, is also needed. With these two detectors, we can assure that the ratio of blue to not-blue is suitable.

Now, let us assume that we want one system to perform both tasks. For the oranges, we need red and green detectors, and for the LEDs, we need blue and not-blue detectors. This seems like four detectors, however, with a little experimentation, we might notice that the sum of the red and green detector signals tracks well the signal from the not-blue detector and that the blue detector signal doesn't change much as we look at the three classes of oranges oranges. This could lead us to believe that three detectors are both necessary and sufficient to define colours. We would turn out to be right but with a few important constraints. After all, what about purple?

10.5.2 Colour Matching

Early on, it was decided that the useful thing to know about colour would be some rigorous description of colour as seen by human eyes that could be used to produce any colour at any time with some assurance that the various colour samples produced would all look the same. In the 1920s several experiments were set up as shown in ➋ *Fig. 10.32* to determine a colour matching function. An observer was positioned in a dark room so that a colour sample subtending a 2° circle, while illuminated by a 5,000° Kelvin incandescent source, could be viewed on the left side of a screen. The viewer was given control of a projector pointed at the right of the screen containing three monochromatic sources at 435.8, 546.1 and 700 nm that illuminated a 2° circle with amounts of light from the sources set by three knobs. Various colour discs were inserted on the left and the viewer was asked to set the knobs so that the projected colour matched the colour sample.


Variable brightness primaries
R
G
B
Screen
5000 K blackbody
Option: Insert filter here and project onto white surface
Merged beams projected onto white surface
Beam projected onto coloured sample surface
2°
2°
Observer

◘ Fig. 10.32
Colour matching is conducted a dark room. The surface whose colour is to be analysed is illuminated by light from a standard 5,000 K blackbody source. (An incandescent filament lamp will normally suffice). The observer controls the intensities of three monochromatic sources, wavelengths 435.8 (B), 546.1 (G) and 700 nm (R). Their beams are merged, by projecting them onto a white surface. The observer adjusts the brightness of each of these three primaries, until he/she judges that the colour of the merged beam matches that of the sample surface

Almost immediately, it was discovered that some of the matches were impossible. In order to match some of the colour samples, a second projector had to be added to the left side so that the viewer could project monochromatic light from one or more of the sources on the sample while those same sources were set to zero on the right. By simple algebra, this meant that to avoid using projectors on the left, projectors that would subtract light from the right were needed. It had turned out that the colour matching function had negative parts, as shown in ➲ *Fig. 10.33*. That finding confirmed that it is not possible to duplicate the appearance of all spectral colours with three monochromatic sources. Since the concept of negative light is inconvenient, some other solution for matching needed to be found.

10.5.3 Tristimulus Values

The solution came again from algebra. It was decided that if three spectral primaries would not work, perhaps a different set of primaries might. To make new colour matching curves that can be combined linearly to form the measured curves, it is only necessary to combine the experimental curves algebraically to produce a set of three that are all positive. Since this process has more degrees of freedom than are necessary, one of the curves was preassigned the shape of another key curve, that which plots the luminous efficacy of the eye. This is the plot of the apparent brightness of equal-energy spectral colours. This was designated the $\overline{y}(\lambda)$curve and the other two computed curves were designated $\overline{x}(\lambda)$ and $\overline{z}(\lambda)$. These curves are shown in ➲ *Fig. 10.34*. Note that the negative parts are gone. With this set of curves, however, the primaries are imaginary, that is, they are physically unrealizable. What they form is a colour space that is larger than that required to produce all physical colours, thus defining a space that provides positive numbers describing every possible colour.

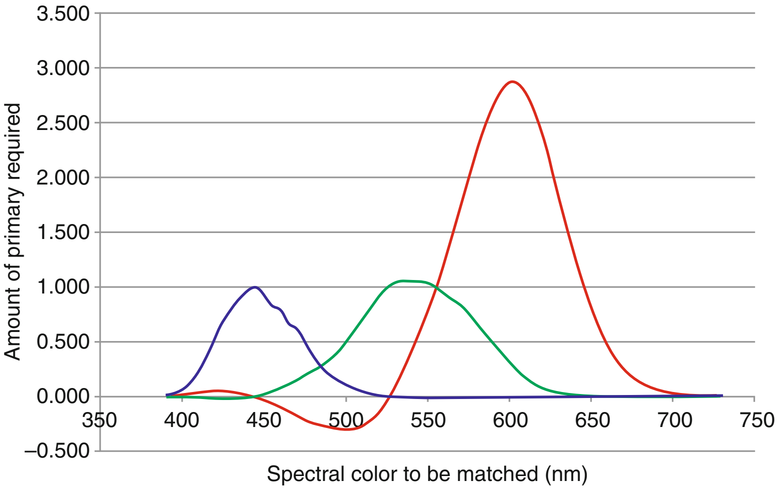


Fig. 10.33

Results of colour matching. The fact that negative contributions are needed from the three primaries is explained in the text

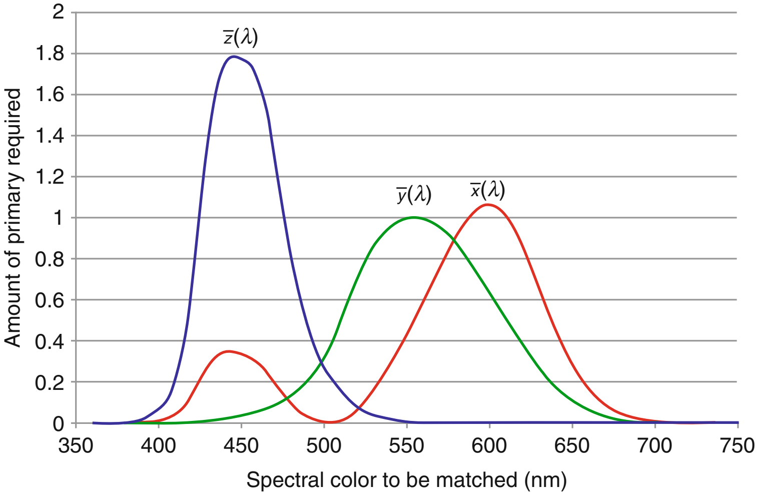


Fig. 10.34
Imaginary primaries (i.e. not physically realizable) can avoid negative spectral values

It is essential when using these curves to understand that they have two separate but equally valid meanings. Assume that it is desired to produce the perception of 500 nm spectral illumination. To do this, the colour matching curves require that 3 units each of the "blue" primary and the "green" primary are required. Conversely, if we had detectors with spectral response curves shaped like these colour matching curves, then applying a 500 nm spectral illumination to them would produce equal signals in the "blue" and "green" channels and no signal in the "red" channel. To extend this principle, if we have a source that is a mixture of spectral colours (which, of course, is what any colour is), then the signal it produces will just be the sum of the amount of each of the spectral colours multiplied by the value of the curves at each colour. This leads to the generalization in which we take the integrals over all wavelengths of the products of the applied spectral distribution and the three colour matching curves. The results of these integrals are designated *X, Y* and *Z* and called the tristimulus values.

10.5.4 Chromaticity Values

One further simplification was implemented. Since brightening or darkening a colour in the XYZ space simply means that all three values are made proportionally larger or smaller, it is possible to remove the brightness variable simply by normalizing the curves. To do this, each of the three curves is divided by the sum of X, Y and Z to produce a new set of curves designated x, y, z. The sum of x, y, and z is always 1 in this calculation so only two numbers are needed to specify a colour (independent of brightness). These are the chromaticity values and can be plotted in a plane to generate the chromaticity diagram shown in *Fig. 10.35*. This plane shows only the colour or chromaticity. The brightness is indicated by the Y value, which is the integral of the product of the luminous efficacy curve and the spectral power distribution of the colour being measured. This combination is called the CIE *xyY* colour space.

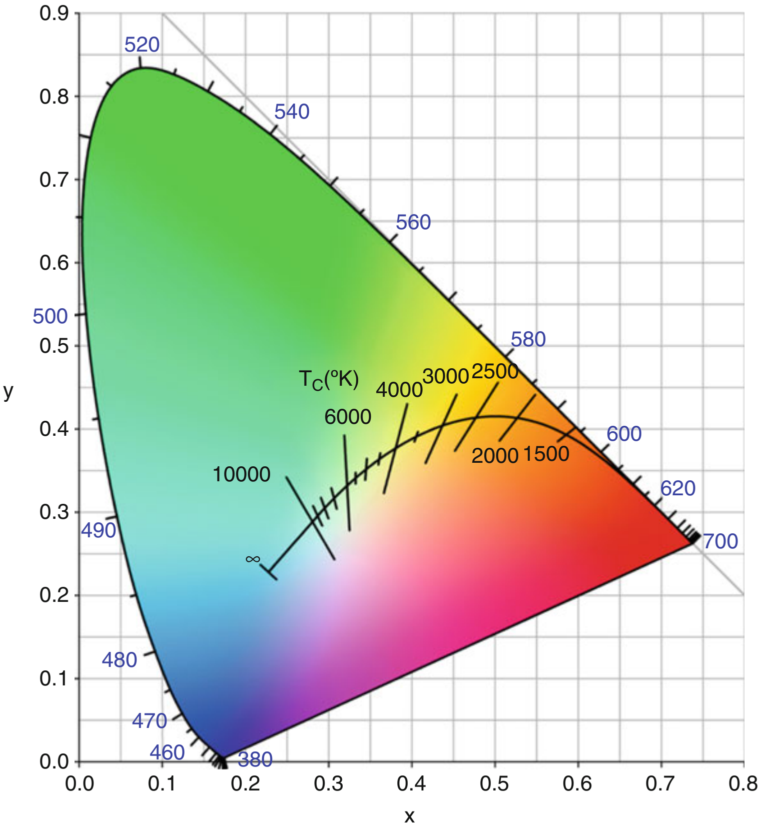


Fig. 10.35

CIE chromaticity diagram. The position of the peak in the spectrum of a blackbody radiator varies with its temperature (See black arc.)

Several important points about the chromaticity chart need to be understood.

- Imaginary primaries – The chromaticity chart shows all visible colours within the central horseshoe shape. The black space around the horseshoe indicates the rest of the space created by the imaginary primaries at $x = 1$ and $y = 1$.
- Spectral locus – The curved portion of the boundary of the horseshoe is the spectral locus. It represents all of the spectrally pure visible colours. The numbers on this line indicate the spectral wavelengths. Part of this locus falls along the line connecting the primaries because these colours have no z content. This can be seen from the tristimulus curves.
- Magenta line – The straight line connecting red and blue shows the non-spectral mixtures of these two real colours.
- White point – Since the three curves have been normalized to sum to 1, the equal energy point, called the white point is at $x = y = z = 0.3333$.
- Blackbody locus – The black line with the numbers indicates the visual colour of a blackbody radiator at the temperature indicated by the number.
- Colour differentiation – moving from any point on the chart to any other point requires changing two of the coordinates; if only x or y are changed, then z will also change. This means

that if a colour change is to be visible, the change must result in the change of two chromaticity values. At the red end of the horseshoe the chart indicates 650 nm but the eye is sensitive out to about 780 nm. However, beyond 650 nm, the x and y curves are nearly proportional. This is equivalent to having a change in only one value so the change in wavelength from 650 to 780 nm will result only in a decrease in brightness, not a change in colour.

The actual visible colours cover a much wider range than can be displayed or printed so the chart must be taken only as a rough representation. Violet at 400 nm, for instance, is missing completely because it cannot be displayed or printed using standard phosphors, dyes or inks.

10.5.5 Colour Spaces

A colour space is a numerical model with a specified mapping, usually involving three or four values, which provides convenient methods of describing or transmitting colour information in specific applications or environments. A common example is the sRGB colour space, used in almost all computers, that maps tristimulus values to the colours capable of reproduction by the phosphors in CRT displays. The numerical representation for sRGB is the familiar 0–255 scale for each of the three RGB channels. Generally, one colour space may be converted to another through the use of a set of simultaneous equations. (➋ Chap. 4) This process may, however, produce negative values, indicating that a colour that can be represented in one colour space is out of range for another.

10.5.5.1 Colour Space Conversion

Sets of simultaneous equations for colour space conversion are often expressed in matrix form. Where the conversions are linear, as in the conversion of XYZ values to sRGB, the matrix is of the simple 3 × 3 element type as shown in ➋ *Fig. 10.36*. Clearly, this can produce both negative values and values greater than 1 for the sRGB coordinates. In practice, those values are discarded because they cannot be reproduced by typical displays. The sRGB colour space can be shown on the chromaticity chart by finding the extremes of x and y that produce sRGB values between zero and one. These will map a triangle, as shown in ➋ *Fig. 10.36*, which has vertices that correspond to the primaries used to define the sRGB colour space. Only those colours inside the triangle can be displayed on a device using the sRGB colour space but in practice, the real primaries on a display may not match the colour space primaries and so the displayable colours may be further restricted. Colour standards also generally specify a white point, the colour generated when the three values in the colour space are equal. For sRGB, this is D65, approximately the colour of a 6,500 k blackbody.

10.5.5.2 Colour Spaces

A wide variety of colour spaces have been developed to provide convenience in different applications. Selection of a proper colour space depends on understanding the various ways in which each colour space represents what the eye can see and on evaluating the suitability of each colour space to simplify colour image data in a way that reduces the computation complexity of the required machine vision tasks.

$$\begin{bmatrix} R,_{RGB} \\ G,_{RGB} \\ B,_{RGB} \end{bmatrix} = \begin{bmatrix} 3.2410 & -1.5374 & -0.4986 \\ -0.9692 & 1.8760 & 0.0416 \\ 0.0556 & -0.2040 & 1.0570 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

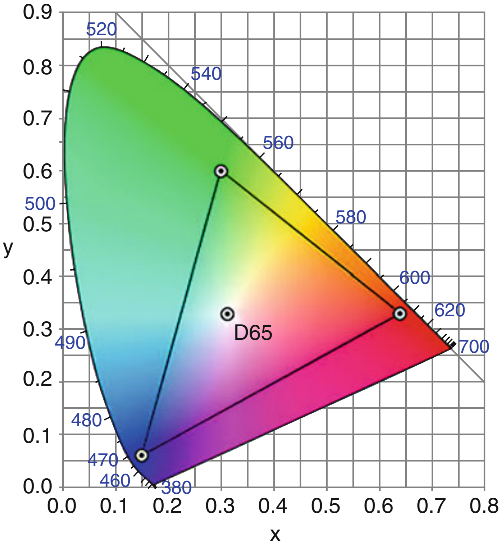


Fig. 10.36

Converting from one colour space to another. (a) Equations for mapping XYZ colour coordinates into sRGB parameters. (b) CIE chromaticity diagram The vertices of the triangle represent three primary light sources. Only those colours within that triangle can be approximated by mixing light from those primaries. This is called the gamut for that particular set of colour parameters

RGB Spaces

All RGB spaces are designed to support a direct representation of the data sent to the three colour sources in a display. (➜ Chap. 4) Originally, these data were three analogue voltages but now they are almost always digital numbers. In the typical RGB space, zero indicates that a particular colour source is turned off and 255 (in 8-bit systems) indicates that it is all the way on. Although the original RGB space was defined based on real CRT phosphors, newer RGB spaces have corners in the triangle that are far enough apart to accommodate a variety of different phosphor sets, LCD filter dyes or DMD colour wheels [7, 8]. In addition, RGB spaces may include gamma correction that may or may not match the characteristics of a particular display. Since sending the same data to these different devices will result in different displayed colours, ICC profiles should be used wherever colour matching among displays is critical. sRGB, Adobe RGB, RIMM, ROMM, xvYCC and others are spaces of the RGB type with different gamuts. (The *colour gamut* is a certain subset of colours. The most common usage refers to the set of colours that can be accurately represented in a given circumstance, for example within a given colour space or by a certain output device.)

Other spaces with structure similar to RGB are:

- $L^*a^*b^*$ – This space uses a highly nonlinear set of matrix equations to create colour charts that are more perceptually uniform than the XYZ space. This means that moving equal numerical distances in different parts of the colour space produces the appearance of equivalent change to the eye. This type of representation conserves data because it does not pack more values into one part of perceptual space than into another.

- CMYK – This space is used to specify colours for print. C, M and Y are cyan, magenta and yellow, the subtractive primaries. K is black, which is substituted for equal amount of CMY in dark areas to prevent putting down too much ink. Because CMYK actually has a relatively small gamut, much high-quality printing now uses six or seven inks to expand beyond the space that CMY can cover.
- YUV – This is one of several colour spaces that use special colour axes rather than *x* and *y* to provide immunity from various types of misadjustment in the colour rendering chain. YPrPb and YCbCr are newer examples.

Chromaticity/Brightness Spaces

These colour spaces separate the chromaticity of the colour signal from its brightness. Whereas RGB spaces may be thought of in terms of Cartesian coordinates, these spaces use a polar coordinate model. Typically, the angle around the cylinder will represent the chromaticity, the radius will represent how much the pure colour is mixed with white and the height will represent brightness. The most common space of this type used in machine vision is HSI for Hue, Saturation and Intensity. Hue is the colour angle, saturation is the distance from the axis (white on axis and pure colours at the outer boundary) and intensity is the brightness running from black at the bottom of the cylinder to full brightness at the top. This sort of space has the advantage that the base colour can be measured without regard to brightness so uniform illumination is not essential. HLS, HSL, HSB and HSV are similar spaces.

10.5.5.3 Colour Gamut

Each colour device has a range of colours over which it can operate. This range is termed the gamut. Printers, displays and most other output devices have colour gamuts that cover perhaps the central 30% of the chromaticity chart; they are not capable of reproducing saturated colours due to the fairly broad spectral characteristics of the inks or phosphors used. Since no two output devices have exactly the same gamut, an additional process, colour profiling, is often applied. Colour profiles are developed for each device. These are typically non-linear lookup tables that map the standard colour space into the actual colour gamut of the specific device so that every value presented to the device has some output. Attempting to get the colours in these actual outputs to resemble one another is the science of colour management. Colour profiles are generally defined according to a format developed by the International Colour Consortium (ICC).

10.5.5.4 Detector "Gamut"

Strictly speaking, a detector alone does not have a gamut because the definition of gamut requires presentation to the eye. However, detectors do have ranges over which they can produce accurate data representing colours presented to them. Except in rare cases, detectors used for imaging do not have spectral response curves that are designed to match the tristimulus curves directly. Instead, the signals from detectors may have a conversion matrix applied of the same form as those used for output devices and may also have colour profiles applied. In fact, the entire ICC colour effort is focused on supporting the concept that all input and output devices should communicate through a common colour space. The colour space of choice is, of course, the tristimulus space because it incorporates all colours discernable to the eye. Thus, the capability for accurate colour reproduction for detectors rests on their ability to produce signals that can be accurately transformed (preferably by linear simultaneous equations) into XYZ tristimulus values.

10.5.5.5 Illumination Effects

When the tristimulus values for a viewed object are calculated, the spectral power density received carries no distinction between the spectral reflectance properties of the object and the spectral emission properties of the illuminating source. The received distribution is merely their product. To separate the object from the source requires either specifying the source or illuminating the object with spectrally flat energy. Because blackbody illuminators follow the theoretical emission curves reasonably well, specifying the source is generally easier so this practice was long ago adopted. This leads to the requirement that measured colour data be supplied with an illuminant tag like D50 or D65, which specify 5,000 and 6,500 K illuminants. Colour data taken with one illuminant can be transformed into data as it would be under another illuminant through a linear 3 × 3 matrix. This process is applicable for any illuminant, not just blackbodies.

10.5.6 Colour Detection

Specifying both chromaticity and brightness requires three numbers regardless of the colour space chosen. The function of a colour detector, in machine vision often a camera, is to determine these three numbers from an incident optical image. The type of detector required depends on the accuracy of the colour data required. Generally, this accuracy is stated in the L^*a^*b colour space because that space was designed to be perceptually uniform. The error in each of the three dimensions could be stated separately but common practice is to combine them as the length of a Cartesian vector from the true colour to the measured colour. This value is reported as ΔE, pronounced delta-E, and is defined so that a ΔE of 1 is the smallest variation that the eye can see. The error in professional colorimeters can be as low as 0.001 while the error in high-quality digital cameras is typically in the range of <1–10 with an average over all colours of about 5–6.

In practice, three detectors with realizable colours filters will not achieve average errors below about 5 if the entire XYZ tristimulus space is to be measured. In real cameras, improved colour accuracy in one colour region – skin tones, for instance – might be traded for larger errors in a less critical region like blues. To reduce the error to the neighbourhood of 1–1.5 for all colours, about six channels are needed. This is why most commercial colorimeters for tasks like monitor calibration have seven or so filtered sensors.

10.5.6.1 Colour Separation

Fundamentally, accurate colour measurement requires that the spectral response of the various detector channels, however many there are, can be linearly combined to approximate the tristimulus curves. For any detector, except a few very expensive colorimeters in which the tristimulus curves are replicated very closely, there will be some error in colour measurement based on the mismatch between the detector and tristimulus curves and some uncertainty based on the S/N of the detector. In cameras, there can also be non-uniformities in the measured colour due to the same problems that appear in monochrome imaging and due to glare in the optics.

The techniques to approximate the tristimulus curves fall in two broad categories, those which try do it with exactly three detector channels and those that use more. Those that use many more channels are essentially hyperspectral imagers, using 10 or 20 or 50 narrowband channels to provide accurate spectral distribution curves for the objects being viewed. With these systems, matching the tristimulus curves becomes a purely mathematical operation in which the spectral sensitivities of the numerous channels are linearly weighted and combined. Increasing the number of channels can bring these approximations arbitrarily close to the normative tristimulus curves. Of course, this technique carries substantial burdens of cost, complexity and speed.

In imaging, cost is still very much the issue so techniques using one or three image sensors account for virtually all commercial implementations. These have been of three basic types: one-sensor sequential, one-sensor simultaneous and three-sensor.

One-Sensor Sequential

Sequential colour imaging is simply an extension of the usual method of monochrome frame scanning. For colour acquisition, three successive frames are taken, each with a colour filter inserted in the optical path. Several varieties of filters can be used. Historically, these have been three Kodak gelatin filters – Wratten types 25 (red), 58 (green) and 47B (blue) are a widely used triplet – that separate the spectrum into three relatively non-overlapping sections. Other dye and interference filters sets with this spectral behaviour have also been developed. Alternatively, the filter set may use complementary colours – cyan, magenta and yellow – each of which takes in two of the RGB colour bands. The CMY scheme is more light-efficient that the RGB but requires calculating strong differences between channels to produce the tristimulus approximations. Which technique is better depends on a variety of image sensor characteristics and the relative importance of various image quality measures.

In so-called *tri-linear* sensors, a variation on this technique uses three lines, each with its own filter, to sample each line on the object three times as it moves past. This operation can be executed at the same speed as monochrome line imaging but carries the risk of slight changes in the object being viewed due to the scanning time separation.

One-Sensor Simultaneous

The most common method of colour separation uses a monochrome image sensor provided with a colour filter array (CFA), with one colour filter per photodiode. Various CFA configurations are possible but the most common is the Bayer arrangement in which a four-pixel group is provided with two green filters and one each of red and blue in a diagonal configuration. A complementary version of this design and other filter arrangements are also used as shown in ➋ *Fig. 10.37*.

Luminance resolution in cameras with colour filter arrays is approximately 70% of what would be produced by a monochrome sensor with the same pixel count. This results partly from the not having direct luminance samples from each pixel and partly from the anti-aliasing filters that are needed in CFA cameras to control colour artefact generation. Colour resolution is lower. As a result, aliasing from undersampling begins at different frequencies for the various

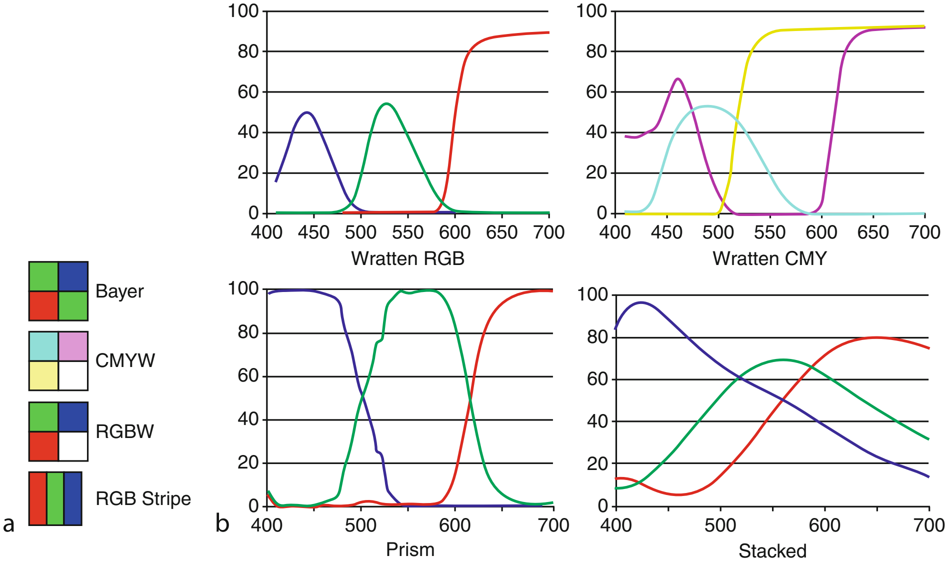


Fig. 10.37
Colour filter arrays (a) Filter configuration. (b) Spectral transmission curves

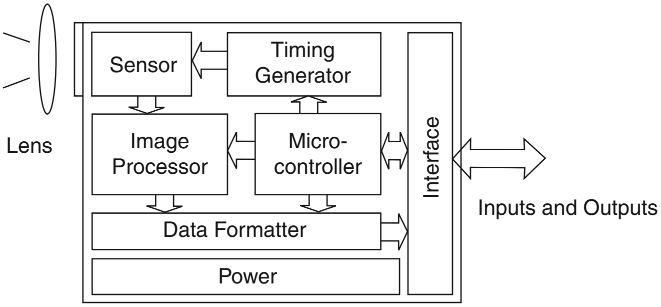


Fig. 10.38
Camera block diagram. The organization of particular cameras may vary but this is a useful conceptual aid

colour components and the direction of strong aliasing components varies by colour. This nearly always results in visible colour artefacts when significant energy at spatial frequencies above the various sampling limits is allowed to reach the sensor. Typically, the missing colour information is filled in with estimates made from close neighbours. This can generate colour errors due to mismatches in the estimates from pixel to pixel. In some implementations, no attempt is made to fill in the missing information; each colour signal is simply treated as complete but of lower resolution. This procedure avoids generation of reconstruction artefacts but reduces perceived resolution and can cause ambiguities in determining the positions of colour edges.

Sensors are now also available in which each of the locations has three photodiodes built in a stack inside the silicon. Since silicon has an absorption curve that allows photons to penetrate farther as their wavelength increases, the silicon itself can be used to separate colour. Charge generated at each level can be separately collected by the three stacked photodiodes to provide three spectrally-separated colour signals. This arrangement assures that the colour channels are optically aligned.

Three-Sensor

In order to combine full resolution and simultaneous colour imaging, three sensors can be used. Most commonly, these sensors are fitted on a prism assembly incorporating internal colour filters to direct the R, G and B bands to the three sensors. In these imagers, the monochrome and colour resolution are identical except for any errors, generally small, in the alignment of the sensors to each other. Separation prisms of this type must be carefully designed to provide identical optical path lengths for the three bands. (➋ Chap. 4) Often, special optics are required for low colour aberration because the lens design must take into account the thick glass region between the lens and the imager. Because dichroic multilayer filters are almost universally used in prism-based imagers, overlap in the colour bands is minimal.

10.5.6.2 Spectral Characteristics

Typical curves for several sensor filter sets are shown in ➋ *Fig. 10.37*. The first three sets show filters alone but the stacked set necessarily includes the spectral response of the sensor. All of these sets would need auxiliary ultraviolet and infrared blocking filters to restrict the light arriving at the sensor to visible wavelengths.

Each filter set when used with a sensor makes up a detector with its own colour space, producing a set of readings for each colour presented to it. This colour space will be different for every detector type and even, in a minor way, for each individual detector. For the output to be useful for anything except comparisons using only one sample of the detector, the colour space for the detector must be transformed into a standard colour space. This can be accomplished using a 3×3 matrix or some other colour coordinate transformation technique. Such matrices are generally developed by imaging a standard colour chart and computing the matrix values that fit the measured values to the true values most closely. While all of the three-sensor separation techniques can produce similar colour accuracy performance, the errors generated by each tend to fall in different locations on the chromaticity chart.

Each of these filter sets has advantages and disadvantages. Mildly overlapping curves enhance the detection of wavelength shifts in narrowband sources better than strongly separated curves while the separated curves are better able to discriminate between narrowband and wideband sources with the same central wavelength. Separated curves often have difficulty accurately detecting changes in wavelength near the transitions. However, if the curves overlap too much, then the numbers in the transformation matrix become large, adversely affecting the S/N of the transformed data. In typical implementations, filters with overlapping curves have high light losses as well because the light not used is absorbed. This does not apply to the stacked sensor because all of the light is absorbed in the silicon. In the three sensor designs, essentially all of the light passes through one of the channels in the colour separation prism so minimal light is lost although some designs incorporate trim filters which will reject or absorb some light.

One example using the Wratten RGB set will serve to illustrate the problems with particular types of curves. Let us assume that we have two monochromatic stage lights, one emitting around 420 nm (clearly violet) and one around 460 nm (deep blue). If we observe the stage with a camera using the Wratten RGB set, we will find that both of these sources produce a signal only in the blue channel. This means that our camera would not be able to tell the difference and would report both of these sources as having the same colour. When this colour data is displayed, it will appear as whatever has been assigned to a blue-channel-only signal. So, there will be no violet in the picture. Of course, this is just as well because there is no violet inside the sRGB display gamut anyway. Next time you are at a concert, wait until the violet spots come on and then look at the video monitors.

In some cases, two objects with different reflectance spectra can generate the same colour coordinates. This phenomenon is called metamerism. In imaging systems, metamerism is locked in at the time of detection. Once the detection channels produce the same signal set with two different stimuli, there is no way to distinguish these cases by subsequent analysis. This is a case similar to that found in aliasing because like aliases, once the confusion is generated it cannot be removed. Metamerism can be minimized through the use of an illumination source with a high colour rendering index (CRI) like halogen lamps. White LED illuminators have generally low CRIs (➜ Chap. 4).

Often, the colour images are not displayed; the goal is instead the accurate measurement of colour. In such systems, the colour separation requirement is unchanged because the accurate measurement of colour requires that the detector channels produce signals that can be combined to produce a three signal set that is the same as the set that would be produced by detectors with the spectral characteristics of the tristimulus curves. To some extent, this is an easier task than colour reproduction because the colour gamut constraints of the display are removed. Fundamentally, though the requirement remains: can the spectral characteristics of the sensor channels be combined to approximate well enough the tristimulus curves?

10.5.7 Multispectral Generalization

The technologies used to produce three channels for colour detection can also be applied to other wavelength bands. One longstanding example is the four-channel scanners used to digitize colour film. Three of the channels are tailored to match the spectra of the three film dye layers and the fourth detects near-infrared. Since all of the dyes are transparent in the infrared, the fourth channel contains only an image of irregularities on the plastic film base – scratches, discolorations and the like – so the fourth channel can be used as a reference to correct these same marks on the images of the colour channels to produce a digital image that is cleaner than the film original.

Near-infrared is also useful as an adjunct in inspecting agricultural products. In some cases, the channels may be just one narrowband visible and two near infrared. Making these changes is relatively straightforward in prism cameras because changing the spectral characteristics in these is precise and reliable. CFA (Colour Filter Array) [9] cameras allow much less flexibility due to the limited selection of the filter dyes available and the need to incorporate the filter deposition into the semiconductor fabrication process. For larger pixel spacings, dichroic filter arrays can be fabricated to provide substantially more options but these sensors still have the same geometric limitations of Bayer CFA sensors. The current limit for one-piece prisms is five channels but the number of different filters can be quite large if the spatial considerations are not important.

10.5.8 Hyperspectral Imaging

A camera with a large number of colour channels can produce hyperspectral images, that is images that detect the complete spectral energy distribution over a particular band for each resolution element at the object. The simplest hyperspectral cameras use some sort of spectrally-dispersive element like a grating or a filter in which the wavelength transmitted changes from one value at one filter edge to another values at the opposite edge. In both cases, an area sensor is presented with an image of a line on the object so that one direction resolves elements along the line and the other direction resolves the spectrum coming from each element on the line. If an entire area must be imaged at once, then the camera can be fitted with a variable filter which can be swept through the wavelength range of interest. Such cameras usually have high-speed sensors so the wavelength sweep, which may include 100 or more images, can be completed as quickly as possible.

In dynamic hyperspectral imaging, the data forms a four-dimensional hypercube – x, y, wavelength and time so the data rates can be high and data volumes can be large. Often, hyperspectral imaging is used as a tool to quickly determine which bands contain useful information for a particular task. Those bands can then be designed into a fixed channel camera matched to the task.

10.6 Cameras

Thousands of camera models are available so selecting the right one for a specific task can consume significant time and resources while still producing a sub-optimal match. Information on cameras is useful but often so much information is available it further complicates the search. Further, each camera seems to carry a few specifications that are based on assumptions that do not match those of any other camera so direct comparisons often require substantial additional research. Testing the candidate cameras would seem to be a useful way of making a selection but interface differences in everything from power to software make such comparisons tedious and often, at the end, unenlightening. Further, testing requires simulation of the expected conditions in the real application. This is rarely achievable in any way approaching completion simply because support, lighting, optics and other systems considerations are usually also undetermined during the camera selection process. The rate of new sensor and camera introduction is so high that the best selection may not even exist at the beginning of a search. It is thus imperative that the important criteria for the success of an application be carefully determined and that all camera selections be measured strictly against those criteria so that new candidates can be quickly accommodated.

Camera configuration and performance specifications begin with a relatively simple set of requirements on which are built layers of increasingly specific details [10]. It is of no use to focus on a detail unless the foundation for and effects of that detail are understood. The camera should be thought of at the beginning as a black box that is required to perform a set of functions. Then the implications of those functions on the camera specification can be considered. Then specific cameras meeting those specifications can be identified. Tradeoffs may be necessary, so the relative importance and range of tolerance on the required functions must be determined to allow ranking of the candidate cameras. Factors external to the camera, such as the reputation of the supplier, may also need to be taken into account but delaying this step until after the candidates are identified is recommended.

Finally, it is extremely unwise for systems designers to take on faith the performance of cameras without understanding in some detail the sensors inside them. Finding examples of cameras that use sensors incorrectly or claim camera performance that exceeds published sensor capabilities or fail to take advantage of sensor functions carefully implemented by their designers is not difficult. As a result, those selecting cameras need to be able to detect these discrepancies and understand the explanations provided by camera vendors. Sensors don’t do mysterious things when run properly and camera electronics do not perform magic so the explanations should be clear and rational. The material below is intended to provide a basis for such rational analysis.

10.6.1 Requirements for Visible Cameras

Strictly speaking, this section applies to cameras with silicon sensors operating in the wavelength band of approximately 350–1,100 nm. This omits something like one part in 10^{10} of all cameras, including primarily various types of infrared cameras, a few ultraviolet models and all high-energy imaging. The important requirements for those bands can be very different from both visible cameras and from each other and can be driven by very specific applications. As a result, generalizations become much less useful and guidance requires much more specific applications engineering. Some of the present discussion is relevant but should be tempered by acknowledgement of the significant differences.

10.6.1.1 Analogue or Digital?

With computers everywhere, this question almost always is really, “Where should the analogue to digital conversion occur”? Even security cameras are rapidly turning digital because Ethernet links compared to analogue coax are:

- Much less susceptible to noise
- Easier to provision with camera power
- Inherently bidirectional
- Easier to use for camera control
- Cheaper and simpler to extend
- Much easier to access from multiple viewing points
- More directly convertible to fibre optics
- More compatible with transmission of compressed or encrypted data
- The winner in the time-certainty vs. signal-delay tradeoff

Digital cameras are not even necessarily more complex than analogue cameras because of the availability of on chip digitization in CMOS sensors. Broadcast television is now digital, consumer camcorders are digital. The only analogue cameras left are single-board types and the heads in some two-piece cameras. Even in these, very little processing is done in the analogue domain. Interestingly, what processing remains in analogue cameras is also still present in digital cameras as part of the signal preparation for digitizing.

10.6.1.2 Mechanical

The basic camera for industrial applications is a metal (usually aluminium) box with holes for mounting, a big hole to let in light, some electronics inside and connectors for power and signals. It is unlikely that anything but metal can serve as an appropriate enclosure due to the requirements for rigidity, thermal stability, electronic shielding and tolerance for holes. Occasionally, users will supply their own boxes and need only to buy the electronics, thus assuming responsibility for the items just listed. In more explicit form, the mechanical requirements are:

- Sufficient mounting provisions and adequate rigidity to assure that the camera will remain pointed accurately in the desired direction and attached securely to its platform.
- Lens mounting fixtures that secure the selected optics, locate them correctly, exclude contaminants and provide lens orientation and back focal adjustments where necessary.
- Enclosure construction that provides appropriate protection against the environment, sufficient radiation of internally-generated heat, easy access to internal assemblies as needed and accommodation for non-interfering attachment, retention and removal of any electronics cables.
- External finishes meeting requirements for durability, electrical conductivity, optical reflectance and compatibility with other system elements.
- Size, shape and weight conforming to any optical and physical constraints.

Some specific design features to look for are:

- Separation of covers from support members. While consumer products are usually satisfactory if they fall apart when the covers are removed, industrial products built this way are at risk of falling apart on their own due to vibration. Of particular concern is the possible misalignment of the mountings for optical components when a camera is disassembled.
- Solid connections between the camera mounting surface and the lens mount. This requirement assures that the camera and the lens point in the same direction. Many industrial cameras are designed to mount using the same plate that holds the lens, using front surface or peripheral threaded holes. Those that mount on the base generally have a single casting for front and base or very sturdy mates between the two pieces. The mate should include a positive alignment feature and not depend solely on pressure applied by fasteners.
- Adjustable back focal distance. Not essential because extenders and shims can be used with threaded mounts, but convenient. Generally not necessary if photographic lenses are used because these are carefully calibrated unless, of course, the cameras are not. Many industrial lenses have built-in adjustments.
- Rugged, reliable cable connections. Not strictly part of the enclosure but very important mechanically. Screw-retained cables are always preferred and non-locking cables like consumer power connectors should be avoided. All cables should be supported separately from the camera anyway. Check to make sure the connector fasteners are installed correctly so the cables fully mate.
- Non-rotating camera mounts. Tripod mounts are discouraged unless they include an anti-rotation pin. Two screws will work but separate pins for location and screws or quick-release fasteners for retention are better.
- Positive board retention. All internal circuits need to be anchored to something solid. Connectors cannot be relied upon for mechanical support. Boards should not provide

support for anything except electronic components. Large components should be individually retained.

- Adequate heat paths. Verify that all components with significant heat dissipation are adequately cooled. Some cameras may require external heat sinking through the mounting surface. Be sure to understand any such requirement and implement them properly. Ask about required air flow rates over finned heat sinks.

Keep in mind that many optical systems need support separate from the mount. Many lenses are now bigger and heavier than the cameras using them. However, it is never acceptable to let the lens support the camera.

10.6.1.3 Electrical

Providing power to a camera seems simple enough since most run on a single DC supply. Unfortunately, many opportunities for injection of noise to the camera signal relate to power provisioning. Because of the wide variety in power and grounding schemes internal to cameras, universal recommendations are not possible. Here are some things that should be determined for each camera under consideration:

- Grounding scheme. In some cameras, the case is grounded to the external supply return but in others the case floats. The importance of this in a system depends on whether the camera mount or the lens are grounded or isolated and the configuration of the grounding on the supply providing external camera power. The battle here is between ground currents and electrical interference. The situation can be even more complicated depending on the way analogue and digital grounds are connected in the camera.
- Power supply noise rejection. Some amount of the noise on the power supply line can find its way into the image data. How much depends on the rejection ratio. Typically, the rejection gets worse as the frequency increases so switching power supplies can be problematic. Simple arithmetic tells how much rejection is needed. If the sensor puts out 1 volt full scale and 12-bit data is needed, then the maximum allowable conducted noise is about 250 μV. If 250 mV (peak to peak) is present on the power supply output, then the rejection needs to be at least 60 db. Higher rejection ratios will be needed if the camera is operated with additional gain.
- Input voltage range. Long power supply runs can reduce voltages at the camera below the camera tolerance range. Many cameras on the same power bus can cause voltage fluctuations. Know what tolerance the camera has and understand what changes at the camera output when the input voltage changes while the camera is running. Putting anything else on camera power buses is a risky practice.
- Overvoltage tolerance. Determine the effect of applying voltages, both DC and transients, above the recommended range. Ask what is necessary to change blown fuses or other protective elements.
- Reverse voltage tolerance. Cameras should not blow up if the power is connected backwards. Some of them will run either way. However, see the point above about grounding.
- Inrush current. Determine if the camera draws significantly more current during turn-on. This is especially important if many cameras are on the same bus because turning them all on can result in a supply voltage below the tolerance.

- Transient recovery. Some cameras will not start properly if the voltage is brought up slowly from zero. Some can even be damaged by this. Others lock up if the voltage drops below the minimum for a short time. These behaviours should be understood for each candidate.

10.6.1.4 Optical

The bulk of optical information will be found elsewhere in this book. In this section, the goal is to discuss the significance of a few optical characteristics that cameras may have. Warnings have been given about the need for controlling light behind the lens and for accommodating microlenses. Here are a few more potential problems:

- Front reflection. If the front of the camera is not hidden behind the lens from the perspective of the object being viewed, then ensure that the exposed surfaces are not reflective. At the least, this can increase stray light problems, especially where uniform illumination is desired. At the worst, the front of the camera can appear as a reflection in the image to be measured.
- On-axis reflection. Any optical element perpendicular to the optical axis of the lens has the potential to reflect light back through the lens and have this return to the image plane out of focus and in a different location. Possible sources for such reflections include the sensor surface, the sensor window and any filters or other optical elements between the back of the lens and the sensor. Reflections should be minimized by choosing optical components that have appropriate coatings (➲ Chap. 5).
- Light leaks. Assure that all materials surrounding the cavity behind the lens are opaque. Most cameras have LEDs inside which can provide opportunities for stray light at any time. Most sensors are silicon so they can detect radiation in the near infrared that won't be obvious to the eye. Some dyes used to make plastic black will still transmit NIR. Small leaks around the lens mount may not show up until a bright strobe is used.

Lens Mount Characteristics

Lens mounts come in three general types: threaded, such as the C-mount, bayonet, such as the Nikon F and breechlock such as the Canon FD. Each type has certain mechanical advantages and disadvantages but these are rarely significant in machine vision use. For compatibility with sensors and cameras, two factors are important: flange focal distance, to indicate how much space may be available for accessories that may be needed behind the lens, and image diameter to assure that the sensor can be covered with minimum vignetting. The JIIA LE-002-2008 lens-mount standard and its companion JIIA LE-001-2007, which defines specifies image sensor sizes, are currently gaining in popularity [11].

➲ *Table 10.1* provides details on some common lens mounts. Note that for all lenses, the flange focal distance is the distance from the reference surface on the lens mount, to the image plane in air. In most lenses, mechanical or optical elements will protrude behind the reference surface into the space between the lens mount front surface and the sensor. It is thus imperative to know the actual mechanical configuration of a lens to assure that no interference will be caused by protruding elements. Image diameters are nominal. Some photographic lenses have mechanical masks that limit the illuminated area on the image plane to slightly larger than the active area on the digital sensor or film. Discovering this is part of lens selection.

Table 10.1

Lens mount	Flange focal distance (mm)	Mount type	Image diameter (nominal – mm)
C	17.526 (0.69 in.)	1-32 UN 2A thread	20
CS	12.52	1-32 UN 2A thread	10
Canon EOS	44	Bayonet	33
Canon FD	42	Breechlock	43
Four thirds	38.67	Bayonet	22
Hasselblad	74.9	Bayonet	85
M42	45.46	42-1 thread	43
Nikon F	46.5	Bayonet	43
Pentax K	45.5	Bayonet	43
T2	55	42-0.75 thread	43

Sensor Size Designations

Sensor sizes are designated according to three different systems: camera-tube equivalent, focal length multiplier and named. The camera-tube equivalents are all calculated from the ratio between the image size and glass envelope diameter in television camera tubes. The original vidicon tube had a one in. OD glass envelope and a 3/8 × 1/2 in. (5/8 in. diagonal) image area. Later, 30 mm (1.2 in., 2/3 and ½ in. tubes were made. The sizes have since been converted to metric so the 1-in.size is now 9.6 × 12.8 mm (16 mm diagonal). Sensors come in many sizes from 1/13 to 1.5 in., most of which never existed in image tubes. As the sizes get smaller, the real sensor dimensions deviate from the calculated ratios.

The scale for focal-length multiplier sensors sizes is based on the field of view of a 35 mm film frame (24 × 36 mm). The size number indicates the change in effective focal length of a lens when used with a sensor that is a different size from film. For instance, an imager with dimensions of 16 × 24 mm would be designated as 1/1.5 because that image size is the 35 mm frame divided by 1.5. The number is called a focal length multiplier because a 50 mm lens on the 35 mm frame would appear to be a 75 mm lens on the smaller frame if the field of view of the two cameras were compared. Unfortunately, the two designator scales overlap so the actual sensor dimensions must always be verified.

Some sensors have only names assigned by manufacturers or trade associations, presumably for commercial advantage. These are almost always sensors for digital still cameras. ➋ *Table 10.2* shows the commonly available sensor sizes. Many others without designators are available.

Sensor Matrix Designations

The number of pixel locations in available area sensors can be anywhere in the range of tens of thousands to tens of millions. At least one sensor with over 100 million locations is now available. The frame rate required and the number of pixel locations determines the volume of data that must be processed in each image. This product must further be multiplied if stacked photodiode sensors or multisensor prism assemblies are used for colour or multispectral

Table 10.2

	Designator	Width (mm)	Height (mm)	Diagonal (mm)	Notes
Image tube scale	1/13	1.120	0.840	1.400	All 4:3 aspect ratio
	1/10	1.440	1.080	1.800	
	1/7	1.850	1.390	2.300	
	1/6	2.150	1.610	2.700	
	1/5	2.250	1.910	3.200	
	1/4	3.200	2.400	4.000	Other sizes have been used
	1/3.6	4.000	3.000	5.000	
	1/3.2	4.536	3.416	5.680	
	1/3	4.800	3.600	6.000	
	1.2.7	5.270	3.960	6.592	Other sizes have been used
	1/2.5	5.760	4.290	7.182	
	1/2	6.400	4.800	8.000	
	1/1.8	7.178	5.319	8.932	
	2/3	8.800	6.600	11.000	
	1	12.800	9.600	16.000	Nominal starting point
	30 mm	17.100	12.800	21.360	
	4/3	18.000	13.500	22.500	Not the same as four thirds
Focal length multiplier scale	1/1.7	21.176	14.118	25.471	All 3:2 aspect ratio
	1/1.5	24.000	16.000	28.844	
	1/1.3	27.692	18.462	33.308	
Designations by name	HDTV	9.600	5.400	11.000	16:9
	Four thirds	17.300	13.000	21.640	4:3 AR (approximately)
	APS-C	21–29	14–19	25–35	3:2 AR, not cine APS-C
	Full frame	24.000	36.000	43.266	3:2 AR
	APS	25.100	16.700	30.148	3:2 AR
	APS-H	28.700	19.000	34.419	3:2 AR, not cine APS-H
	645	60.000	45.000	75.000	4:3, also smaller sizes

applications. While there is no reason why any particular number of locations is better or worse for fabrication, some sizes show up repeatedly, mostly for historical or other non-technical reasons. ➲ *Table 10.3* shows common pixel matrix sizes. These arrays can be made to match sensor sizes shown in ➲ *Table 10.2* within the constraints of manufacturable pixel pitch. As

Table 10.3

Name	H Count	V Count	Matrix AR	Point AR	Notes
SQCIF	128	96	4:3	11:9	
QQVGA	160	120	4:3	1:1	
QCIF	176	144	4:3	11:9	
QVGA	320	240	4:3	1:1	Also called SIF
CIF	352	288	4:3	11:9	
HVGA	480	320	4:3	1:1	
CGA	640	200	4:3	1:1	
EGA	640	350	4:3	1:1	
4QCIF	704	576	4:3	11:9	
CCIR	720	576	4:3	8:9	ITU R-601
NTSC	768	480	4:3	5:6	ITU R-601, also 480i, SDTV
480p	768	480	4:3	5:6	Also EDTV
VGA	640	480	4:3	1:1	
SVGA	800	600	4:3	1:1	
XGA	1,024	768	4:3	1:1	
XGA+	1,152	864	4:3	1:1	
720	1,280	720	16:9	1:1	HDTV-720p
Unnamed	1,280	960	4:3	1:1	Equivalent to quad VGA
SXGA	1,280	1,024	5:4	1:1	
WXGA	1,366	768	16:9 approx	1:1	Also several other matrices, loosely used
SXGA+	1,400	1,050	4:3	1:1	
16CIF	1,408	1,152	4:3	11:9	
WXGA+	1,440	900	16:10	1:1	
UXGA	1,600	1,200	4:3	1:1	
WSXGA+	1,680	1,050	16:10	1:1	
1080	1,920	1,080	16:9	1:1	HDTV-1080p, also 1080i
WUXGA	1,920	1,200	16:10	1:1	
2 K	2,048	1,080	1.896:1	1:1	Digital cinema, use ROI for higher AR
QXGA	2,048	1,536	4:3	1:1	
WQXGA	2,560	1,600	16:10	1:1	
QSXGA	2,560	2,048	5:4	1:1	
QSXGA+	2,800	2,100	4:3	1:1	
QUXGA	3,200	2,400	4:3	1:1	
WQUXGA	3,820	2,400	16:10	1:1	
QHDTV	3,840	2,160	16:9	1:1	Quad 1080 HDTV
4 K	4,096	2,160	1.896:1	1:1	Digital cinema, use ROI for higher AR
HXGA	4,096	3,072	4:3	1:1	
WHXGA	5,120	3,200	16:10	1:1	

Table 10.3 (Continued)

Name	H Count	V Count	Matrix AR	Point AR	Notes
HSXGA	5,120	4,096	5:4	1:1	
WHSXGA	6,400	4,096	25:16	1:1	
HUXGA	6,400	4,800	4:3	1:1	
UHDTV	7,680	4,320	16:9	1:1	Super Hi-Vision, 16× 1080p
WHUXGA	7,680	4,800	16:10	1:1	
8 K	8,192	4,320	1.896:1	1:1	Digital cinema, use ROI for higher AR

smaller and smaller pixel pitches become more feasible, the number of locations in any particular sensor size increases. Care must be taken to assure that the resolution spatial resolution requirement at the array does not exceed the resolution capabilities of the optics. Here are explanations of a few terms in *Table 10.3*:

- Name. Most of these array sizes correspond to designators for video display resolution. These need to be used with care because in displays, the names refer to displayable white points, usually implemented with RGB triads. This means that the number of addressable elements in colour displays is actually three times the stated matrix count. This is not the case with CFA sensors, where the matrix stated is the total number of photosites which are partitioned into colours. Stacked diode sensors and RGB prism assemblies are categorized according the matrix size of a single photosite plane; however, the total data rate must take into account the multiple photosite planes.
- H count and V count. These are the number of active matrix locations in the horizontal and vertical directions. All of these are nominal because even the display manufacturers are not perfectly consistent and the display standards allow a range for many of the designators.
- Matrix aspect ratio. Originally, television images had an aspect ratio of 4:3 and computer displays followed suit. Later, widescreen displays were added to both television and computers but with different shapes. All along, some displays have had unique characteristics. Some of these decisions still affect compatibility of software and hardware in imaging systems.
- Point aspect ratio. While most displays and sensors now are arranged with equal horizontal and vertical pitch, those based on television standard are not symmetric due to the original accommodation of the unequal horizontal and vertical resolution of the eye. Arrays with non-square point aspect ratios will be squeezed in one direction when viewed on equal-pitch displays.

Matrices are listed in order of increasing horizontal count.

Decoding the names requires extensive reading on the history of displays. Here are presented the words that form the mnemonics, but not the historical background.

- Names ending in CIF. CIF means Common Interchange Format and is a reduced resolution matrix based on PAL digital video resolution. CIF is 1/4 of the PAL matrix. QCIF is 1/4 of the CIF Matrix, SQCIF is 1/9 of CIF. 4CIF is the same as the PAL matrix and 16 CIF is 4× PAL or 16× CIF. All of these are defined in the ITU H.261 and H.263 standards.

- Names ending in “GA”. GA means Graphics Array of which there are many types – C = Colour, E = Enhanced (except that for these two, A means adapter because it refers to actual hardware), V = Video and X = eXtended, There are modifiers H – which has two meanings: “Half” (approximately) when used with VGA and “Hex” (16×) when used with everything else – Q = Quad S = Super, U = Ultra, W = Wide and + for a few array sizes that don’t follow simple arithmetic.
- Video formats. These are really abbreviations using either the name of the standard that defines them, like NTSC, or the actual number of rows in the display as in 1080p.
- Interlacing. All formats are assumed to be scanned progressively unless the name ends in “i”, meaning interlaced.

Although square arrays (512 × 512, 1,024 × 1,024, etc.) have no names, they are also common in industrial and scientific imaging. Square displays are rare.

10.6.2 Environmental

Environments for industrial camera can include essentially every potential hazard, from electrical discharge to heat to mechanical shock. The fundamental requirement for successful implementation of any machine vision application is assuring that the image collected adequately captures the required information from the object being viewed. The straightforward implications of this are that the view cannot be obscured or significantly distorted, the camera cannot move relative to the object and the acquired image must be accurately translated to a signal for transmission. The first two of these relate primarily to concerns external to the camera. Here are some important considerations regarding the third:

- Vibration. Cameras must be isolated from high-frequency (more than a few hundred Hertz) vibration. The amplitudes of such vibrations are usually too small to cause substantial blurring in the image but they will eventually disassemble the camera regardless of the care taken to secure fasteners. Worse, they will likely lead to fracture of solder joints on the boards and even failures inside some components. Lower-frequency vibrations cannot be tolerated because they will move the camera relative to the object causing loss of resolution.
- Shock. Typical cameras now are relatively immune to shock because their masses are so small. Shock during operation can set up resonances that blur the image but isolated shock during handling and even operation are not likely to result in serious damage. Even so, stay within the limits established for specific cameras because striking a camera just the wrong way can damage the sensor window. Of course, if the camera needed for a particular application is large or contains devices like image intensifiers, shock should be carefully avoided even in shipping.
- Temperature. The insides of cameras can be much warmer, perhaps 30°C or more, than the ambient temperature. The specific issue with this is that the sensor is not running at ambient temperature so any calculation of sensor performance must use the real sensor operating temperature. Most other components and properly-designed circuits are not susceptible to temperature changes in ways that affect the camera signal more than the changes from the sensor will. Perhaps the most important non-sensor change involves timing of the sensor drive signals and the digitizing clock. In CCDs, for instance, relatively

minor shifts in the transport register drivers can have significant effects on CTE, causing image blurring in the line direction.

One other concern is condensation at low temperatures. Few cameras are sealed against moisture so falling temperatures can lead to condensation. The absolute temperature is not important. At high relative humidity, condensation can occur during falling temperatures even if the camera is hot. Condensation often shows on the sensor first because the sensor window may be the coolest place in the camera. It is useful to understand how each camera type will behave if condensation occurs and what should be done subsequent to a condensation event.

One extreme case of a temperature effect is the permanent conversion of amorphous selenium x-ray detection layers to a different phase that is not sensitive simply by exposure to the low temperatures encountered during air freight transport. Such devices cannot be air shipped.

- RFI/EMI. The camera body is essentially a Faraday cage for the interior components except for the perforations for the lens and connectors. However, these small holes can still allow reception of significant RF interference, especially if the lens or the cables are acting as antennae. Proper shielding and grounding can mitigate these problems but if a big motor contactor sits right next to the camera, it will generate white dots in the image no matter what is done. Protection is not impossible. Cameras on can coating lines can endure hundreds of what are essentially lightning strikes without failing because the system is designed to accommodate those. There is likely no substitute for testing candidate cameras in the target environment to uncover EMI/RFI problems.
- Ionizing radiation. When a camera is to be used in a system where radiation is used to form images, candidate cameras must be evaluated for their resistance to radiation at the expected exposure levels. Even where cameras are heavily shielded, random hits on the sensor and camera electronics are to be expected. Even in some applications away from main radiation beam, such as monitoring inside radiation therapy treatment suites or monitoring reactor processes, significant exposure can occur. The transient and cumulative effects of any exposure should be understood, especially if those effects are irreversible.
- Pressure. Changing pressure can have disastrous effects on some electronic components, not the least, image sensors. Most image sensors, unless in hermetic packages, use epoxies to secure the window. Gases can flow through epoxies, especially under the influences of pressure differentials. Pressure higher than the pressure under the window can force in water vapour, which can later condense on the inside of the window and the sensor surface. Generally, this occurs when the external pressure is first lowered, as during flight, long enough for the pressure under the window to drop so that subsequent return to normal atmospheric pressure forces water vapour in. Plastic bags and desiccants are recommended whenever cameras are air shipped.
- Cleaning materials. On food lines especially, the cameras must be protected from washdown and sterilization requirements. It is unlikely that many cameras are suitable for operations in such environments without additional protective enclosures. Nevertheless, it is wise to understand before an accident happens what effects contact with the cleaning materials would have on the camera so appropriate service actions can be ready to execute.

The various environmental factors can interact to produce higher susceptibility to poor operation or failure inside what would be normal tolerances for a single factor. In cameras intended for repeated flight, for instance, repeated temperature, pressure and humidity changes can lead to condensation problems even if none of the specifications have been exceeded.

10.6.3 Regulatory

Regulatory compliance requirements vary by application. Laboratory applications may require no compliance at all, while medical applications involving control or life support may require complex compliance including formal verification of software. The most common requirement is third-party safety certification such as that required to get UL certification or a CE mark. Lack of required compliance documents can result in stop work orders, especially in jurisdictions where safety signoffs by the fire department, for instance, are required to allow system turn-on. Regulatory requirement that may apply to cameras include:

- General safety. These certifications address primarily fire and shock hazards. Different classifications exist for various types of equipment and sophistication levels of target users. Specific responsible agencies are usually geographically determined. Each country may have its own agency with some deviation from universal requirements. Be sure any camera selected meets the requirement of the planned final installation locations. In some cases, entire systems may be submitted for certification by the system builder even if not all of the components are certified.
- EMI/RFI. Certification may be required to demonstrate both immunity to external electromagnetic interference and freedom from excessive radio frequency emissions for cameras and other active electronic equipment. Testing can include cables and require measurement of both radiated and conducted energy. Different standards of performance and methods of certification may apply based on application and equipment type.
- Ionizing radiation. While cameras do not emit ionizing radiation, they may constitute part of the shielding in radiation imaging systems or they may form part of an imaging chain that must meet certain performance requirements. Testing in shielding applications is nearly universal but image quality verification usually only applies to medical systems.
- Design and production control. Most medical equipment must be designed and manufactured under written control procedures certified by a third party or regulatory agency. Records must be kept for extended times and, in some cases, components and even materials must be traceable to the origin. Personnel training is generally required and even marketing literature may be subject to vetting of any claims of efficacy they state.
- Space and defence. Equipment supplied for space or defence purposes may be subject to stringent specific test requirements, third-party certification and inspection by the buyer. For purchasers of cameras, the requirements of flow-downs of this type from customers must be completely understood and specifically addressed.
- Quality systems. Many customers now require vendors to be certified to ISO9001:2008 or to demonstrate existence of a comparable quality management system. This may require that each vendor in a supply chain require the next vendor up also to be certified. Other certifications, such as ISO14000 or industry standards such as ISO/TS 16949:2002 (for automotive suppliers) may also apply.

If any of these regulatory requirements appear to apply, the vendor must take responsibility to determine applicability and assure compliance. Contracting qualified experts is sometimes necessary.

10.6.4 Block Diagram

The purpose of a camera is straightforward: convert a flat optical image into an analogue or digital signal containing the image information. However, there are myriad details of this conversion process to consider when selecting a camera. Each of the steps in this process must have a particular set of controls and ranges and formats. Comparing these with the requirements of an application is often tedious on account of the divergence of the names and operation of various functions. Those presented here are in common use but cannot be considered universal.

A representative camera block diagram is shown in 38 The organization and operation of particular cameras can vary substantially but this diagram can serve as a starting point for understanding and comparing them. The blocks should not be taken as descriptive of individual parts or boards in any camera. In many designs, especially with CMOS sensors, all of the control, processing and interface functions may be contained in a single FPGA. It is worthwhile, however, to understand how the functions are implemented because the choice of parts may have an influence on reliability, maintenance and updating of cameras in the field.

10.6.4.1 Microcontroller

Except in the very simplest cameras, setup and operation are managed by a microcontroller, whether as a separate component or as an IP core realized inside an FPGA. The microcontroller is itself controlled via a typically EIA-232 serial, CameraLink, USB or IEEE 1394. Responsibilities of the microcontroller can include:

- Managing the external communications port.
- Conveying control data to registers in the sensor or to a register set in an FPGA.
- Monitoring the operation of the camera – determining its health and taking action or reporting.
- Executing commands not requiring real-time synchronization.
- Calculating camera settings based on commands and status.
- Sequencing turn-on or mode changes.
- Loading RAM-based FPGAs.
- Storing settings for power-on states or rapid configuration switching.
- Illuminating status lights.
- Running general-purpose software.
- Doing essentially anything else a controller can do.

If a camera has a microcontroller, it also has firmware. Firmware may need to be updated, so it is wise to know what is required to implement updates. In some cameras, updates can be applied through the control port. Others have a special connector to which a programming board must be attached. Still others require return to the factory. If updates are a significant concern, the update method should be understood.

10.6.4.2 Timing Generator

All timing for a camera is usually is generated by a timing generator built inside an FPGA or an integrated timing circuit designed for a specific sensor. Typical timing waveforms provided by the timing generator include:

- Accurately timed pulses to operate all sensor functions. In some CMOS sensors part of the timing may be generated inside the sensor and so the inputs may be a mixture of waveforms and clocks. CCDs may require separate power drivers for the transport registers. Sensor may also have additional inputs for mode switching that are not time-sensitive.
- Sampling pulses for analogue to digital conversion. Some sensors have internal ADCs but even those may be externally timed. Some ADC chips have serial outputs so the FPGA may need to provide a very high speed clock to run a deserializer.
- Clocks for real-time processing. Some processing in cameras must be done in synchronism with scanning to preserve real-time transmission of image data. The clocking needed could be for a high-speed ALU, for buffer or frame-storage memory or for clocking of an auxiliary processor with its own functions.
- Special waveforms to control data formatting and transmitting. Each external camera data interface has particular data formatting and timing requirements. Real-time interfaces like parallel LVDS and CameraLink require provision of low-skew data properly formatted for the transmission arrangement desired. Computer interfaces like Ethernet, USB and IEEE1394 require proper data preparation and transmission on request so data may have to be separately buffered in a FIFO controlled by the timing generator. Cameras with analogue outputs will need pulses tailored to insertion of sync and blanking.
- Waveforms to re-time external signals. Certain functions like external synchronization or triggering require retiming of the incoming signals to correspond with the camera timing. Rarely is it desirable or even possible to use external time bases to operate phase-locked loops to bring multiple cameras into synchronization. Retiming of image data from multiple sources is almost always better accomplished after data reception even if the exposures are closely synchronized by external pulses. The sole exception is the use of multiple sensors in a single camera as in prism assemblies or some other special configurations.

The microcontroller may have its own clock even if it is contained inside an FPGA that has other functions run by the timing generator.

10.6.4.3 Data Formatter and Interface

Before transmission, data must be put in the proper form to be compatible with the chosen data link. Most interfaces have special devices available to provide that last step in formatting. CameraLink has special parallel to serial converters and each of the standard computer interfaces has dedicated devices. Similar devices deconstruct the image data on the receiving end so that the receiving device can properly arrange the data for further use. Some links require addition of block or other data control information but these functions are performed by the dedicated chips.

10.6.4.4 External Controls

Every camera has a set of operations that can be externally controlled. In the simplest cameras, these may only be some mode-switching and on-off controls for a few automatic functions. Complex cameras can have literally hundreds of functions. The exact from of these functions will depend on the type of camera interface used and the actual commands used to communicate controls to the camera may never be seen by the user unless custom control software is needed. It is worth noting that if a camera is used with a framegrabber, the software path used to communicate with the camera is the framegrabber driver, not a separate camera driver. If the camera uses a standard computer interface, a special camera driver is likely required. In those cases, someone thoroughly familiar with drivers should be involved in camera selection to assure that the required functionality and compatibility with user software is achieved. Here are some typical controls arranged by category.

Setup

Commands are generally provided to allow control of the link between the camera and the computer. The nature of these depends heavily on the link chosen. In CameraLink, for instance, the mode (base, medium or full) may be selected. Baud rate and parity may be set for serial links. Commands may also be supplied for handling of error messages, verification of command receipt and query of the camera status. It is important in designing link control commands to assure that the link cannot be put in a mode that causes the camera to be unable to receive commands.

Commands to initiate a general reset to a known good state, to report camera hardware and firmware configuration, to report camera operating status and to set system monitor modes for upgrades are also often useful.

Power

External control of camera power is generally limited to putting the camera into and out of a standby mode. The actual effect internally in any camera will depend on the sensor and the electronics configuration. Power-down is useful to conserve power when the camera is not in use and to extend the life of parts that may be temperature sensitive. In practice, it is necessary to know how long a camera will take to stabilize electrically and thermally after the power is restored and what the starting state of the camera will be. Some cameras will accept configuration commands during power down and some will not. As a further complication, some cameras cannot respond to certain commands when in power down mode because access to sensor registers is blocked. It is recommended that a complete set of commands be sent on power up or that the camera be queried for all control settings to assure that the camera is in a known state before image acquisition commences.

Scan

While many CCD cameras permit some control over scan configuration, the advent of CMOS sensors has greatly expanded the possibility for control. The list of commands below is typical of a CMOS camera with a fairly broad control repertoire. Other commands could be unavailable or added depending on the sensor capabilities. Some cameras also have composite commands, which are not listed here. Typical of these are preset scan modes that could instruct the sensor to scan a particular matrix at a particular frame rate. Other, even more complex, preprogrammed modes may be available. It is worth asking any camera vendor if some special mode needed for a particular application can be made available by command.

- Size – to set the number of vertical and horizontal matrix locations to be read out
- Position – to set the starting points for the scan
- Pitch – to set the number of elements between each scanned point and the next
- Binning – to set the number of adjacent elements to be combined into one output sample
- Test – commands transmission of a synthetic test image

Acquisition

Acquisition of images can be repetitive or one-shot controlled by internal timing, software or external triggering. Acquisition will always be subject to the sensor timing rules, discussed above, and any additional constraints imposed by the camera or communications link.

- Trigger – to set the source of acquisition initiation; internal, external or software command
- Exposure – to set the exposure time or to enable external pulse control of exposure
- Mode – to select among the available exposure modes; rolling or global, for instance

Point Processing

In point processing operations, each sample is modified based only on its own contents and using no data from any other sample. Point processing operations are by nature monochrome, even in colour cameras. These are applied to each colour channel individually. The content of one colour channel does not affect the operation on another channel.

- Gain – sets the signal output level for a particular input relative to some reference level
- Offset – sets the shift in output for a particular input relative to some reference level
- Linearity – sets the curve of output vs. input; can be gamma, log or other function
- Clipping – sets the level above which a further increase in input causes no increase in output

Neighbourhood Processing

In neighbourhood processing, the output for each sample depends on the signal in nearby samples. Typically, the neighbourhoods are on the order of 3×3 or 5×5 but much larger neighbourhoods can be used for complex tasks. Bayer demosaicing, for instance can take data from up to 10 samples distant. Colour space transformation is a neighbourhood process because data from all three colour planes are used to modify all three colour planes. Many more processes are available than are listed here but most would not be implemented in a camera.

- Sharpen – sets parameters for any of a number of edge enhancement algorithms
- Blur – sets parameters for any of a number of smoothing algorithms
- Filter – sets parameters for nonlinear filtering such as with a median filter
- Colour space – sets the nine matrix parameters for a linear colour space transformation
- Demosaic – set the parameters for Bayer or other CFA demosaicing
- Encode – sets parameters for real-time image coding; run-length, wavelet or encryption, for instance

Temporal Processing

In temporal processing, samples are recomputed based on other samples taken at a different time. These previous samples may be from immediately preceding scans or from stored image data. Invoking temporal processing necessarily slows the acquisition process and may cause unwanted image data to persist from one scan to another.

- Integrate – sets parameters for accumulation of multiple scans so the sum may be read out
- Average – sets parameters for the averaging of multiple scans
- Smooth – sets parameters for computing of a weighted moving average of continuous scans
- Subtract – subtracts a stored image (usually dark) from all incoming scans
- Filter – sets parameters for non-linear inter-scan processing; limiting scan to scan changes, for instance

Accessories

In cameras with nearby accessories such as lighting, shutters, coolers, filter wheels, motorized lenses and similar items, including provisions for accessory control can save extra wiring and simplify computer interfacing. Such cameras will need some sort of accessory connector to pass on such commands. In some cases, the camera firmware may also need to reformat the commands for use by the accessories.

10.6.4.5 Analogue to Digital Conversion

Whether A/D conversion takes place in the sensor, in the camera or in the framegrabber, this interface must be properly utilized for maximum image quality. Complete discussions of A/D conversion can be found in several items in the recommended reading list. Here we include only a few important rules of thumb:

- Bit terminology – MSB is the bit that requires the largest change in input signal to change state. LSB sets the smallest change in signal that can be recognized digitally.
- Gain setting – to capture the full dynamic range of a sensor, allow about 10% headroom between the largest possible sensor signal and the maximum input signal.
- Dynamic range – DR of an ADC is 6.021 db (20 log 2) per bit. The ADC in a camera needs at least four more bits of dynamic range than the dynamic range of the sensor if images are going to be processed temporally in order to avoid generation of artefacts from ADC non-linearity during, summation, averaging or subtraction.
- SNR – The SNR of an ideal ADC when digitizing a sine wave is improved by 6.021 db for each additional bit. However, it should be assumed that this is never achieved in a real ADC.
- Offset – When the signal from a sensor is zero, the noise can go below zero so some offset must be provided at the input to the ADC to avoid noise clipping. The amount needed will depend on the relative levels of the various noise sources.
- Low-level signals – When low-level signals are to be digitized, add gain before the ADC process to preserve maximum ability to apply noise reduction algorithms. Adding gain after digitizing is simply shifting the bit position of the data leaving the lower bits empty. Information lost before digitization cannot later be reclaimed.
- Sampling – Make sure that the output signal from the sensor is clean during the interval that it is sampled by the ADC. Use the longest sample window possible.

10.6.4.6 Smart Cameras

Cameras must no longer be dumb transducers. The rapid increase in computing power available in small packages has led to moving some feature recognition and even decision-making functions to cameras. For systems using lower-speed connections between the camera and host, it is essential to minimize the amount of camera data carried over the link, especially if multiple cameras are to be assigned to a single computer port. Incorporating data-reduction is straightforward as long as the reduction ratios needed are small. Run-length encoding, for instance can provide lossless reductions around 2× as can discrete cosine transforms. If larger reductions are required, more complex coding can be applied but these are always lossy to some degree. Unfortunately in applications, the loss of data may make it difficult for the computer to make an accurate decision or measurement. In addition, high-ratio compression methods tend to involve block encoding that can introduce artefacts into the image that could be mistaken for edges in the object.

Smart cameras have substantial capabilities for detection and measurement as shown below but care must be taken to avoid running smart cameras at their limits of computation or giving them tasks that require frequent parameter adjustment, For example, a smart camera may be able to detect an measure two objects simultaneously but run out of power when presented with four. These limitations must be clearly understood before deployment and managed in case of failure by assuring that the camera can signal the host when it detects that it has failed its task. In situations where the product on a conveyor, for instance, changes often, changing the detection algorithms to match may be more effectively managed by keeping at least the highest-level software operations in the host rather than attempting to reprogram the smart camera for each different product.

Camera Structure

Smart cameras are similar in structure to the standard camera block diagram except that the processing block will include additional steps for feature extraction and measurement and the data formatter will not have to handle video-rate information. In addition, most smart cameras, except those preprogrammed for a specific function, require the microcontroller to run operating system software to provide programmability for the user. Such cameras generally operate in conjunction with a controller that can send and receive data from the camera, support programming and provide a conduit to a host computer for more complex coding using standard computer languages. To reduce the image data to features and, in some smart cameras, to provide decisions, requires that the camera take on some of the analysis capabilities formerly reserved for the host computer.

Feature Location

The simplest features to recognize are high-contrast edges. Sometime, simple thresholding can find them but more often, one of the two-dimensional edge-finding filters is used. Of course, the filter can only produce an image of the edge, not tell the camera where it is in the image so additional steps are needed to turn the edge into a real line and then to locate the line in the image. Alternatively, a grey-level edge-tracing algorithm might be implemented. These can provide location information directly but are more subject to upset from local variations than edge-finding filters. Other features such as holes, slots, notches and the like can be similarly detected and located. The location data can then be supplied to a host for analysis and action. This sort of scheme is applied often where the dimensional data must be combined from multiple locations that can't be viewed with a single camera or for special functions like stereo viewing or tracking in a volume.

Dimensional Measurement

From the location data, distances can be calculated for comparison with a standard to produce an accept/reject action or for transmission to a machine which might act by moving another part into alignment with a located feature. Depending on the capabilities of the programming tools supplied, the measurement capabilities of smart cameras can be quite sophisticated. A typically complex task might be checking the pitch and depth of a thread or verifying the spacing of a matrix of holes. It must be mentioned (even though this is not the optics section) that many precise dimensional measurement tasks require telecentric optics. Some smart cameras are designed to permit programming for a particular measurement task –perhaps measuring the diameter of a hole – and then retaining this program to act as if it were a dumb hole gauge.

Character Recognition

One common function performed by preprogrammed smart cameras is reading of alphanumeric characters or bar codes. Character reading is relative easy if the characters are well-formed and optimized for automatic reading like those on checks but reading characters stamped into metal is substantially more difficult. Nevertheless, this task is now routinely performed by smart cameras as are reading other types of characters on everything from warning signs to microchips. Barcode reading, too, is common. Smart cameras can read both one and two-dimensional codes, now including those with information coded in colour, and put out character strings representing the code content. Due to the variety of barcodes used, a typical camera designed for this task will have software controls to set the type of barcode to be read.

Object Recognition

Using blob analysis, in which a perimeter is drawn around an object in an image to provide dimensions for various ratios, individual cases of previously defined objects can be found. General sets of rules can be made to find cars, for instance, to permit managing of traffic flow. With the current power available in small processors, cameras can include these blob analysis and object detection capabilities. Programming these cameras requires the use of external tools but, once programmed, the cameras can run on their own. Often, this sort of camera will need to transmit to the host not only the detection information but a highly-compressed image with overlays to indicate what objects have been found in the image.

Higher-level functions may also be provided in the camera such as the ability to track identified objects, that is, identify objects that persist from frame to frame, even if they move. More detailed tasks may also be included. In the car application, locating and reading license plates might also be required. This is now within the capabilities of smart cameras. All of these functions depend on identifying actual objects in the scene.

Grey-Scale Analysis

In addition to geometric operations, smart cameras can measure intensities, calculate from these measurements and perform higher-level operations based on these measurements and geometric data. The available functions include classification of objects based on grey-scale properties. This operation could be used to sort objects or to determine surface damage. With proper calibration, smart cameras can make absolute measurements of parameters like reflectance and transmittance and quantize surfaces for texturing and other treatments. The limits here are associated with the number of layers of processing that are required. At some point, a smart camera will simply not have enough speed or memory. These limits continually shift so more capability is always coming.

Colour Analysis

Accuracy of absolute colour measurements is limited in any camera, not just smart cameras by the various factors discussed in the colour section. However, smart cameras are used for sorting on the basis of colour in agricultural applications and other manufacturing processes where acceptance is coded in colour. As long as high precision is not required, smart cameras can also measure colour uniformity and check for colour changes in a production stream. Also within the capabilities are slightly higher level functions like observing thin films and converting the colour of these to thickness using known colour maps. Colour-based blob analysis is likely beyond the capabilities of smart cameras, at least for a while.

10.6.5 Control and Data Interfaces

Interfacing to cameras offers a variety of options ranging from simple analogue and parallel digital data channels, which have been in use for decades, to the latest versions of general-purpose networking protocols. All of these have seen some modifications to handle the typically large data sets associated with images and, especially, with image streams. It is essential in selection of a camera to consider its capacity to provide images with the resolution, data depth and speed required for satisfactory application performance. In all links, some proportion of the time is used for signal or link overhead, more so with some links that others. Several other issues arise in link comparisons:

- Latency – The delay from acquisition of an image to receipt of data can be quite different for various links. Consider that if a strobe shot of an object is taken, the delay out of the camera for the last element is already more than one frame time. If some sort of temporal processing is used in the camera, this delay can be extended to two or three frames. Some links require data buffering that can further increase latency. If isolated imaged apart in time are suitable then none of this may be of concern but if the camera is in some sort of control loop, such delays can prevent successful system operation.
- Speed – Links based on direct wires from the camera to the host – called point-to-point connections – are generally only limited in speed by cable type and length. Protocol-based links can offer the same speed in the physical medium but much of the time is taken by overhead so the maximum camera data rates are reduced. For higher data rates, the properties of copper transmission lines limit cable length so in some cases fibre optic links are needed.
- Link support – Each camera requires support for its communication link from the host. For high-speed parallel links, this generally means that the data receiver must be a framegrabber. Link types that are commonly found on computers should not require additional hardware for support but in some cases the burden on the host CPU for the support of these links is excessive and additional hardware has to be added just to offload link service.

- Cabling and connectors – Parallel links tend to have larger, thicker, shorter and more expensive cables than protocol-based links. For some applications, the best compromise is a link in which the parallel camera data is reformatted for transmission on a smaller number of higher-speed serial data lines. Most of the computer links have fairly fragile connectors and low-strength connector retention to facilitate ease of insertion and removal. Using special versions of these connectors with some sort of retention lock or providing a separate retention device is recommended. In some cases, the connector entries may need to be sealed to keep foreign materials from entering the camera.

10.6.5.1 Analogue

Analogue signals actually convey a large amount of image data on a single wire but the susceptibility to signal degradation and the difficulty in providing sufficiently accurate inter-camera timing have nearly eliminated their use in industrial systems. However, some long-standing analogue interfaces based on television standards are still in use. EIA-170 defines the 525-line interlaced scanning use in NTSC video. It is still often implemented in cameras using chips design for NTSC resolution or VGA sensors. Similar chips are available for 625-line CCIR imaging. Camera tubes were operated at higher resolutions using the EIA-343A standard but this is rare in cameras with solid-state sensors.

In analogue cameras, the cabling is generally 75-ohm RG-59/U coaxial cable with BNC connectors. Runs to several hundred feet are feasible as long as electrical interference is minimal. Framegrabbers are used to digitize the video signal for computer use. Some cameras provide sync or horizontal and vertical drive signals and possibly a pixel clock to assure accurate detection of the time base by the framegrabber.

10.6.5.2 Digital

Digital interfaces offer various degrees of speed, ease of use, immunity to noise and link cost. In some cases, the tradeoffs are quite stark so understanding the specific limitations of each link is very important. For each of the links described, the salient issues are presented.

Parallel (AIA & Others)

Factor	Discussion
Architecture	One conductor for each bit of data transmitted plus additional conductors for timing signals
Speed	Up to 400 MHz per conductor using LVDS differential drivers and receivers
Support	A framegrabber is required for computer interface
Cabling	Multiconductor shielded twisted pair. Connectors of many types have been used
Concerns	Big, stiff cables. Distance limited to about 10 m. Potential skew errors
Documents	BSR/AIA A15.08/3-1993 (Automated Imaging Association)

CameraLink

Factor	Discussion
Architecture	A set of conductors, one for each serial stream of data transmitted plus additional conductors for timing signals. Optional power conductors. Three levels of implementation depending on number and width of data streams required
Speed	Up to 85 MHz pixel clock, up to 8 8-bit simultaneous data streams
Support	A framegrabber is required for computer interface
Cabling	1, 2 or 3 26-conductor shielded twisted pair. Connectors are MDR-26. Fibre optic extenders are available for links to 1 km
Concerns	Distance limited to 10 m
Documents	CameraLink® Specification Version 1.2 (AIA)

Ethernet

The particular version of Ethernet used for imaging is called GigE Vision™. The GigE Vision standard imposes certain constraints to assure availability of the data channel. Even so, about 10% of the channel capacity is taken by link management overhead. For cameras, Ethernet is almost always used as a point-to-point connection with each camera having its own port in the computer. This avoids camera data transmission request conflicts.

Factor	Discussion
Architecture	A four-conductor cable with the Ethernet protocol. Optional power
Speed	About 900 Mbps on a dedicated gigabit Ethernet link
Support	Can run on Ethernet computer port but an Ethernet card with its own protocol manager is recommended to free up the CPU. Communications may managed under the GenICam programming interface
Cabling	Cat5e or Cat6, up to 100 m. RJ-45 connectors
Concerns	No real-time triggering. Can have lost data
Documents	GigE Vision standard – AIA
	GenICam standards – European Machine Vision Association

USB 2.0

Factor	Discussion
Architecture	A four-conductor cable connected to a computer port. Provides 2.5 W power
Speed	Up to 300 Mbps on a 480 Mbps link (USB 3.0, under development – up to 4.5 Gbps available bandwidth)
Support	USB 2.0 port with custom drivers for each device

Cabling	Standard USB cables qualified for USB 2.0. Fibre optic bridges are available for links to 1 km. Standard USB A and B connectors
Concerns	Low data rate. Distance limited to 5 m. No real-time triggering
Documents	USB 2.0 Specification – USB Implementers Forum, Inc. USB 3.0 Specification – USBIF

IEEE 1394b

Factor	Discussion
Architecture	Nine-conductor cable connected to a computer port
Speed	About 500 Mbps on a 786 Mbps link. Faster implementations are allowed but uncommon
Support	FireWire 800 computer port. IIDC protocol is often used for communication
Cabling	Standard FireWire cable. FireWire beta connector. Fibre optic bridges are available for links to 1 km
Concerns	Distance limited to 4.5 m. Various speed levels of FireWire incompatible. No real-time triggering
Documents	IEEE 1394b-2002

Graphics Links

The HDMI graphics interface is very similar to CameraLink in that parallel data is multiplexed and serialized for transmission. Currently, HDMI ports are supplied only as outputs on computers but framegrabbers to accept camera data in this format are available. Since HDMI implements HDCP content control flags, it is important to assure that any camera made with an HDMI output does not assert these flags.

Factor	Discussion
Architecture	19-conductor cable connected to a computer port
Speed	Up to 8.16 Gbps or 340 Megapixels per second with 24 bits per pixel (colour). Triggering can be sent using audio channels
Support	Requires a framegrabber
Cabling	HDMI cable. HDMI connector
Concerns	Distance limited to 15 m
Documents	HDMI Specification 1.3 – HDMI Licensing, LLC

10.6.5.3 Industrial Nets

Smart cameras which are configured to accept simple setup and control commands and put out decision information are capable of being incorporated into industrial control networks.

This market is just beginning to develop as many industrial nets are abandoning proprietary architectures and switching ti Ethernet. To accommodate use in industrial environments, cameras have been built with isolated digital I/O ports for direct connection to programmable controllers, with RS-232 ports for instrument control and with 10 or 100 mbps Ethernet for network setup and monitoring. Cameras utilizing industrial protocols like MODBUS and others are also available. As smart cameras become more and more able to perform vision functions without attention, the industrial appliance models can be expected to increase.

10.7 More Information

Topic							
Sensors	Colour	X-ray	UV & IR	Cameras	Applications	Type	Title
X						Books	Prof. Dr. Albert JP Theuwissen, Solid-state imaging with charge-coupled devices
		X					X-ray data booklet. http://xdb.lbl.gov/
X		X	X				RCA Staff (1974) RCA electro-optics handbook (EOH-11)
		X			X		The Physics of Medical X-ray Imaging, 2nd Edition; Bruce H. Hasegawa
	X					Papers	Prism-based colour separation for professional digital photography. http://www.dicklyon.com/tech/Photography/PICS2000-Lyon.pdf
X							A brief history of "Pixel". http://www.dicklyon.com/tech/Photography/Pixel-SPIE06-Lyon.pdf
	X						Adrian F, Alan R, Colour space conversions. http://www.poynton.com/PDFs/coloureq.pdf
X	X	X					Norton P, Detector focal plane array technology. http://www.nomic.net/~tplagge/norton-arrays.pdf

X							Hynecek J, Impactron – a new solid state image intensifier. http://www.imagesensors.org/Walter%20Kosonocky%20Award/2003%20Hynecek.pdf
X			X				Norton P, HgCdTe infrared detectors. http://www.wat.edu.pl/review/optor/10(3)159.pdf
							Baker et al., Performance of image intensifiers in radiographic systems. http://www.osti.gov/bridge/servlets/purl/750414-oGzvNf/webviewable/750414.pdf
X						Articles	El Gamal A, Eltoukhy H, CMOS imaging sensors. http://isl.stanford.edu/groups/elgamal/abbas_publications/M001.pdf
			X	X	X		Richards A, UV imaging opens new applications. http://www.vision-systems.com/articles/article_display.html?id=259249
X							Janesick J, Dueling detectors. http://spie.org/x25373.xml?highlight=x2410&ArticleID=x25373
X	X					Presentations	The history and future of colour photography, http://www.dicklyon.com/tech/Photography/Plenary_Lyon_ICIP03.pdf
X							El Gamal A, High dynamic range image sensors. http://www-isl.stanford.edu/~abbas/group/papers_and_pub/isscc02_tutorial.pdf
				X	X		Scholles M, Industrial applications of IEEE 1394. http://www.1394ta.org/developers/Seminars/2008_IndustrialApp.pdf

X		X		X			Hornsey R, Digital imaging electronics. http://www.ece.uwaterloo.ca/~ece434/Winter2008/Imaging.pdf
X							Hornsey R, Fabrication technology and pixel design. http://www.ece.uwaterloo.ca/~ece434/Winter2008/Imaging2.pdf
X							Hornsey R, Noise in image sensors. http://www.ece.uwaterloo.ca/~ece434/Winter2008/Noise.pdf
		X				Websites	X-ray interaction with matter, http://henke.lbl.gov/optical_constants/
X				X	X		http://www.advancedimagingpro.com/
X	X	X	X	X	X		http://www.photonics.com/
X	X			X	X		Molecular expressions. http://micro.magnet.fsu.edu/primer/index.html
	X						Colour data tables. http://cvrl.ioo.ucl.ac.uk/index.htm
	X						Munsell Colour Science Laboratory, http://www.cis.rit.edu/mcsl/
X	X						Sensors, pixels and image sizes. http://www.shortcourses.com/sensors/index.html
X	X						"RGB "Bayer" colour and microlenses. http://www.siliconimaging.com/RGB%20Bayer.htm
					X		High resolution test patterns. http://www.bealecorner.org/red/test-patterns/
	X						Wratten filter data. http://www.mat.uc.pt/~rps/photos/filter-data.html

		X			X		Physics and technology of medical imaging. http://www.sprawls.org/resources/
							CCD-cameras for X-ray investigations. http://www.proscan.de/ber_1.htm
		X					X-ray mass attenuation coefficients. http://physics.nist.gov/PhysRefData/XrayMassCoef/tab4.html
				X	X	Magazines	Vision Systems Design. http://www.vision-systems.com/
X	X	X	X	X	X		Photonics Spectra. http://www.photonics.com/Content/Default.aspx?P=5&V=22&I=159
X				X	X		Advanced Imaging. http://www.advancedimagingpro.com/publication/pub.jsp
X				X			Optics and Laser Europe. http://optics.org/cws/Ole/Welcome.do
						Journals	J Electron Imaging. http://spie.org/x868.xml
							Opt Eng. http://spie.org/x867.xml
X	X	X	X	X	X	Trade Shows	SPIE conferences. http://spie.org/x306.xml
				X	X		AIA conferences. http://machinevisiononline.org/public/calendar/
				X	X		Robots, vision and motions control. http://www.robots-vision-show.info/robots_vision_show_info.html
	X					White Papers	Poynton C, Colour FAQ. http://www.poynton.com/ColorFAQ.html

			X				Hamamatsu Photonics, Characteristics and use of infrared detectors. http://sales.hamamatsu.com/assets/applications/SSD/infrared_kird9001e03.pdf
				X		Standards	CameraLink specification. http://www.machinevisiononline.org/public/articles/index.cfm?cat=129
				X			GigE vision specification. http://www.machinevisiononline.org/public/articles/index.cfm?cat=167
				X			GenICam specification. http://www.genicam.org/
				X			IIDC 1394-based digital camera specification ver.1.31. http://www.1394ta.org/developers/specifications/2003017.html
				X			High Definition Multimedia Interface, Specification version 1.3. http://www.hdmi.org/manufacturer/specification.aspx
				X			Universal Serial Bus, Revision 3.0 specification. http://www.usb.org/developers/docs/usb_30_spec.zip
X	X	X	X	X	X	Organizations	SPIE, http://spie.org/
				X	X		AIA, http://www.machinevisiononline.org/
				X	X		EMVA, http://www.emva.org/
		X			X		ASNT, http://www.asnt.org/
X						Videos	Lyon RF, Pixels and me. http://video.google.com/videoplay?docid=−5655850487750051532

				X		Data Sheets	T-20 USAF 1951 chart standard layout product specifications. http://www.aig-imaging.com/mm5/PDF/USAF%201951%20Test%20Target%20T-20_v1-04.pdf

References

1. Coghill J, Digital imaging technology 101. http://www.eng.yale.edu/elab/eeng427/lectures/EENG427l07DigitalImaging.pdf. Accessed 12 Apr 2011
2. Wikipedia, Quantum efficiency. http://en.wikipedia.org/wiki/Quantum_efficiency. Accessed 12 Apr 2011
3. X-ray attenuation length. Center for X-Ray Optics Research, Lawrence Berkeley Laboratory. http://henke.lbl.gov/optical_constants/atten2.html. Accessed 22 Apr 2011
4. X-ray mass attenuation coefficients. National Institute of Standards and Technology (USA). http://physics.nist.gov/PhysRefData/XrayMassCoef/ComTab/cesium.html. Accessed 22 Apr 2011
5. Wikipedia, 1951 USAF resolution test chart. http://en.wikipedia.org/wiki/File:1951usaf_test_target.jpg. Accessed 22 Apr 2011
6. EIA Wikipedia, http://en.wikipedia.org/wiki/Optical_resolution. Accessed 22 Apr 2011
7. Wikipedia, Digital micromirror device. http://en.wikipedia.org/wiki/Digital_micromirror_device. Accessed 13 Apr 2011
8. Wikipedia, Digital light processing. http://en.wikipedia.org/wiki/Digital_Light_Processing. Accessed 13 Apr 2011
9. Wikipedia, http://en.wikipedia.org/wiki/Color_filter_array. Accessed 14 Apr 2011
10. European Machine Vision Association, Standard for measurement and presentation of specifications for machine vision sensors and cameras. http://www.emva.org/cms/index.php?idcat=26. Accessed 14 Apr 2011
11. Machine Vision Online, Additional machine vision standards. http://www.machinevisiononline.org/vision-standards-details.cfm?type=7. Accessed 14 Apr 2011

11 Selecting Cameras for Machine Vision

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_11,

11.1 Types of Cameras

Machine Vision requires a systems approach; there are many subtle interactions that involve widget transportation, lighting, optics, cameras, video-capture hardware, computer system architecture, algorithms and software. When beginning a design, application requirements should always be considered first, in order to build a broad conceptual framework. It is most unwise to select any element of a Machine Vision system without considering all of the others as well.

In this chapter, we explore some of the practical issues that should be considered when choosing a camera. There are many different types of cameras and there are many subtleties associated with the way they work. In addition, the camera is a sensitive device that must often operate in a hostile environment. In this chapter, we shall therefore attempt to remove the mystery surrounding this, the central component of a Machine Vision system.

11.1.1 Area Cameras

An *area camera* is any camera designed to image a two-dimensional field-of-view. CCTV (closed-circuit television) cameras historically were extensively used in Machine Vision but they have become much less common in new system designs, due to the emergence of low-cost, high-resolution, digital area cameras.

Although rapidly disappearing, CCTV cameras can still be found in many legacy Machine Vision systems. The technologies of progressive scan and the LVDS (low-voltage differential signalling) interface facilitated the transition from analogue television to modern digital cameras. Other area cameras are also available, such as webcams and those used for electronic conferencing, but they are generally not suited for Machine Vision. Since CCTV cameras, analogue progressive scan cameras and digital cameras with LVDS interfaces can still be found in Machine Vision systems, they are briefly described below.

Generally, Machine Vision area cameras can be characterized by the basic image sensor technology, the image resolution and the interface technology. Cameras can also be classified into colour and monochrome categories. Each of these is discussed in the material that follows.

11.1.1.1 CCTV Cameras

CCTV cameras conform to existing well-established standards such as RS170, RS330, NTSC, CCIR and PAL. These standards dictate the image aspect ratio, signal amplitude, scanning pattern and timing.

Most of the common video standards use interlaced scanning in which the complete image (frame) consists of two partial images or fields. The first partial image (odd field) contains scan lines 1, 3, 5, ... The second partial image (even field) contains scan lines 2, 4, 6, ... (➲ *Fig. 11.1*). The RS330 specification is unique in allowing non-interlaced operation, in which there is one field with half as many scan lines as would be present in interlaced operation. Interlaced scanning was developed specifically to reduce the bandwidth required for standard (terrestrial) television broadcasting. It exploits a physiological property of the human visual system that makes it possible to obtain the benefits of good spatial resolution without actually sending all of the data simultaneously. This is achieved at the cost of allowing a barely perceptible degradation in rapidly changing scenes.

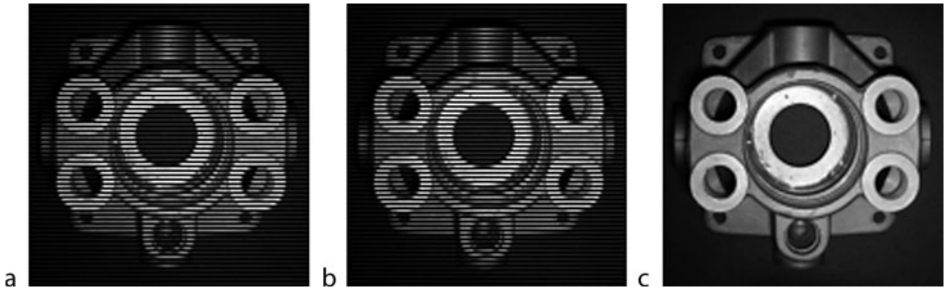


Fig. 11.1
Interlacing. (a) Field 1: odd-numbered lines. (b) Field 2: Even-numbered lines. (c) Full image

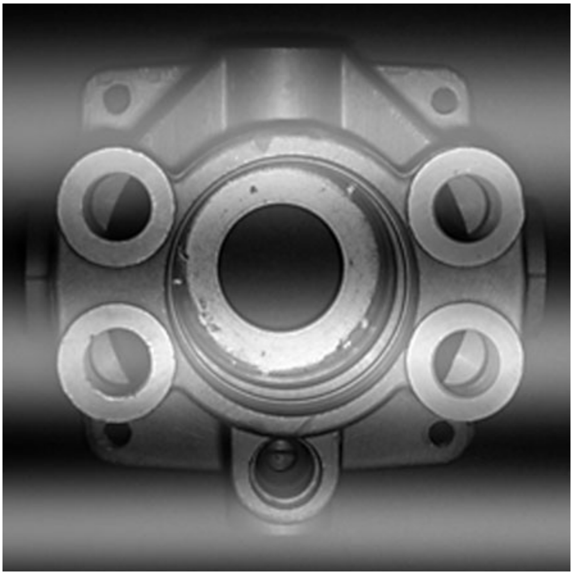


Fig. 11.2
Hum bars (simulated, to emphasize the effect)

Under the RS170 and NTSC standards, 30 frames/second (f/s) is used compared to 25 f/s for CCIR, and PAL. Note that these frame rates correspond to half the power-line frequency in the countries where they are most often used. Scan line timings are very similar for all of the standards. RS170 and NTSC standards provide about 480 active scan lines, while there are about 512 active scan lines for CCIR and PAL. Significant problems occur with traditional lighting, especially fluorescent or discharge-tube lighting, if the camera frame rate is not synchronous with the power-line supply. This can result in so-called hum bars in the camera video output (➋ *Fig. 11.2*).

For technical imaging applications including Machine Vision, interlacing caused problems when acquiring images of moving objects. Since the two fields are captured at different times, they will not be aligned properly resulting in a badly distorted image, called *image tear*, when objects are moving rapidly (➋ *Fig. 11.3*).

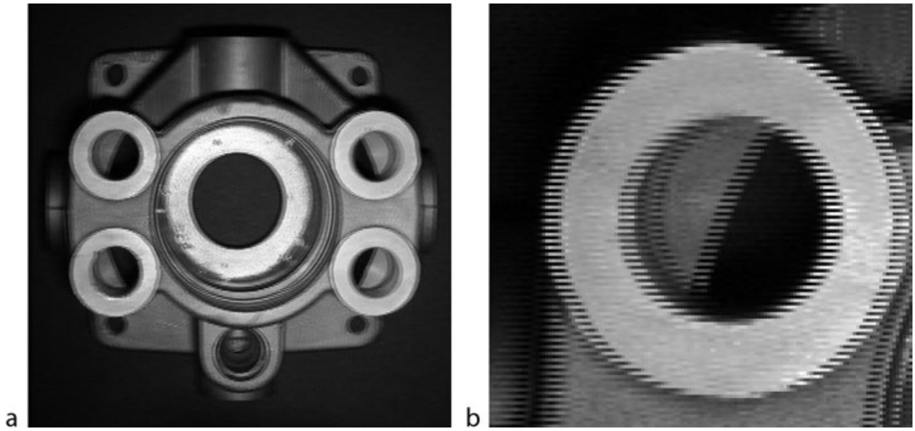


Fig. 11.3
Image tear (simulated, to enhance the effect). (a) Full image. (b) Detail

11.1.1.2 Progressive Scan Cameras

Manufacturers addressed the problem posed by interlaced scanning by developing progressive scan cameras where scan lines are read out sequentially. In some progressive scan cameras, the pixel data can only be read out sequentially in a non-interlaced format. Other cameras provide the option of either progressive or interlaced output. This gave the camera the ability to be compatible with legacy equipment while still retaining the benefits of progressive scan. With this type of camera, a standard TV monitor could be used for camera setup and then switched to progressive operation as required by a given application.

For applications using a strobe light for illumination and an interlaced CCTV camera, only a single field could be captured with a single flash, so the vertical resolution of the image was reduced by half. Even though the photo-sensors associated with the other interlace are exposed, the information is discarded when the first field was read out. With a progressive-scan camera, all scan lines acquire image data that can be read from the camera, and an image illuminated by a strobe had full resolution.

11.1.1.3 Aspect Ratio

At the dawn of television, the standard set the image aspect ratio, the ratio of image width to image height, to 4:3. This was a standard copied from the motion picture industry. This aspect ratio is still widely used today, but other aspect ratios are also common. The transition from analogue television to HDTV (high-definition television) started with an aspect ratio of 16:10 but eventually settled on the aspect ratio to 16:9. Photography had its own image aspect ratios, the most popular being that of 35 mm film; an aspect ratio of 3:2. There are non-standard image sensors that have aspect ratios ranging from 1:1 (square) to 25:1.

A given Machine Vision application will have a preferred aspect ratio based on the required field-of-view. Since custom image sensors are normally not practical, the application developer needs to choose the best aspect ratio available at reasonable cost.

Another consideration is the aspect ratio of pixels. This is determined by the centre-to-centre spacing of sensing elements on the image-sensing array in the camera. Most Machine Vision cameras have “square” pixels – the pixel spacing is the same horizontally and vertically. A few cameras do not have square pixels, and for those cameras, the image-processing software must be able to accept different calibration factors for the horizontal and vertical direction. Also, non-square pixels can make some image-processing algorithms more complex to implement.

11.1.1.4 Image Format

Back in the early days of CCTV, the most common image sensor was a vidicon image pickup tube. This was an enclosed cylindrical glass tube with a target on one end to sense the image, and pins for electrical contact on the other end. The image size was a function of the diameter of the glass tube, and the image format, a label for the image size was this diameter in inches and not the actual image size. This designation continues into the present for area cameras even though image pickup tubes have been replaced by solid-state image sensors.

Some of the common image format designations are:

Format (in.)	Image diagonal (mm)	Image width (mm)	Image height (mm)
1/4	4.50	3.60	2.70
1/3	6.00	4.80	3.60
1/2	8.00	6.40	4.80
1/1.8	8.93	7.18	5.32
2/3	11.00	8.80	6.60
1	16.00	12.80	9.60

For a camera's image sensor designation, the format is determined by the image diagonal and not by the height, width or aspect ratio.

11.1.1.5 Image Resolutions

As area cameras departed from the analogue CCTV standards, the image aspect ratio and the image resolution were no longer tightly constrained by specifications. Many different image resolutions and aspect ratios now exist; the most common ones are identified by their equivalent computer display resolutions. The table below lists some of the more common.

Image designation	Horizontal (pixels)	Vertical (pixels)	Aspect ratio
VGA	640	480	4:3
WVGA	752	480	16:10
SVGA	800	600	4:3
XGA	1,024	768	4:3
WXGA	1,280	720	16:9
HD_DVD			
UXGA	1,600	1,200	4:3
WUXGA	1,920	1,200	16:10

11.1.1.6 Monochrome Cameras

Monochrome cameras are sometimes called *black-and-white cameras,* which erroneously suggests that they are limited to sensing only black or white. In fact, this type of camera provides useful greyscale information as well. While the term *monochrome* implies a single colour, these cameras typically respond to the entire visible (VIS) spectrum and frequently beyond it as well. Silicon, the material from which most image sensors are made, is responsive over the spectral range of 300 (NUV) to 1,100 nm (NIR), with a peak sensitivity between 500 and 900 nm depending on the process used to fabricate the image sensor.

The spectral range of a camera can be further constrained by placing filters in front of the camera, or between the light source and object being viewed. In this way, certain features of an object are enhanced, while suppressing others (➋ *Fig. 11.4*). Similar advantages can be realized for wavelengths outside the visible spectrum. Many cameras employ an IR cut-off filter to limit the sensor's response to wavelengths shorter than about 750 nm (see ➋ Sect. 11.3.2).

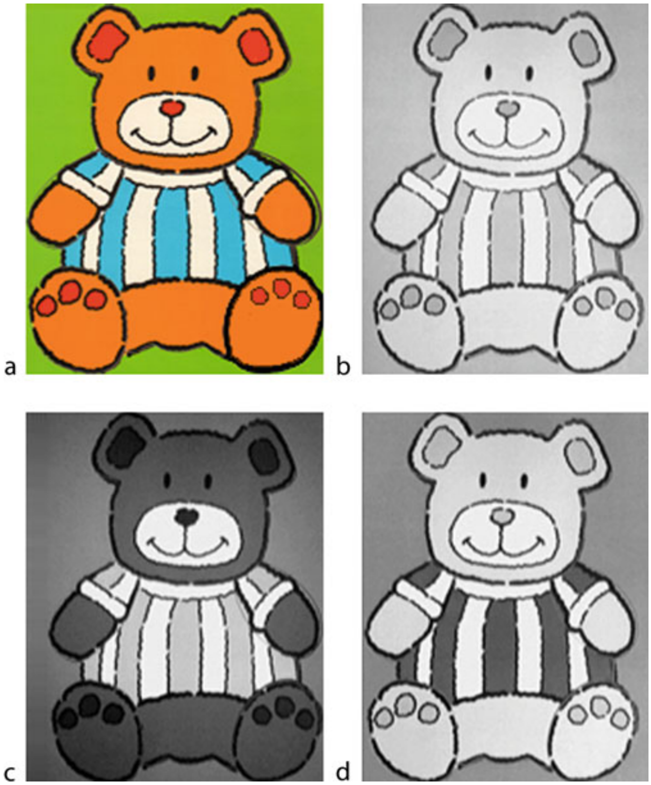


Fig. 11.4
Optical filters can greatly increase image contrast. (a) Original image. (This is a very simple printed jigsaw puzzle for young children.) (b) No filter, monochrome CCD sensor. (c) Short-pass filter (Cut-off: 550 nm). (d) Long-pass filter (Cut-off: 550 nm)

11.1.1.7 Colour Cameras

With colour cameras, there are two additional points to consider: what spatial resolution is provided, and how well the camera's colour response matches an established standard (➋ Chap. 4). Colour cameras measure light intensity in three different spectral bands at each pixel location. (These are commonly referred to as the *red, green* and *blue* colour channels. although this has been the source of considerable confusion. It is better to refer to them as the RGB channels.) The proportions of the three "primary colours" provide a measure of the colour and brightness of an object feature. When colour matching is important, both the camera and the illumination source must be carefully calibrated and checked regularly. Even with high-quality cameras, deficiencies in illumination can cause significant colour measurement errors. Some cameras provide compensation for different types of illumination (e.g., incandescent filament lamp, fluorescent tube, LED). This may help but resolving colour information at shorter wavelengths is difficult without wide-band illumination sources.

When colour television was under development (1950s and 1960s), advantage was taken of the relatively poor human visual perception in the red and blue portions compared to the central parts of the visible spectrum. Broadcast standards for encoding colour signals make use of the low spatial resolution at the ends of the visible spectrum to save bandwidth. This can, however, be detrimental to the performance of a Machine Vision system.

It is important to note that, given that RGB, sensing of colour is fundamental to nearly all colour cameras. Hence, there is no intrinsic advantage in converting to another format for representing colour, such as HSI (hue, saturation and intensity). For practical reasons, it may be convenient to do so, but information is not gained by calculating any other parameters. Although many physiologists and colour theorists complain that there are better ways of representing and describing colour, sensing based on spectral sampling cannot be avoided. RGB cameras provide the simplest, cheapest and most frequently used basis on which to base colour recognition and processing. Multispectral cameras exist but are expensive. The spectral response of a three-chip RGB camera can be modified, by the simple expedient of removing the standard RGB filters and then inserting different filters in the optical path.

11.1.1.8 Single-Chip Colour Cameras

Single-chip colour cameras are made by depositing colour filters over the sensing elements on an image sensor (➋ Chap. 10). The fidelity of colour reproduction depends to a large extent on how accurately these colour filters are applied in manufacturing. In general, the colour reproduction of these cameras is adequate for viewing scenes but is usually not sufficient for precise colour matching.

Since, at most, one out of three sensing elements will have a red filter over it, and more commonly only one out of four sensing elements has the red filter, the effective image resolution of the red-channel image will be substantially less than that of a monochrome camera with the same number of photo-detectors. The same is true for the blue channel. It is common practise to provide two green, one red and one blue photo-detector sites for each group of four sensory elements (➋ *Fig. 11.5*). Hence, the spatial resolutions of the red and blue sub-images typically are reduced by a factor of four, compared to an equivalent monochrome camera. Since the two green sub-images are normally merged, the resolution of this channel is

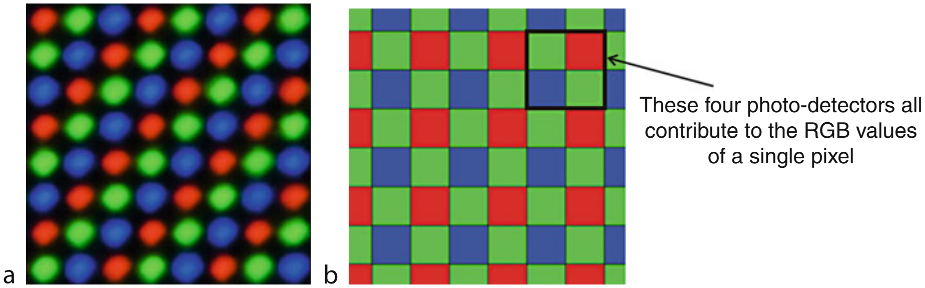

a
b
These four photo-detectors all contribute to the RGB values of a single pixel

Fig. 11.5
Bayer filter mask for a single-chip colour camera. (a) View through the mask. The unprocessed photo-detector outputs constitute a RAW image. (b) Converting the RAW image into RGB values. The outputs from two neighbouring green photo-detectors are averaged to form the G channel output, The R and B channels are derived directly from nearby red and green photo-detectors. Hence, there are twice as many photo-detectors contributing to the G channel as there are to the R and B channels. This reduces the appearance of noise effects. Moreover, the RGB photosensors, though very close together, are actually in different places

also reduced by a factor of four. Advantage is taken of the limited acuity of the human visual system in the red and blue portions of the spectrum by allocating fewer photo-detectors for the spectral extremities.

11.1.1.9 Three-Chip Colour Cameras

A three-chip colour camera uses three separate identical image sensors; an arrangement of dichroic beam splitters separates the optical image into three colour components, which are then projected onto the sensors (➔ *Fig. 11.6*). This type of camera produces three parallel video output channels corresponding to the red, green and blue portions of the spectrum. Each of these has the same full resolution as a monochrome camera using the sensor chip but is less sensitive than the monochrome sensor, since only a portion of the light energy reaches each sensor. This type of camera employs optical filters that are not integrated into the image sensor. As a result, more consistent and accurate colour measurements are possible compared to single-chip colour sensors. The response of each channel depends on that of the colour filter and the sensor chip, as well as the angle at which the light passes through the filter. Even with high-quality cameras, there is some unpredictability in its response, due to fabrication variations. Some sensors use the back of the sensor as the imaging surface, to avoid passivation layers and conductors that will alter the sensor's spectral response. This offers significant benefits for colour measurements.

The final consideration for accurate colour image acquisition is that the spectral band pass of an optical filter is specified for normally incident light rays. Because the optical path is longer for oblique rays, the filter response will be altered for these rays. This effect is most severe with wide-angle (short focal length) lenses since light at the edges of the image will be highly oblique. For most applications, it is not too significant. For maximum colour measurement accuracy, telecentric optics should be employed, since all rays are incident normally on the sensor.

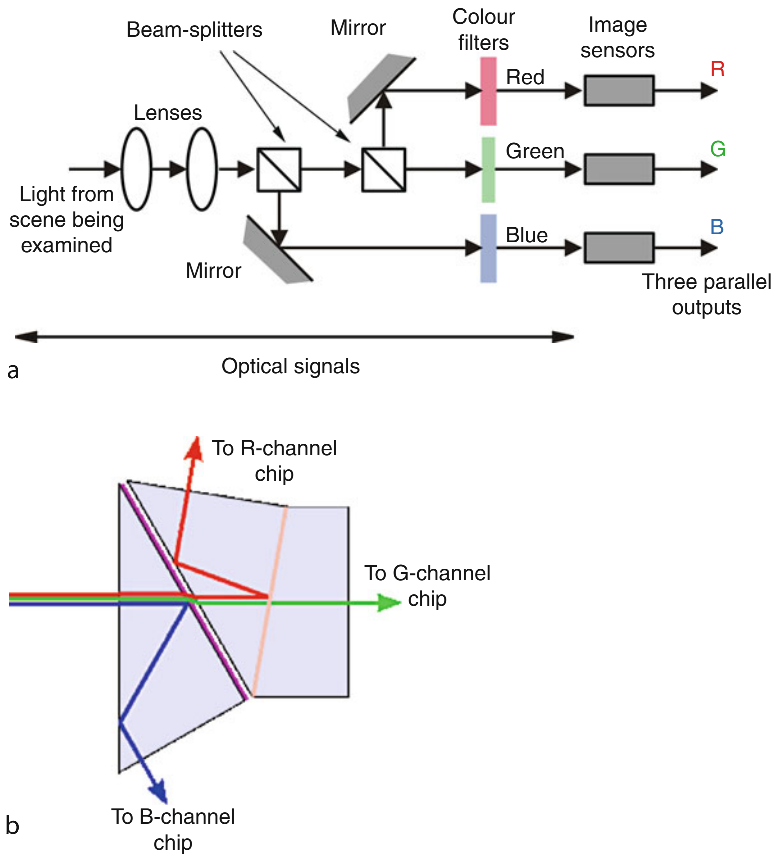


Fig. 11.6

Three-chip colour camera. (a) Using two beamsplitters. Notice that other colour discriminations can be achieved, by simply replacing the RGB filters. It is possible to perform fine colour separation, for example between different shades of "yellow," by using sharp cut-off (interference) filters. (b) Using a multi-element prism. This consists of three prisms with dichroic coatings cemented together

11.1.2 Line-Scan Cameras

The most commonly used line-scan camera consists of an image sensor with a single row of photo-detector elements. To acquire two-dimensional images, either the object under observation (the *widget*), or the camera/sensor must be moved. Through the acquisition of a series of scans, a two-dimensional image of the part is acquired (*Figs. 11.7* and *11.8*).

Many line-scan image sensors provide dual readout paths: one for odd-numbered pixels and one for even-numbered pixels. Some line-scan cameras merge the data from these two paths to form a single video output to facilitate ease of use. But, where speed is important, each of these paths is routed to its own video output. Even higher-speed line-scan cameras are available with multiple outputs.

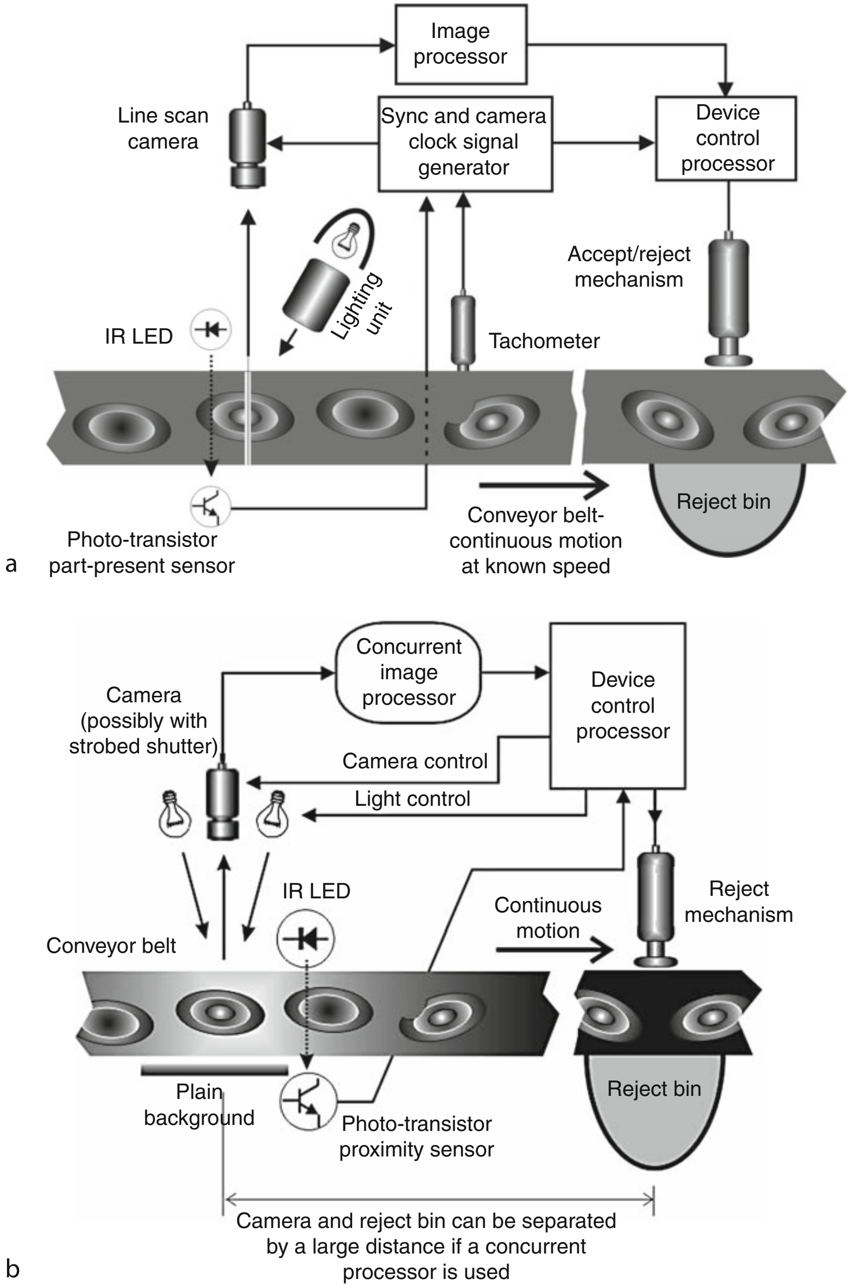


Image processor
Line scan camera
Sync and camera clock signal generator
Device control processor
Accept/reject mechanism
Lighting unit
IR LED
Tachometer
Reject bin
Photo-transistor part-present sensor
Conveyor belt-continuous motion at known speed
a
Concurrent image processor
Camera (possibly with strobed shutter)
Device control processor
Camera control
Light control
IR LED
Continuous motion
Reject mechanism
Conveyor belt
Reject bin
Plain background
Photo-transistor proximity sensor
Camera and reject bin can be separated by a large distance if a concurrent processor is used
b

Fig. 11.7

Using a line-scan camera to inspect objects on a continuously moving conveyor. The tachometer ensures that the camera clock and belt are synchronized. This is necessary, so that the image (and pixel) aspect ratio is known precisely. The image acquisition and processing are triggered when the object being examined breaks the IR beam of the part-present detector. (a) Area camera, (b) Line-scan camera

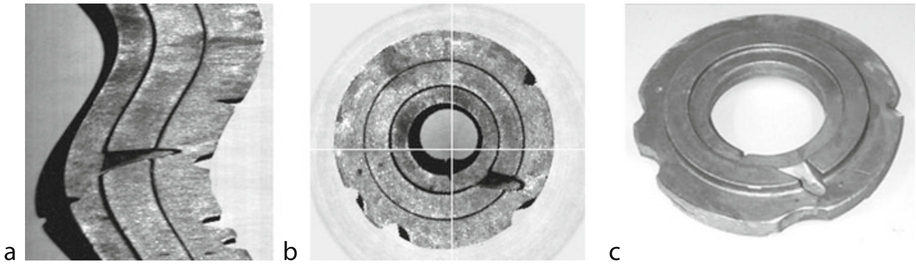


Fig. 11.8
Using a line-scan camera to inspect objects on a turntable, rather than a linear conveyor. (a) Image generated. (b) After Cartesian-to-polar coordinate mapping. (This "bends" the image so that it restores the geometry of the original object.) The crossed lines indicate the centre of rotation. (c) Photograph of the object

For high-speed image acquisition with line-scan cameras, very intense illumination is required due to the short integration ("exposure") time. An area camera's integration time is typically about 17 ms but a line-scan camera frequently may have a scan rate in the range of 0.1–1 ms/line. To compensate for this reduction, a wide lens aperture can be used but this may introduce undesirable optical aberrations. For some applications, illumination devices such as LED arrays, fibre optics or linear fluorescent tubes can be used to concentrate light around the very narrow strip that the sensor "sees"; there is no point in "wasting" light by illuminating a larger region when only a line of light is needed. Indeed, the unwanted and unused light may well reduce the image contrast.

Line-scan image sensors used in Machine Vision almost always have square sensing elements – the height of the sensing element is equal to their horizontal spacing. There are some line-scan arrays, like those made for spectroscopy, where higher sensitivity is needed. Their height is many times greater than their horizontal spacing. These image sensors are rarely used in Machine Vision.

The effective pixel size on the object in line-scan imaging depends on the spacing of the pixels along the line-of-view for one image axis and the speed of the part past the camera together with the line scan rate for the orthogonal axis. Therefore, the effective pixel aspect ratio is application dependent.

Line-scan cameras are available with a range of line lengths in pixels. Some of the common ones are: 512, 1024, 2048, 4096 and 8192. Simple calculations relating to the operation of a standard line-scan camera are presented in ❯ Sect. 11.5 of this chapter.

11.1.3 TDI Sensors

In order to mitigate the short exposure of line-scan cameras, the *time-delay integration* (*TDI*) architecture was developed. A TDI sensor incorporates a number of parallel rows of photo-detectors, typically 64–100, on a single chip. At the end of each scan, the accumulated charge in each row is transferred to the adjacent row, where further photo-generated charge accumulates. This process continues until the charge pattern reaches the final row where it is read out. The output of a TDI sensor is essentially the same as the output from a line-scan sensor, except that

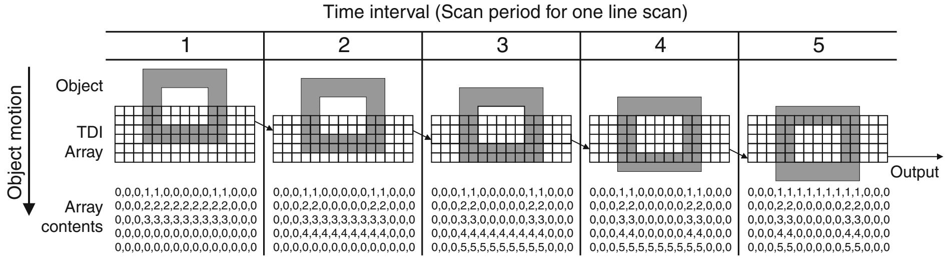


Fig. 11.9
Operation of a TDI camera

far less illumination is required. As for a conventional line-scan camera, the sensor chip or the widget must be moved at a constant, well-defined speed. For a TDI sensor, this motion must be carefully synchronized with the scan rate of the sensor, so that the same region of the widget is imaged on adjacent rows of the sensor at each scanning interval. Significant image blurring results, if the motion and scanning rate are not closely matched (➋ *Fig. 11.9*).

11.1.4 Camera Operation

A standard video (free-running) camera adhering to one of the legacy broadcast television standards, e.g., RS-170 (North America) or CCIR (much of Western Europe), is designed to acquire images continuously and generate video data in a defined format. While this type of camera works well for some vision applications, serious problems may occur when the widget is moving without proper synchronization. The widget-transport mechanism and the camera must be synchronized. Since the former is difficult to control precisely, a phase-lock loop receiving a signal from the mechanical system may be needed to derive the synchronization signals for the camera and its interface electronics.

Most cameras now being designed into Machine Vision systems are asynchronous. An *asynchronous camera* acquires images on command when a trigger pulse is received. Image acquisition and readout can be triggered at any moment, provided that the interval between trigger pulses is greater than the time taken for image acquisition and readout. Some asynchronous cameras are available with internal memory for storing a number of images. These cameras can accept a series of trigger pulses very close together. Image data is stored as it is acquired, and sent to the computer as fast as possible using the available transmission bandwidth.

Some asynchronous area cameras will not run at all without the appropriate control signals: external pixel clock, line-start and frame-start signals. A pixel clock synchronizes the scanning of photo-detectors in the camera with the collection of pixels within the digital image inside a computer. This is particularly useful for precision gauging applications, where sub-pixel interpolation is required. Line-scan cameras typically require only an external pixel clock and a scan initiating pulse.

In a free-running camera, the integration ("exposure") time is often the same as that needed to transmit one video frame (or field). However, many cameras allow shorter exposure times than this, to reduce blurring due to rapid movement. See ➋ Sect. 11.2.11 for more information on electronic shutters.

11.1.5 Image Sensor Architecture

A variety of camera types are available for Machine Vision applications, including:

- Interline transfer CCD
- Frame transfer CCD
- CMOS

Image sensors may be categorized in two principal ways, the type of sensor, and the structure of the readout electronics. Although it is not necessary to know the details of the architecture to choose a suitable camera, there are applications where some basic knowledge of image sensor architecture is useful.

The two basic types of solid-state photosensors, P–N junction (photodiode) and MOS capacitor (CCD) elements both employ a depletion region where photo-generated electron-hole pairs are separated and stored. A key property of a photosensor is its sensitivity at different wavelengths. Photons impinging on an image sensor penetrate into the sensor substrate some distance before being absorbed. The actual distance a given photon penetrates is dependent on its wavelength, and is expressed by a probability function. Shorter wavelength photons (e.g., blue and ultraviolet) tend to be absorbed nearer the surface while longer wavelengths (e.g., infrared) are more likely to penetrate deeper before absorption. Only those photons absorbed in the vicinity of the depletion region contribute significantly to the signal, so the depth of the depletion region influences the responsivity of the image sensor. Typically, shallow depletion regions are used to improve sensitivity in the blue region and reduce sensitivity at red and infrared wavelengths. Since longer wavelength photons are much more prevalent, this is often a desirable trade-off for cameras operating with visible light.

Another important factor affecting sensor responsivity is in the layers of materials placed over the photoreceptor. These layers have an optical effect similar to that of an interference filter. The responsivity of a photodiode sensing element, which is usually covered with only a passivation layer of SiO_2, is somewhat different from that of a MOS sensing element. The latter is usually covered with several layers of materials, including polycrystalline silicon. The photodiode spectral response is considerably smoother than that of an MOS device that has a series of peaks and valleys. To avoid these irregularities, some detectors using MOS sensing elements are back thinned, so that the image is projected on the back of the sensor where there is no passivation or conduction layers rather than its front. By etching the back very close to the depletion layer, the resulting spectral responsivity curve is much smoother.

For most Machine Vision applications, the details of the sensor construction are usually not very critical; other camera characteristics are more important. However, given that each sensor type has certain advantages and disadvantages, exceptionally challenging applications may be addressed more effectively by selecting the most appropriate sensor type.

11.1.5.1 Charge Coupled Device

Charge coupled device (CCD) sensors feature modest cost, low noise and high speed. They usually use a MOS capacitor as the sensing element, but there are some sensors that use a photodiode sensing element. CCD shift registers are used to read out the image information. There are two principal types of CCD image sensors, *interline transfer* and *frame transfer*, which differ in how data is read out.

Interline transfer CCDs have vertical CCD readout registers adjacent to their respective sensing elements. These sensing elements are typically photodiodes. The chip area is substantially less than for a frame transfer CCD, but the relatively low fill-factor (about 30%) reduces sensitivity and the ability to resolve small features (The fill-factor is the percentage of the area of the photo-detector array that is actually sensitive to light.)

As a result, interline transfer CCDs do not function optimally for applications requiring sub-pixel measurements. To mitigate these limitations, some interline CCD image sensors have arrays of plastic lenslets in which each photo-detector element is covered by a small lens to increase the amount of light reaching it. With lenslets, the effective fill factor increases to around 60–70%.

Frame-transfer CCD sensors use an array of MOS capacitors as the light-sensitive elements and have CCD readout registers in a separate area on the integrated circuit. Photo-generated charge is shifted through the array of sensing elements into the readout registers at the end of the exposure time and just prior to readout. Fill factors of 80–95% can be achieved, thereby enhancing sensitivity and resolution. However, this type of device requires a greater chip area and is therefore more expensive than interline transfer CCDs. Also, the transfer of charge from the sensing elements into the readout array uses the sensing elements themselves. Thus, there is the risk of some low level image smear from the light energy that falls on the sensing elements during this transfer.

11.1.5.2 CMOS (Complementary Metal Oxide Semiconductor)

Another class of image sensors is based on CMOS technology, which is in widespread use for integrated circuits, such as microprocessors. The sensing element can be either a photodiode, or MOS capacitor. Sensing elements can share an output circuit, or each photo-detector can have its own output circuit. In the first case, each sensor element is switched sequentially to an output circuit. In the second case, each output circuit is switched sequentially to the sensor’s output line.

Some CMOS sensors have no storage ability for photo-generated charge. The exposure time for a sensing element is the time between successive periods when data is read out. That means that the exposure interval for every sensing element is the same, but the time at which the interval occurs is different for each sensing element. This is called a *rolling shutter*. Rolling shutters are acceptable for imaging stationary objects, but they give a skewed image when imaging moving objects. (Vertical edges appear inclined in the digital image.)

Some CMOS sensors incorporate temporary storage of the photo-generated charge for each sensing element. The photo-generated charges for all sensing elements are transferred simultaneously into their corresponding storage. These storage elements are selected sequentially for readout. This is called a *global shutter*. All sensing elements are exposed at the same time, and no image skew results when imaging moving objects.

The major advantages of CMOS sensors are their low cost and their ability to incorporate other processing functions on the same chip as the image sensor.

11.2 Camera Characteristics

Solid-state image sensors are complex photo-electronic devices that have a variety of features that should be considered when choosing a camera for a given application. If these are ignored, grossly suboptimal performance will be achieved.

11.2.1 Resolution

Two types of resolutions are important when selecting cameras:

- *Sensor resolution.* The number of rows and columns of photo-detectors in the sensor array determines the maximum number of pixels that can be created in the digital image.
- *Optical resolution.* This refers to the ability of the optical system to resolve fine detail in a scene.

The overall resolution of a system is a function of a number of variables, including sensor resolution, focus and depth of focus. Imaging system resolution is frequently characterized by the *modulation transfer function* (*MTF*) and describes the ability of a system to discriminate patterns such as fine parallel lines or concentric circles (➲ *Fig. 11.10*).

11.2.1.1 Sensor Resolution

The sensor resolution is defined as the number of photo-detector elements in each row and column. Typical values were covered above in ➲ Sects. 11.1.1.5 and ➲ 11.1.2 on area and line-scan cameras.

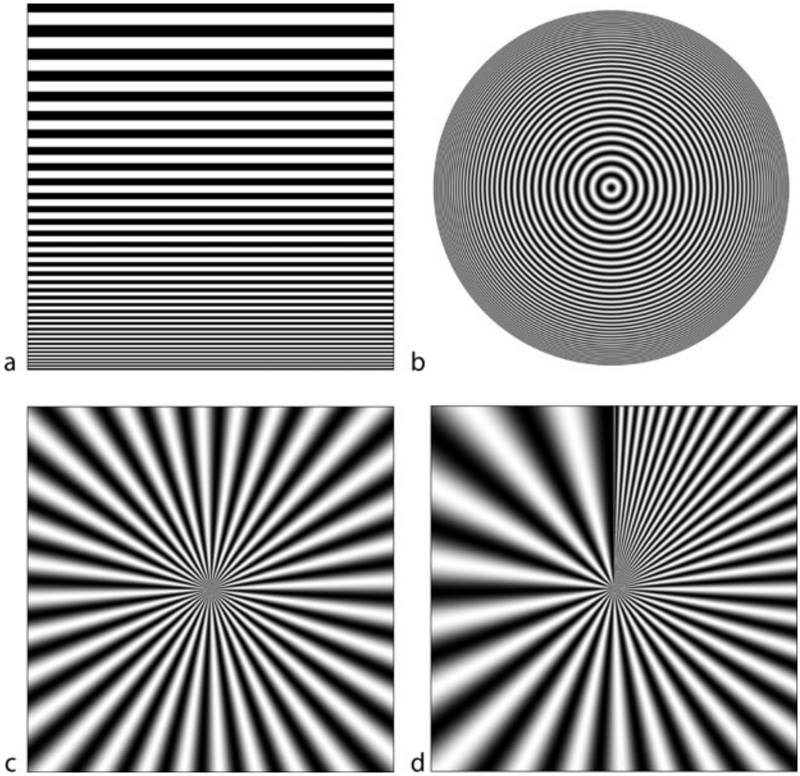


Fig. 11.10

Patterns for measuring acuity of a camera. (a) Variable frequency, linear. (b) Variable frequency, circular. (c) Fixed frequency, radial. (d) Variable frequency, radial

The cameras used most frequently in today's Machine Vision design all have digital outputs, in which case the image resolution exactly equals the sensor resolution. (This ignores situation in which the camera transmits only a portion of the sensed image. See ➔ Sect. 11.2.9.) In cameras with analogue outputs, the frame grabber samples the analogue signal to create the digitized picture. For solid-state analogue cameras, the image resolution is not necessarily equal to the sensor resolution.

11.2.1.2 Video Resolution

Video resolution indicates the size of the smallest feature that can be resolved by an image acquisition system. This includes the lens, image sensor, signal-conditioning electronics and digitiser. Video resolution indicates the number of vertical black and white lines across the field of view that can be resolved across the image. Both black and white stripes contribute to the count, whereas optical resolution is measured in terms of line-pairs. For this reason, the video resolution stated in lines may seem better than the optical resolution stated in line pairs. However, the reverse is always true; video resolution is never better than optical resolution. ➔ *Figure 11.10* illustrates how lines eventually blur together with decreasing pitch.

11.2.2 Sensitivity

Camera sensitivity is specified in two ways:

1. The light level necessary to provide a full-amplitude video output at the edge of saturation.
2. The light level necessary to give a "useful" image, as judged subjectively by a person.

Method [2], widely used in the surveillance field, has very obvious limitations for Machine Vision, while Method [1] is useful, as it is easy to measure and quantify. The factors that control a camera's sensitivity specification include:

- The presence or absence of an IR-blocking filter
- The illumination source used
- What lens, if any, is used and its aperture setting
- Whether or not AGC is enabled
- The gain setting of the camera electronics if AGC is disabled
- The exposure time

The image sensor's spectral responsivity is not equal to that of the human eye. Hence, radiometric (actual power density: watts/m^2) measurements are useful, since they allow accurate quantitative comparisons to be made. However, it is common practice for camera sensitivity to be expressed in photometric terms. This measures perceived brightness (lux or lumens/m^2) of light on either the target that is imaged by the camera or on the image sensors "retina."

Most camera sensitivity tests are performed with an incandescent light source (filament lamp), because its spectrum is very broad and follows a relatively smooth curve that is easy to characterize. Most of the energy radiated by such a source is in the infrared region, where a silicon-based image sensor has the most sensitivity (0.7–1.2 μm). To maintain a sharp image, an IR-blocking filter is very commonly used in practical systems. For this reason, some camera manufacturers specify camera sensitivity using an IR-cut filter. Since some companies do not, it is important to check how the sensitivity has been measured. By definition, non-visible illumination, in the UV and IR spectral bands, does not affect photometric measurements but it does affect the output of the camera, if not blocked by a filter. Therefore, camera sensitivity measured without an IR-blocking filter and with the illumination expressed in photometric terms is very misleading.

Some companies specify camera sensitivity with a given lens, at a given *f*-stop imaging a target with a certain level of illumination. Some others do so with no lens at all, specifying the illumination level incident on the image sensor. The comparison between them is relatively complex.

Ideally, all cameras would be measured at the same gain setting but this is not the current practise, since there is no standard, or convention. This is another reason why comparing cameras is difficult.

Saturation occurs when an increase in scene illumination does not result in an increase in the video signal level. There are three possible reasons for this:

1. The photo-detectors have collected their maximum charge.
2. The (analogue) video amplifier has reached its maximum output.
3. The analogue-to-digital converter is at its maximum output value.

Clearly, it is desirable that the camera be designed so these three situations occur at the same illumination level.

It is common practise in the surveillance field to operate a camera with its *automatic gain control* (*AGC*) enabled. On the other hand, AGC is rarely useful in Machine Vision, since the light levels are normally held within relatively closely defined limits. When a camera's sensitivity is measured with AGC enabled, the amplifier should be providing significant gain, with the sensor not in saturation.

Many older cameras, especially those used for CCTV applications, operate at a fixed clock rate, with a fixed exposure time. However, modern cameras have electronic shutters that provide an adjustable exposure time. Within the "active" period, the charge collected by a photo-detector is proportional to the integral over time of the light flux. Hence, if the illumination is constant, the integration time needed to generate a given output is inversely proportional to the light level. So, to minimize motion blur, high light levels and a short integration time should be used.

11.2.3 Noise

Noise is any component of the signal that does not contain useful information. For a camera, this is any component of the video output that is unrelated to the scene being captured. Video noise sources include:

- Random electrical noise
- Dark current
- Fixed pattern noise (FPN)
- Non-uniformity in sensitivity across the image sensor
- Blemishes (defective sensing elements)

Although it is camera noise that concerns us here, it is important to bear in mind that the digital representation of the scene being examined is also degraded in other ways:

- Fluctuations in the voltage applied to the lamps
- Lamp ageing and malfunction
- Dirty optical surfaces, due to dust, smoke and fumes in the atmosphere, and fingerprints!

11.2.3.1 Random Electrical Noise

Random electrical noise (temporal noise) is present in every electronic system. By careful design of the circuit and its layout, it can be reduced, often very significantly. The noise specification given on camera data sheets reflects only this random noise, which is affected by the temperature of the sensor chip and the clock rate. Cooling a camera is an effective way to reduce its random noise. For cameras that use an adjustable or external clock, the lowest practical clock rate should be used to minimize noise.

11.2.3.2 Dynamic Range

A camera's dynamic range (D_R) is the ratio of its saturation level (S) to its temporal noise level (N) expressed in db.

$$D_R = 20\ \log_{10}(S/N) \tag{11.1}$$

For a camera with 8-bit pixels, the minimum acceptable dynamic range is 48 dB (= 20.log(256)). For 10-bit pixels, this is 60 dB and for 12-bit pixels it is 72 dB. These values do not take into account any other factors contributing to the noise, so the system noise level is not the same as the camera manufacturer's quoted noise level.

11.2.3.3 Dark Current

Even when a camera is in complete darkness, the photo-detector array will provide an output, as if it were receiving a very small amount of light. This signal is called the *dark current* and is caused by the leakage of charge across a photosensor's depletion region. It is an inherent property of silicon-based image sensors. Dark current is proportional to the exposure time and is very temperature dependent; it doubles with every 7°C increase in chip temperature. It can be a significant factor when a camera is operated at elevated temperatures, or long integration times are used.

Most image sensors incorporate reference sensing elements at the beginning of each line. These are the same as normal sensing elements, except they are covered by metal and so are insensitive to light. The output of these elements is determined only by the dark current. At the beginning of each line, the camera electronics samples the signal from the reference sensing elements, and then subtracts that dark-current value from the output of the uncovered sensing elements in that scan line. This is effective in correcting for dark current changes. However, there is some variation in dark current from one photo-detector to another. This cannot be fully corrected in this way. These element-to-element variations in dark current will be exacerbated as the temperature increases.

11.2.3.4 Fixed Pattern Noise (FPN)

Capacitative coupling of control signals on the chip, or its surrounding circuitry, and variations in the readout circuitry produce some subtle unwanted variations in the signal level. Most often, this is a fixed offset to the pixel intensity data. It varies across the image sensor and does not change with exposure. For most cameras operating under normal conditions (i.e., moderate temperature and adequate exposure), this fixed pattern noise (FPN) is usually less than can be detected with 8-bit (256 grey level) digitization. If the voltages applied to the image sensor chip are misadjusted, the camera is operated in temperature extremes, or if the video signal is digitized to 10 or more bits, then fixed pattern noise may be observable. Fixed pattern noise can be corrected by subtracting an individual offset for each pixel.

11.2.3.5 Double Correlated Sampling

Double correlated sampling is a hardware technique used in many CCD image sensors to reduce both dark current noise and fixed pattern noise. Video information consists of a series of pulses in which the amplitude of each pulse is associated with the output of the value. The background level between pulses is a function of dark current plus fixed pattern noise. In double correlated sampling, the camera first samples the level between pulses, and then samples the level of the pulse representing the sensing element output. The main contribution to the difference between those two samples is a function of the amplitude of the light level at a given pixel, with much of the dark current and fixed pattern noise eliminated.

11.2.3.6 Photo Response Non-uniformity (PRNU)

Due to differences in sensing elements from the semiconductor process (e.g., exact size or doping level), not all sensing elements on an image sensor have the same sensitivity. Across the array, the sensitivity can vary by as much as 10%. In applications with high-contrast features, this variation is of little significance but in other cases this factor should be considered. Changes in exposure time can affect photo response non-uniformity (PRNU).

A technique for mitigating the effect of PRNU, called *uniformity correction* or *flat-field correction*, is to associate a stored gain value for each pixel in the image sensor array. This gain is calibrated by imaging a uniformly illuminated target. Thus, the correction not only compensates for PRNU but also for variations in the lens' light gathering capability (e.g., cosine^4 falloff and vignetting) as well as for any non-uniformity in illumination. While the correction can be applied digitally in a look-up table after the pixel's signal has been digitized, it is preferable for it to be applied to the pixel's analogue signal before digitization, to avoid any non-linearities or missed codes.

11.2.3.7 Blemishes

A blemish is one or more adjacent sensing elements whose response differs significantly from neighbouring pixels. Such pixels may be fully on or off all of the time, or just respond very differently. With improved manufacturing techniques, blemishes are much less of a problem now than in years past. However, for sensors with high resolutions, blemishes are still present. Usually, sensor manufacturers offer a range of grades of sensors where the grade is determined by the number and type of blemishes. Fewer blemishes give a higher grade sensor for a higher price.

11.2.4 Blooming

Every photon that is absorbed by a photo-detector element generates a hole-electron pair. The electric field in the detector's depletion region separates the electron from its corresponding hole.

The result is an accumulation of charge in the depletion layer, which in turn reduces the electric field, so that further charge carriers are held there less tightly. When the charge reaches a certain level, some of it "bleeds" into the depletion regions of nearby photo-detectors, or into other nearby circuitry. Hence, a very brightly illuminated spot on the sensor chip produces a much larger blurry region, or a streak in the digital image generated by the camera. Such an effect is called *blooming*. On an interline transfer CCD device, blooming usually appears as a bright vertical streak. Blooming is particularly likely to occur when the camera is pointed directly towards a lamp, or the highlights caused by glinting from a metal or smooth, polished surface.

While image sensor designs incorporate circuitry to remove the excess photo-generated charge, every solid-state image sensor suffers from blooming under intense lighting conditions. A good measure of blooming resistance is the ratio of exposure level at which blooming begins to the exposure level causing saturation of the image sensor.

11.2.5 Gamma (γ)

One way to express the transfer function of a device's output signal (S_{out}) is with the following equation:

$$S_{out} = S_0 . I_{in}\gamma$$

where S_0 is a positive constant, I_{in} is the input light level and γ is a positive parameter. For cameras using a vidicon tube, γ is 0.7. A similar equation relates the brightness of a display to its input signal; for a conventional monochrome CRT display, γ is about 1.3. The combination of the two gammas, gives a combined transfer function with an effective γ of 1. Photographers and users of Photoshop will also be familiar with the use of γ as a way of characterizing the non-linear response of a system.

Modern displays are designed with a γ of 2.2. Cameras designed for use with these displays incorporate circuitry to give them a γ of 0.45 so that again the pair together has a γ of 1.

Modern solid-state cameras are commonly available with γ in the range 0.45–1. In many cameras, γ can be changed by jumpers, switches or, in the case of digital cameras, by software. For Machine Vision a γ of 1 provides the best results. However, when greyscale information is important as, for example, inspecting textured surfaces, it may be preferable to use a different value. Small values of γ emphasize low intensity values and help to suppress highlights due to glinting. (The reader can experiment with this using the QT functions *dga* and *gma.*) Digitizing a camera's signal when it uses a gamma other than 1 will cause non-linearities in the video amplitude that are not anticipated by some image-processing functions. The designer of a Machine Vision system should use a γ of 1 unless it is clear that a different value is beneficial and compatible with the image-processing algorithm.

11.2.6 Electronic Shutter

A solid-state image sensor can emulate a mechanical or opto-electronic shutter, by adjusting the timing between control signals, so that the sensing elements do not accumulate any photo-generated charge. The exposure timing is set up by software.

11.2.7 Region-of-Interest Readout

In applications that do not require the use of all the pixels from the camera, significant improvements in speed can be achieved, by transferring only the data from those pixels needed for image processing. CCD image sensors typically provide for selective readout in which unwanted scan lines are discarded in the image sensor, and only the desired range of scan line information is output from the sensor and transmitted by the camera.

Its architecture limits a CMOS sensor to generating data from a rectangular region of interest (ROI).

11.2.8 Binning

Binning is a technique in which the video signal from two or more adjoining photo-detector elements are combined to define the intensity in a single pixel. Averaging four photo-detector outputs has the advantage of increasing the apparent sensitivity of the camera by a factor of 4 and reducing the noise level by 50% but at the cost of halving spatial resolution both horizontally and vertically. (The image will have only one-fourth as many pixels as it would if binning were not used.)

11.2.9 Camera Power

The majority of cameras require a single power supply, either an AC source or more commonly, a 12 V DC power module. A few speciality cameras, such as line-scan cameras, require two or even three, different power supply voltages. To avoid problems caused by ohmic voltage drop in the power supply cables, these cables should be as short as possible using as large a gauge wire as practical.

11.2.10 Size and Weight

A camera's size is usually specified by giving the height, width and depth of the box that will just contain it. This does not include the lens or other optics. Nor does it make any provision for the space required for cables/connectors. Furthermore, it may not include the necessary mounting hardware. Camera size is clearly important in many applications. Optical techniques such as folded optical paths or the use of coherent fibre optic bundles may save space at the point of imaging. Miniature cameras exist and are suitable for imaging in very constrained spaces such as inspecting the insides of very narrow pipes.

Cameras may be supplied as a one-piece device, or as a small remote sensing head along with a control unit containing the power supply, video amplifier, signal conditioning, timing and control circuitry. Two-piece cameras are more expensive than one-piece cameras with the same performance and are also available with superior performance. The distance between the camera head and its control box in a two-piece camera is normally limited to about 2 m.

The manufacturer's quoted weight for a camera is that of the camera body, and does not include the lens, power supply, connectors or cables. Generally, the weight of the camera is only a serious consideration when the camera is part of the payload on a robot arm; it is not usually important for inspection applications.

11.2.11 Cooling

The power dissipated by the camera can be computed from the product of the supply voltage and current. This, together with the protective housing, determines the temperature differential between the camera and its ambient environment. Cooling may be required for

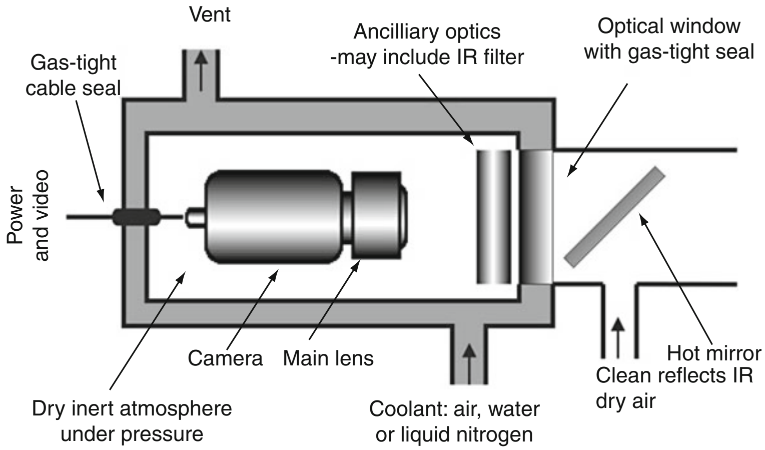


Fig. 11.11
Camera protection. In most applications, some but not all of these precautions are necessary

high-ambient-temperature conditions to ensure proper camera operation. In some cases where excess heat needs to be dissipated, a passive heat sink may be sufficient. At the other extreme in very hot environments, camera cooling may require a double-wall enclosure with circulating air, water or liquid nitrogen to cool the camera (*Fig. 11.11*). The choice of protective optical window for such an enclosure is discussed in Chap. 5.

11.2.12 Temperature

Cameras, like other electronic equipment are limited by the ambient temperature range over which they can be stored and operated. The storage temperature range is usually wider than the operating temperature range. Storage temperature is limited by the range over which thermal expansion will not cause irreversible damage. Operating temperature is limited by the temperature range over which the camera will meet is minimum specifications or, for some manufacturers, just continue to operate.

The performance of a camera is affected by temperature. Usually low temperature limits the upper speed (clock rate) of the camera. High temperature increases the dark current of the image sensor, and reduces the camera's lifetime. The lifetime of electronic components is halved by each 7°C rise in temperature.

11.2.13 Humidity

High humidity is not good for any electronic equipment. Cameras are especially prone to damage from condensation, which can degrade its performance and cause lens fogging. If a camera must be used in a high humidity environment, care must be taken to prevent condensation. This may be achieved by a protective housing, by a simple purging stream of dry, clean air or by slightly heating the camera a few degrees above ambient temperature.

11.2.14 Shock and Vibration

Cameras designed specifically for harsh environments will typically have specified shock and vibration limits. There are two concerns: one is the complete failure of the camera and the other is the loss of alignment between the lens, sensor and the camera's mount. The manufacture's specification usually defines limits to maintain good lens-sensor alignment. Vibration can cause image blurring, particularly when using a telephoto lens. As a general principle, shock and vibration should be minimized to avoid these problems. Since some lenses weigh considerably more than the camera, it may be necessary to support the lens separately. It should also be remembered that vibration can be transmitted to the camera via its cabling.

11.2.15 Radiation

Specialized x-ray sensors have been built by the simple expedient of placing a suitable phosphor on a line-scan camera. These can operate for many months without serious degradation of performance. These are suitable for direct sensing of soft x-rays (i.e., beam energy less than 50,000 keV). However, at high energies, the crystal lattice becomes damaged. Irreparable damage is also caused by high-energy particle beams. The particle stream emanating from a strong radioactive source, such as that found inside a reactor will effectively destroy an image-sensing chip within a few minutes. It will even damage standard glass optical components, rendering them opaque and will make insulating sleeving brittle. Specialized tube cameras are needed for applications where there is a strong particle flux. For less hostile situations, it may be possible to provide sufficient protection for the camera using an optical relay system, such as a periscope, that allows the sensor chip to be placed out of harm's way behind a shielding wall.

11.3 Camera Interface Standards

As discussed earlier, Machine Vision started out using CCTV cameras with analogue outputs meeting one of a number of television standards. Analogue cameras are no longer common in the design of modern Machine Vision systems. Leading the transition into digital camera interfaces, many camera companies offered LVDS (low-voltage differential signalling) interfaces where the signal levels met the RS422 standard. The LVDS interface was never a standard. Typically, there was one line pair for each bit of a pixel's output (e.g., eight pairs for 8 bit digitization) and some number of pairs of signals for camera control. There was no standard for the number, naming or timing of control signals, nor was there a standard for the connector or its pin assignments. With the emergence of standardized digital interfaces, the availability and use of LVDS cameras in new designs has vanished.

There are currently four commonly used interface standards for digital cameras: universal serial bus (popularly known as USB), IEEE 1394 (also known as FireWire), gigabit Ethernet known as GigE, and CameraLink. ➲ *Table 11.1* gives a comparison of the different standards.

The specified bandwidth is the maximum data transmission rate. Except for CameraLink, the data includes header and other information in addition to pixel data. In these three cases, the maximum pixel rate for 8-bit pixels is 90%, or less, of the rated bandwidth.

Table 11.1
Camera interface standards

Feature	Interface type			
	USB 2.0	IEEE 1394b	GigE	CameraLink
Bandwidth (Mbytes/s)	50	80	100	≤ 680
Topology	Client/host network	Peer-to-peer network	Peer-to-peer network	Point to point
Interface protocol	None	IIDC (formerly DCAM)	GenICam	CameraLink
Guaranteed delivery	Yes	No	Yes	No
Guaranteed bandwidth	No	Yes	No	Yes
Cable length (m)	≤5	≤10	≤100	≤10

Except for CameraLink, the protocols are all for networked connections. The topology of a network indicates how it is managed. A client/host network is managed by the host, with all data transmissions being between the client to, or from, the host; clients cannot interface directly with each other. A peer-to-peer network allows any device connected to the network to communication with any other one. If the image capture rate is low and the real-time demands on the system are very relaxed, cameras can share the network with other devices. However, when the Machine Vision system requires the use of much of the available bandwidth, or the system has real-time requirements, then the camera or cameras must be on a dedicated network that is not shared with other devices.

An *interface protocol* allows developers to design software to work with cameras from different manufacturers who conform to the same standards. This allows a camera to be exchanged without needing to change its interface.

Asynchronous protocols guarantee data delivery, by facilitating handshaking and acknowledgement between the sender and receiver. If the transmitted data is not received, or if it is received corrupted, then the data is retransmitted. Thus, delivery is guaranteed, but the retransmission uses up some of the allocated bandwidth and real-time operation cannot be guaranteed.

I*sochronous protocols*, guarantee bandwidth even when other devices are connected to the network. IEEE 1394b provides this capability. When a device is connected to the network, it is given the ability to send a packet of data every 125 μs; the size of the packet is fixed to grant the device the bandwidth it requests. The isochronous protocol does not facilitate handshaking, so there is no guarantee of delivery.

Each protocol has a maximum cable length. In every case, hubs and repeaters that can extend that distance. There are, however, limits on the number of hubs that can be chained together between any two devices on the network. There are interfaces to optical fibres that can extend the data transmission lengths to much greater distances.

It should be noted that each of these standards is under continuing development with higher performance versions expected in the near future. In most cases, the future versions will increase the available bandwidth along with other changes that may, or may not, be useful for interfacing cameras.

11.3.1 USB

The universal serial bus (USB) is a ubiquitous interface facilitated on virtually all PCs. As such, it is a natural choice for a camera interface. Earlier versions of USB, versions 1 and 1.1, were too slow to facilitate a camera interface for all but the slowest Machine Vision systems. USB version 2.0 has sufficient bandwidth to be a viable interface. However, note that the speed of the network is determined by the slowest device connected to the network.

No industry software standard currently exists for interfacing USB cameras to a computer; each manufacturer defines a different standard. Therefore, to switch between cameras usually requires the implementation of a different software interface.

Although the USB specification provides for isochronous operation, most camera manufacturers elect to use the more common asynchronous operation. Delivery is guaranteed but bandwidth is not.

11.3.2 IEEE 1394b

IEEE 1394 is popularly known as *FireWire*. The current version, IEEE 1394b, is a popular camera interface because it offers good bandwidth for most Machine Vision applications, guarantees bandwidth with virtually no image data delivery problems and is a standardized software interface. The earlier version, IEEE 1394a, provides half the bandwidth of IEEE 1494b, but is nevertheless adequate for many Machine Vision applications.

IEEE 1394 supports both asynchronous and isochronous data transmission modes. The IIDC camera standard specifies the use of the isochronous mode. The camera, if accepted onto the network is guaranteed bandwidth independently of what other devices are connected to the network and whether or not they are actively transferring data.

11.3.3 Gigabit Ethernet

Gigabit Ethernet, also known as *1000 base T* or *GigE*, is the latest widely adopted Ethernet standard for copper wire interconnection. It is strictly an asynchronous networking protocol, because it allows any device to transmit data any time it detects the network is not being used. This means that it is possible for two devices to try to transmit data at the same time. The network protocol facilitates detection and recovery from these collisions through both senders detecting the collision and retrying their transmission after a random delay. Handshaking between the transmitter and receiver of the data ensures delivery of the data. GigE is the highest speed network connection currently available, but higher-speed versions are in development.

The *GenICam* specification defines the interface for a Machine Vision camera over the GigE network.

In most computers, the native GigE ports are managed by the CPU. High data rates can consume up to 30% of a CPU's resources. Separate interface boards that contain onboard processing to manage the data transmission are available. These boards reduce the CPU load to a negligible level.

11.3.4 CameraLink

CameraLink is a standard developed specifically for very high-speed Machine Vision camera interfacing. It is not a network protocol, but provides the highest bandwidth standard currently available. Camera data is sent directly from the camera to a dedicated port in the CameraLink frame grabber in the processor. The CameraLink specification dictates the control signals, the data signals and an RS232 serial link between the camera and the interface. There are three versions of CameraLink that differ in the number of parallel data signals: Base, Medium and Full. The differences are in the maximum bandwidth, 227, 453 or 680 Mb/s, and the number of connectors: 1 for the Base version, 2 for the Medium and Full versions. The RS232 serial link is used to program camera parameters, such as exposure time.

11.4 Practical Issues

In this section, two important points are considered that if ignored, will lead to grossly suboptimal performance.

11.4.1 Lens Mount

The lens mount should provide a secure and vibration-proof fixture, and determines:

- The focal distance, measured from the lenses seating flange. (This is called the flange focal distance.)
- The maximum diameter of the optical path.
- The maximum size of the image sensor.

Camera lenses used in Machine Vision often use the C-mount, CS-mount or one of the 35 mm camera lens standards. Mechanical mounts can be classified as either screw or bayonet type. The predominant screw-mount type used for Machine Vision is the C-mount format originally developed for 16 mm movie cameras. The thread diameter is nominally 1 in., with 32 threads/in. In the ANSI B1.1 standard for unified screw threads, it is designated as *1–32 UN 2A*. The flange focal distance is 0.69 in. (17.526 mm) for a C-mount, and 0.5 in. (12.52 mm) for CS-mount, which is otherwise identical.

Bayonet mounts were originally developed for single-lens reflex 35 mm cameras to facilitate rapid lens changes. They are used for Machine Vision applications that require large image-sensing arrays, such as long line-scan arrays. Considering their price and image quality, 35 mm camera lenses can be very cost effective. (Since they are made in very large quantities for the photographic market, manufacturing costs can be kept low.) C-mount lenses are smaller and lighter than 35 mm camera lenses. It is also possible to adapt C-mount cameras to use 35 mm camera lenses, usually with excellent results. More generally, it is possible to adapt a lens with one type of mount to fit a camera with another mount, provided the flange focal distance of the former is greater than that of the camera's mount. In other words, the lens can be moved further away from the camera but not closer. Thus, a C-mount lens can be fitted to a CS-mount camera by using a 5 mm spacer ring.

Some lenses, particularly wide-angle lenses, protrude into the camera body, behind the front flange plate. For monochrome and single-chip colour cameras, this is not normally important. However, three-chip colour cameras require an optical beam splitter in front of the image sensors, so it is not possible to operate a camera of this type with a wide-angle lens.

Bayonet-mount lenses have two drawbacks:

- The springs in the mount have a limited ability to support large heavy (e.g., telephoto) lenses properly. In some attitudes (e.g., viewing horizontally, or vertically downwards), the mounting springs may not provide sufficient tension to keep the lens seated against the mounting flange. This problem can be mitigated by ensuring that the lens mount has sufficiently strong retaining springs for the given lens. The situation is, of course, made worse if there is any significant level of vibration.
- The lens-centring tolerance is relatively loose, to facilitate the rapid changing of lenses. This clearance provides the opportunity for the lens to shift on the camera and can degrade calibration. (On a photographic camera, the ability to change lenses quickly and easily is important. On a fixed-lens Machine Vision camera, it is not!) Modification of bayonet mounts, using set-screws or a similar mechanical approach, is effective but must be done on a bespoke basis.

11.4.2 IR-Blocking Filters

Silicon image sensors respond to light with wavelengths in the range 300–1,100 nm. The glass cover normally placed over the sensor effectively eliminates NUV (i.e., wavelengths shorter than 400 nm). In the NIR region, 700–1,100 nm, the image sensor is very sensitive, but typical lenses are only colour-corrected in the visible region (400–700 nm). So, a good deal of spherical and chromatic aberration occurs at NIR wavelengths and the resulting images will be blurred. Hence, many cameras are provided with IR cut-off filters, either as standard or as an option. With these IR-blocking filters, the image quality is improved dramatically, but at the cost of significantly reduced sensitivity.

For monochrome cameras, with the IR-blocking filter in place, the camera's spectral response roughly approximates that of the human eye. This is, of course, important when the vision system is expected to evaluate the cosmetic appearance of a product.

For colour-imaging applications, the IR-blocking filter is essential, since colour filters often pass IR radiation with little attenuation. Removing NIR radiation also avoids the poor focusing that occurs because *achromatic lenses* do not provide appropriate correction outside the visible wavelengths. Since most silicon-based sensors are very sensitive to NIR wavelengths, this can be important in ensuring good image quality.

11.4.3 NIR Imaging

Since a silicon-based sensor is able to detect near NIR radiation, it is possible to extend the working spectral range of a camera outside, simply by removing its IR-blocking filter. The result is a camera with a wide spectral response (300–1,100 nm). This may be useful in certain applications. By inserting a filter that eliminates all visible light, we produce a camera that is sensitive only to NIR (700–1,100 nm). This is a simple and inexpensive way to produce a camera that has applications in areas where natural products are inspected and will often suffice for preliminary feasibility studies (➲ *Fig. 11.12*).

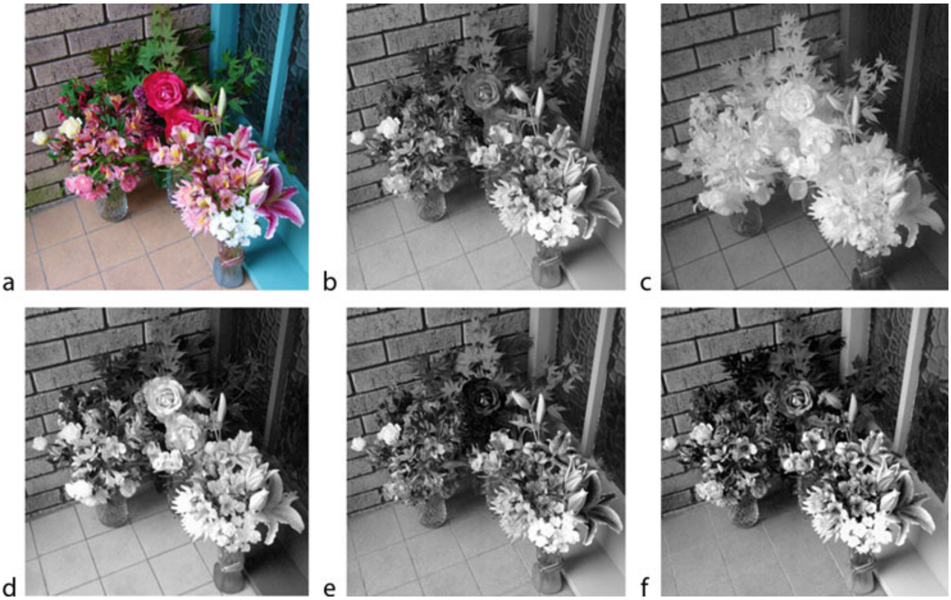


Fig. 11.12
Visible and NIR images. (a) Natural colour. Image obtained using a Sony DSC-H9 digital camera. The NIR-blocking filter is in situ. (b) Intensity channel. At each pixel, the intensity here is equal to the sum of the RGB values. (c) The NIR blocking filter was removed and a VIS-blocking filter inserted. The spectrum detected here is approximately 850–1,100 nm. (d) R-channel. (e) G-channel. (f) B-channel

11.5 Operation of a Line-Scan Camera on a Linear Conveyor

11.5.1 Notation

f – Clock pulse frequency (Hz)
N_a – No. of *active* photo-detectors in the line-scan array. (Defines no. of pixels across the image.)
N$_d$ – No. of *dead* photo-detectors. (This is a fiction, introduced to simplify the calculations.)
V – Conveyor speed (m s^{-1})
T_i – Integration time for the photo-detectors to accumulate charge. Note: $T_i \leq N_a/f$
M – Optical magnification
W – Width of the line visible to the line-scan camera

- Scan time

$$T_s = (N_d + N_a)/f \text{ s}$$

- Active scan interval

$$T_a = N_a/f \text{ s}$$

- "Dead" interval

$$T_d = N_d/f \text{ s}$$

- During the scan interval (T_s) the conveyor moves

$$V(N_d + N_a)/f \text{ m}$$

- Conveyor movement between consecutive clock pulses (1/f s)

$$D_1 = V/f \text{ m}$$

- Conveyor movement during the active scan period (T_a)

$$D_a = N_a V/f \text{ m}$$

- Conveyor movement during the “dead” period (T_d)

$$D_d = N_d V/f \text{ m}$$

- Conveyor movement during the complete scan period ($T_d + T_a$). This is the effective height of pixels.

$$D_t = (N_d + N_a)V/f \text{ m}$$

- Effective pixel width (across conveyor)

$$W/N_a \text{ m}$$

- Effective angular offset of the line-scan array due to scanning it in finite time

$$\theta = \tan^{-1}(N_a V/fW)$$

(This can be corrected by rotating the camera by an amount $-\theta$.)

Acknowledgement

Much of the source material for this chapter was kindly supplied by Perry West, Automated Vision Systems, Inc., San Jose, CA, USA.

12 X-Ray Inspection

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_12,

Abstract: Automated X-ray Inspection (AXI) is useful for two types of application: detecting foreign bodies, particularly in food, and examining the internal detail of safety-critical products. At first sight, AXI merely seems to be a straightforward extension to Machine Vision but there are several complications. There is only a limited scope for adjusting the spectrum of an x-ray beam. It is not easy to build x-ray mirrors and lenses, so we cannot effectively focus, or collimate, an x-ray beam. Since an x-ray ray tube generates a conical beam, the images it forms are complicated. The diverging beam introduces parallax errors. Furthermore, objects near the x-ray source appear to be bigger than those further away. Features along the beam are superimposed, creating a jumble of detail that is difficult to analyse. AXI has its own set of special features that distinguishes it from other areas of Machine Vision. One particular point emphasises this. X-rays present a clear danger to health, so there is a need for good radiation screening and a variety of other safety measures. Mathematical and software modelling of AXI systems can often help in devising effective image analysis algorithms. These are simple and effective when inspecting close-tolerance accurately aligned artifacts. In this case, a point-by-point comparison of the test image and a reference image may be effective. As the image variability increases, more complex inspection algorithms may be required, often involving the use of sophisticated AI methods. The chapter concludes with a discussion of twin-beam x-ray systems that can improve the detection of defects and foreign bodies.

12.1 Introduction

The term *X-ray* is used to refer to electromagnetic radiation with wavelengths between those of ultraviolet and gamma rays (➔ *Fig. 12.1*). While the exact boundaries are blurred, we shall take it that *X-ray* refers to radiation within to the following approximate limits:

Frequency: 3×10^{16} Hz to 3×10^{19} Hz
Wavelength: 10^{-8} to 10^{-11} m
Energy: 120 eV to 120 keV

In fact, many of the remarks below are also relevant to gamma rays, which have higher energies than X-rays.

We shall take it that X-rays are generated by a cathode ray tube, while gamma rays are emitted from radioactive materials, such as radium and uranium. Again, this boundary is blurred. Both X-rays and gamma rays are forms of *penetrating ionizing radiation* and are capable of travelling through many common materials, including human tissue. Of course, X-rays are familiar to everyone through their use in medicine but it is industrial applications that interest us here. In a naive way, we might study Automated X-ray Inspection (AXI) by

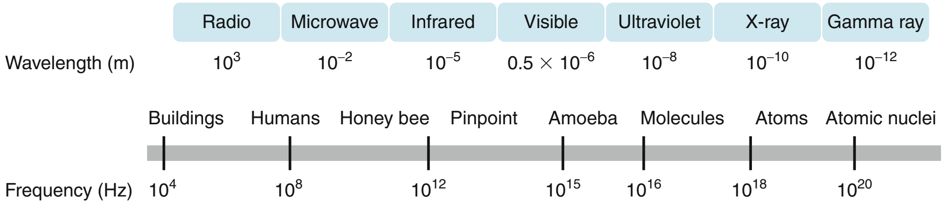


Fig. 12.1
The electromagnetic spectrum (EMS)

merely applying established image processing techniques to the pictures obtained by taking a few samples to the Radiography Department of the local hospital. That is the Computer Science approach! However, it does not give us the complete picture of the potential and complexities of AXI. Firstly, there are many different types of X-ray generator. We can use X-rays covering a wide range of spectral forms. We can use either area-scan or line-scan sensors and specialised image processing algorithms are often needed. AXI is a rather specialised area of Machine Vision but it is, nevertheless, part of the broader subject of this book as it is concerned with all aspects of system design, including the very important health and safety issues that must be addressed when using X-rays. The engineering of X-ray image-forming systems requires specialist skills and knowledge. By claiming that AXI is an area of Machine Vision we are simply reflecting the fact that it is engineering, not merely science. This chapter should not be seen as giving a complete coverage of AXI; we are only able to point to some of the highlights and problems. This is a fascinating and, as yet, largely under-used area of Machine Vision. We hope to show the potential for the future, as well.

Automated X-ray Inspection has two principal areas of application:

(a) Examining the *internal* structure of engineering products and assemblies. Checking the integrity of high-value, safety-critical items is particularly important.
(b) Detecting embedded foreign bodies, particularly in food materials, food products, and pharmaceutical products.

For tasks like these, X-rays provide an obvious sensing mechanism, since they provide hygienic, non-contact sensing that does not damage even delicate and fragile structures. (At the low X-ray doses involved, there is no significant damage, even to protein and other large organic molecules.) There is a popular misconception that X-rays make materials radioactive. *This is not so!* As soon as the X-ray beam is switched off, there is no significant residual radiation. There is no danger to health from a sample that has been inspected using X-rays. Of course, special care must be taken to protect people from exposure to the X-ray beam. *This is a legal requirement!* With good equipment design, routine maintenance, regular independent testing and diligent monitoring of worker exposure dose, there should be no reason to fear the presence of X-rays in the factory.

In the past, X-rays have been used in a wide variety of inspection applications. ➲ *Table 12.1* lists a representative sample of these.

12.2 Elements of an X-Ray System

An AXI device has the same basic features as any other Machine Vision system (➲ *Fig. 12.2*). However, the former presents some special requirements and challenges, due to the shorter wavelengths involved. In an AXI system, the standard modules can readily be identified, albeit in a disguised form:

1. Mechanical handling of the widget/material to be inspected. This presents very special requirements for hygiene when working in the food and pharmaceutical industries. (These are two of the subject's most popular areas of application.) In view of the danger associated with ionising radiation, the mechanical handling system must be designed with worker safety in mind.
2. Radiation source (X-ray tube)

Table 12.1

Applications of AXI

Application	Detecting/inspecting/checking
Aluminium castings	Inclusions, voids
Baby Food	Foreign bodies
Baked Products (e.g. cakes, pies)	Amount of filling
Ball grid arrays	Integrity of solder joint
Blister packs	Existence of contents, foreign bodies
Bread	Texture analysis
Cans	Structural integrity
Cartons and boxes (pharmaceuticals)	Presence of medication and leaflet
Ceramic packaging (microelectronics)	Structural integrity
Ceramics	Structural integrity
Chicken, fish and meat	Bones
Cultural/archeological artifacts	Statues, pots, paintings, mummies, etc.
Eggs	Developing embryo
Electronic/electrical assemblies	Loose wires, correctness of assembly
Fish	Bones, weighing
Food materials (milk, soup, etc.)	Foreign bodies
Gems	Inclusions, air bubbles
Glass jars and bottles	Birdswings, spikes, malformation
Leather	Thin spots due to earlier animal injury
Meat	Weighing
Multi-packs (filled jars on tray)	Foreign bodies
Pipes	Structural integrity of seams
Plastic Bottles	Dimensions, inclusions,
Plastic jars and bottles	Dimensions, inclusions, wall thickness
Printed circuit boards	Conductor alignment and integrity
Product counting	Various
Ready meals	Foreign bodies, Filling levels
Rice (plant stems)	Infestation by insects
Seals	Structural integrity
Timber	Knots, splits, fungal attack
Turbine/compressor blades	Integrity, clearance of cooling channels
Tyres (automobile and aircraft)	Structural integrity
Vegetables	Infestation by insects
Welding	Structural integrity

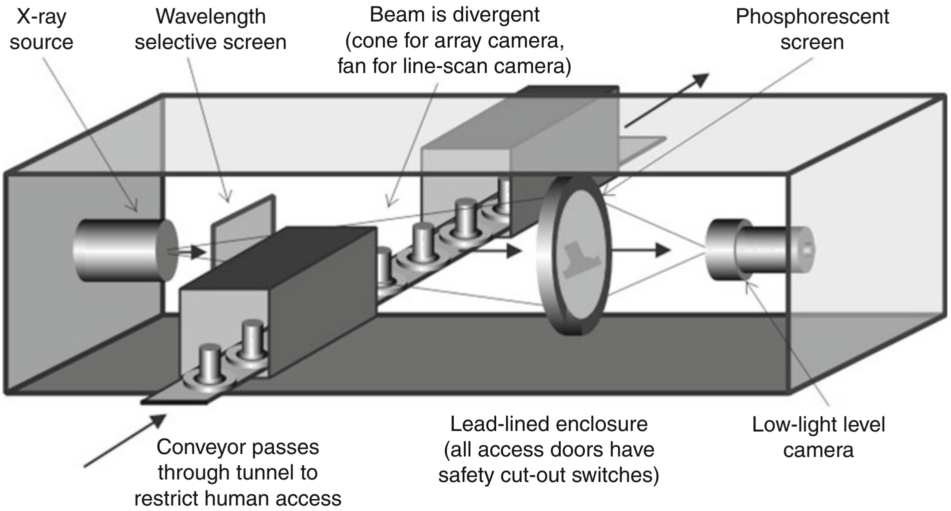

X-ray source
Wavelength selective screen
Beam is divergent (cone for array camera, fan for line-scan camera)
Phosphorescent screen
Conveyor passes through tunnel to restrict human access
Lead-lined enclosure (all access doors have safety cut-out switches)
Low-light level camera

Fig. 12.2

Imaging formation system for AXI. The parts handling system is designed to prevent workers from putting their hands in the X-ray beam

3. Device to modify the radiation properties (X-ray filter)
4. Image sensor and capture module. This may be a standard monochrome visible light camera, or a camera that has been modified to detect X-rays directly. It may be an area-scan or line-scan device.
5. Image processing. X-ray images require both standard techniques and a degree of innovation in algorithm design, in view of the special nature of X-ray images.

Of these, 2–5 are discussed below, in view of their rather special nature in AXI. One additional feature is essential: *radiation protection*. This involves

(a) Shielding using lead sheeting to line the walls of the inspection chamber
(b) Masking to block X-rays except when they are needed
(c) Automated parts delivery and removal (e.g. a conveyor belt running through the inspection chamber)
(d) Tunnels at the entrance and exit ports, so that hands cannot reach inside the inspection chamber
(e) Safety cut-off switches on all access doors
(f) Detectors (e.g. infra-red) to sense hands being placed close to the beam
(g) Anti-tamper fixings; locks on access doors
(h) Regular checking for radiation leakage
(i) Monitoring of staff exposure to X-rays
(j) Automatically switching off the X-ray beam when no part is present for inspection

Additional, features required by certain X-ray systems include:

(k) Cooling water or air
(l) Stabilised power supply
(m) Air conditioning, for certain types of sensitive X-ray detectors

12.2.1 Generating X-Rays

The construction of an X-ray generator is roughly similar to that of a cathode ray tube; a beam of electrons is accelerated by an electric field to high energy (➋ *Fig. 12.2*). The energy of electrons in the beam is proportional to the accelerating voltage (e.g. cathode–anode potential difference) The beam is focussed onto a metal target, such as copper, tungsten, molybdenum or silver. X-rays are generated through the interaction of the energetic electrons in the beam to those bound to the metal atoms. Much of the beam energy is absorbed by the anode, causing intense local heating. The remainder of the beam energy actually generates X-rays, through the *Bremsstrahlung effect.*

Continuum Spectrum A broad spectrum of X-ray energies is generated by a beam of high-speed electrons falling on a metal surface and can be modelled approximately using *Kramer's equation* [1]

$$\mathrm{I(E)} = \mathrm{I_B Z((E_0 - E)/E)} \tag{12.1}$$

where

- I_B is the anode current in the tube. (This is proportional to the number of electrons forming the beam.)
- Z is the atomic number of the tube target material (Tungsten is often used for this purpose, in which case Z =74. Copper is also used sometimes. Z=29).
- E_0 is the cathode–anode potential difference. (The velocity of electrons in the beam is proportional to the accelerating voltage.)
- *I(E)* is the intensity of the X-ray beam. (*I(.)* is proportional to the number of X-ray photons created.)
- *E* is the X-ray energy (measured in keV).

Energy is related to frequency (*f*) using the following equation:

$$E = hf \tag{12.2}$$

where h is Planck's Constant ($4.13566733 \times 10^{-15}$ eVs) Since

$$f = c/\lambda \tag{12.3}$$

where c is the speed of light (c=299,792,458 ms^{-1}) and λ is the wavelength (m), we also have

$$E = hc/\lambda \tag{12.4}$$

➋ Equation 12.1 generates an hyperbolic curve with $I(E_0)=0$ and which increases to infinity at E=0. It describes the *Bremsstrahlung effect* (German. In English, *Braking Radiation*) but does not give a complete picture. The observed output spectrum drops off at very low energies (long wavelengths), because very soft X-rays are easily re-absorbed within the target material. Increasing either the atomic number (*Z*) of the sample, or the beam current (I_B) produces a higher continuum, that is a more intense X-ray beam but does not alter its spectrum.

Refinements Two other factors modify the spectrum (➋ *Fig. 12.3*):

- The exit window of the x-ray tube. This absorbs some of the energy emitted from the anode.
- The anode material. Some elements generates additional X-ray energy within narrowly defined energy bands, superimposing characteristic spikes on the continuum spectrum.

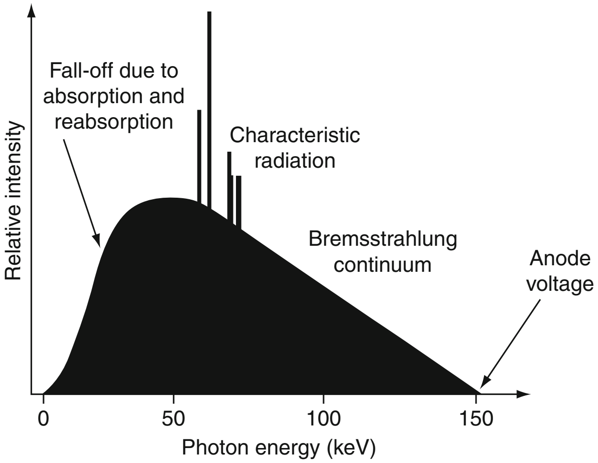


Fig. 12.3
In practice, the X-ray spectrum is determined by four major factors: Bremsstrahlung effect (continuum, Eq. 12.1), narrow-band characteristic of the anode material, reabsorption within the anode, absorption within the exit window

These spikes are caused by quantum energy level changes in the electrons surrounding atoms within the anode material. Although these peaks are relatively large compared to the continuum spectrum, the total amount of emitted energy they contribute is small compared to that from the *Bremsstrahlung* effect.

Beam Power Only a very small fraction of the incident energy from the electron beam is emitted as X-rays within the continuum spectrum. The proportion of the beam energy converted to X-rays is approximately

$$1.1 \times 10^{-8} ZE_o \tag{12.5}$$

Hence, if the anode voltage is 25 KeV and $Z=74$ (tungsten target), only about 2% of the beam energy is converted to X-rays. The remainder is converted to heat in the anode. Since the beam current is I_B, the (continuum) beam power is about

$$1.1 \times 10^{-8} ZI_B E_o{}^2 \tag{12.6}$$

The power dissipation in the tube is almost equal to $I_B E_o$. (This ignores the filament power.) A fundamental difficulty with this method of generating X-rays is that this power is concentrated over a very small spot on the anode. It can be high enough to cause evaporation from a static anode, even from tungsten (Melting point 3,695 K; boiling point 5,828 K), leaving the surface pitted. As we shall see, special precautions are sometimes needed to avoid this.

12.2.1.1 Types of X-Ray Tube

There are several types of X-ray tube (*Fig. 12.4*):

- *Oil Coiled*: The whole tube is immersed in oil to remove excess heat.
- *Gas Cooled*: The tube is surrounded by gas. Heat transfer in gas is much lower than oil, so this type is suitable for lower power levels.

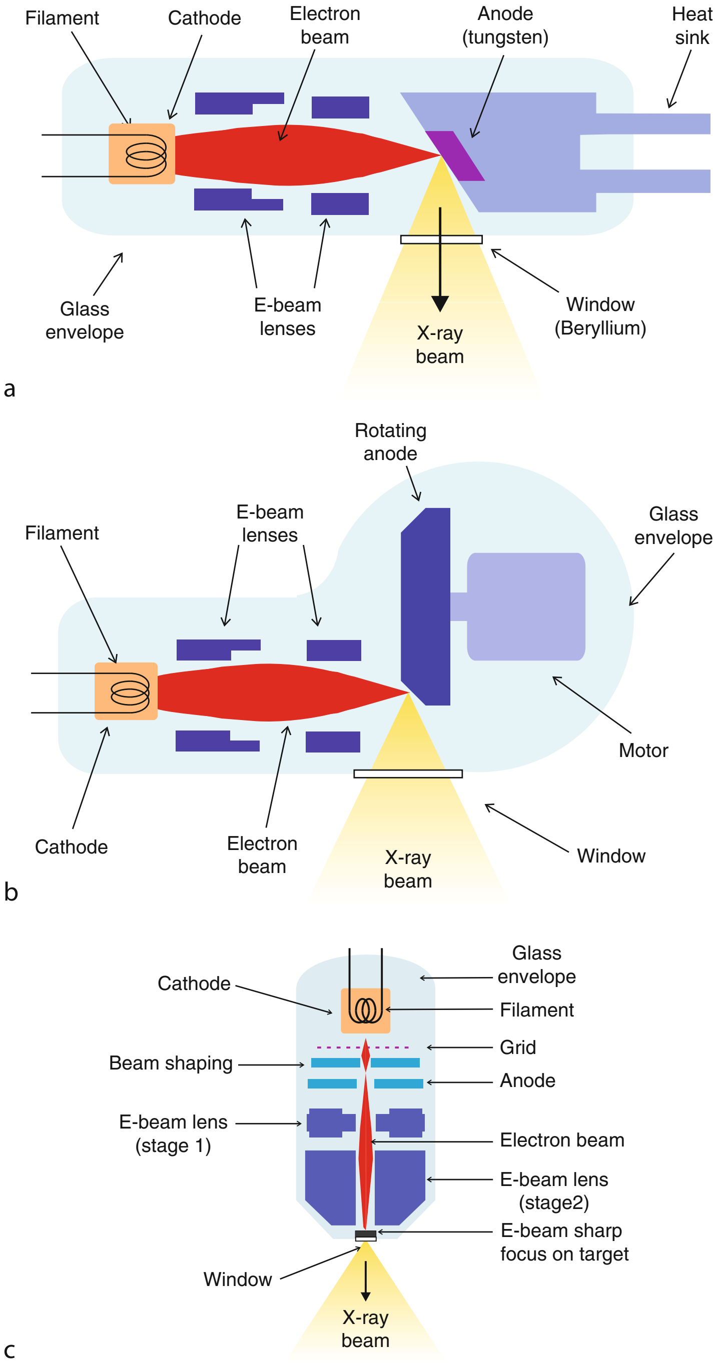


Fig. 12.4 (Continued)

- *Rotating Anode*: The target is a thick metal disc (e.g. tungsten, copper or silver) that is rotated rapidly so that the heat generated by the electron beam does not cause pitting as metal is evaporated from its surface. To increase the power levels further, the whole assembly can be immersed in oil.
- *Micro-focus*: The target is a thin film. Electrons strike it on one side and X-rays emerge from the other. The spot formed by the electron beam on the cathode has a small diameter (a few in microns). The electron beam can be diverted to different parts of the target by a magnetic deflection unit. This allows the life of the tube to be extended, by allowing different parts of the target to be used.

Intense high-energy (i.e., short wavelength) X-rays are produced in a particle accelerator (cyclotron and synchrotron) generating Bremsstrahlung radiation. They been used for high-value, specialist applications, such as examining aircraft and automobile engines.

Beam Switching: Just as in thermionic valves, a grid can be used to control the anode current. This allows the beam to be switched on only for the very short interval when it is needed. This makes the system safer and increases the life of the tube, by reducing anode heat dissipation.

Alternative Sources: Radioactive sources can also be used to produce X-rays. One important advantage of nuclear sources of penetrating radiation is the fact that they can be formed into a variety of shapes: plate, wire, rod, disc, sphere, torus, etc. They are also available in liquid form. Unlike an X-ray tube, radioactive materials cannot be "switched off", so they have to handled with extreme care. In addition, there are strict legal controls about their use, storage and disposal. For these reasons, nuclear sources are unpopular but they do offer three very important benefits: versatility, low running costs and the fact that they do not require any power at all. They do have a finite life but this can be predicted accurately, as can the spectrum. ➋ *Figure 12.5* shows how a nuclear source can be used to generate a crudely collimated beam of X-rays.

12.2.2 Beam Divergence

An important feature of an X-ray tube is that all rays are generated within a very small area on the target. The rays then spread out to form a conical beam (➋ *Fig. 12.4*). Beam divergence is an

◘ Fig. 12.4

Physical construction of three different types of X-ray tube. (a) Standard low-power device. The filament is heated by an electric current to generate a cloud of electrons. To facilitate electron emission, the filament may be coated with a material such as caesium. These thermal electrons are then accelerated by the high potential difference applied between the cathode (filament) and anode. This accelerates the electrons to high speed (energy). The electron beam is focussed to a small spot on the anode, which consists of a block of metal (e.g. copper or tungsten) X-rays are generated as these electrons are suddenly decelerated. Notice the diverging (conical) X-ray beam, which is a feature of the Bremsstrahlung effect (b) High-power X-ray tube [2]. In this case, the target is a thick disc of copper or tungsten that rotates rapidly, to avoid local over-heating. To improve cooling, the whole tube may be immersed in oil. (c) Micro-focus X-ray tube [3]. The electron beam is focussed by an electromagnetic lens onto the target. As a result, the focal spot may be only a few microns in diameter. The target is formed by a thin layer of tungsten on a plate of other metal. The latter also acts as the exit window for the X-rays

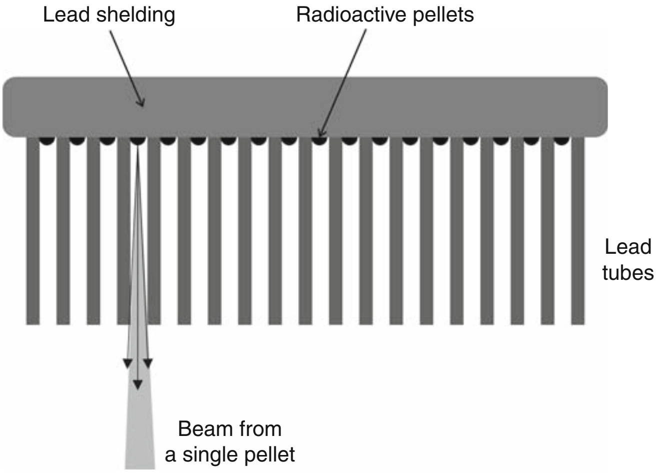


Fig. 12.5

Collimator for a nuclear radiation source. When a line-scan sensor is used, the collimator is a comb-like structure. For an area-scan sensor, the collimator consists of a series of parallel, deep, narrow wells, set on a 2-dimensional array

important feature of many AXI systems that use an area scan image sensor. It complicates the structure of the images produced and consequently has major implications for the design of AXI image processing algorithms. It is difficult to produce a lens, or mirror, that can effectively deflect X-rays and correct beam divergence. So, it is left to the image processing software to accommodate the images produced by a radiation source that is in many ways far from what would be ideal. More will be said later about the consequences of beam divergence.

Beam divergence is important along only one image axis in a line-scan AXI system (➔ *Figs. 12.21–12.24*). It can effectively be overcome by using a nuclear radiation source and a collimator. (➔ *Fig. 12.5*)

Beam divergence can be used to good effect to build an X-ray microscope. Placing the sample close to micro-focus tube and the scintillator screen much further away can provide high magnification. If the target-sample distance is D_1 and the target-scintillator distance is D_2, the X-ray magnification is D_2/D_1. The resolving power is, of course, limited by the spot size.

12.2.3 Scintillators (Phosphors)

Neither a conventional video camera, nor the human eye can sense X-rays directly. (It would be dangerous to try to view them!) So, a scintillator is used. This consists of a thin screen coated with a phosphorescent material that converts X-rays to VIS light (VIS). A variety of materials, many quite exotic, has been used. For example, *yttrium oxysulphide* will glow when it is irradiated with low-energy X-rays, while *gadolinium oxysulphide* can be used for sensing high-energies.

An ideal scintillator would absorb all the X-rays it receives and generate a bright VIS image. However, thick film of phosphorescent material would trap a lot of the visible light it generates, thereby reducing the sensitivity of the detector.

Three important factors when choosing a phosphor are:

(a) The spectrum of X-rays that it absorbs. Clearly, this must be compatible with the spectrum of the X-ray source (i.e., the tube and filter sub-system). Doping the phosphor material affects the light emitted. Gadolinium oxysulphide doped with europium emits light with wavelengths close to the peak in the spectral response of a silicon photodetector. This allows efficient conversion of X-rays to electrical signal.
(b) The spectrum of VIS light that it emits. (This should match the spectral response of the photo-detector array.)
(c) The decay of visible light after the X-ray beam has been switched off. If the time constant of this decay is too long, a blurred image will usually result. If it is too short, the final (i.e., digital) image will be noisy. Some temporal smoothing can be deliberately introduced by using a slow-decay phosphor. The decay process consists of two parts. The first is an exponential decay to a low level of light emission, known as *afterglow*. The second is a slower arithmetic dimming of the afterglow. (The decrease in intensity is approximately linear over time.)

In ❯ *Fig. 12.1*, the camera is shown on the opposite side of the scintillator screen from the sample. Of course, this requires the transmission of radiation through the screen. It is possible to place the camera and sample on the same side, thereby avoiding this and improving the sensitivity for a given X-ray intensity. Although the camera is then off-set, the resulting image warping is slight and can easily be corrected optically.

The sensor geometry may be 2-dimensional (array-scan), or 1-dimensional (line-scan). In the latter case, a 2-dimensional image is formed by repeatedly scanning the photo-detector array, while the object being examined moves past, at a constant and known speed. In most systems, particularly those involving large inspection samples, it is necessary to use a separate scintillator screen, together with a standard (i.e., monochrome, area-scan) camera. Notice that it may be necessary to make the scintillator screen considerably larger than the object being examined. There are two reasons for this:

- The beam continues to expand after passing the inspection sample.
- Different inspection samples may not always be located in exactly the same position.

Another option is to use a VIS line-scan camera, with a strip of phosphor material placed in front of its lens. The conical beam will probably be masked off, close to the generator tube, to produce a fan-shaped beam. (This reduces the radiation hazard.) Proprietary line-scan sensors for X-rays are also available. These use several phosphor-coated, line-scan array chips. These can be butted end-to-end, to produce a very long line-scan device.(There are inevitably small gaps between the chips.) Twin line-scan sensors, overlaid with different phosphors, have been used in dual-wavelength X-ray systems (❯ *Fig. 12.6*).

A summary of the characteristics of commonly used phosphors is given in ❯ *Table 12.2* [4, 5, 11].

12.2.4 X-Ray Filters

Just as in conventional optics, it is possible to adjust the spectrum of an X-ray beam using a filter. It is possible to enhance image contrast using a flat plate composed of some suitable material. This is particularly important for detecting foreign bodies in food and pharmaceutical

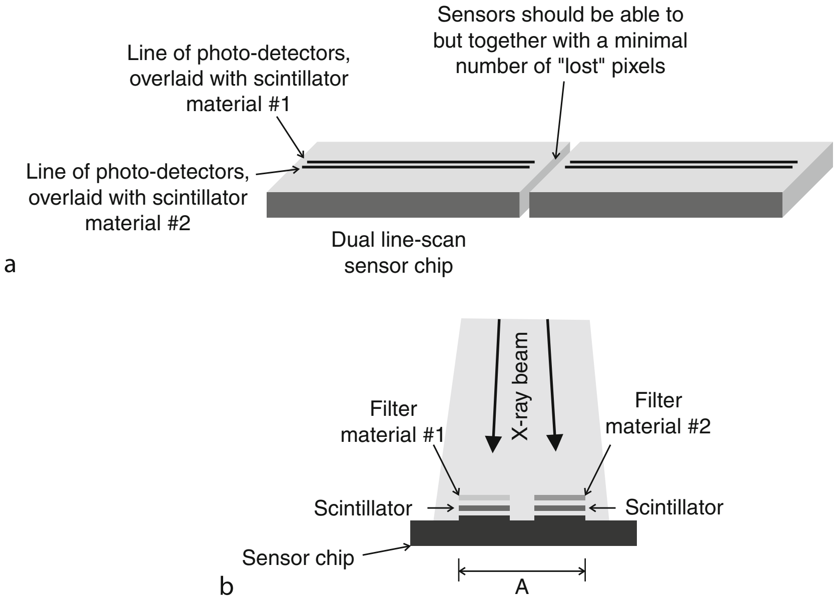


Fig. 12.6

Sensor for dual energy sensing of X-rays. (a) Two line-scan arrays on a single chip. These are conventional VIS sensors but are overlaid with a phosphor and different filter materials. Protective layers are not shown. (b) End view (schematic; the fan-shaped X-ray beam is viewed from the side). For this arrangement to be effective, features in the inspection sample should be several times larger than distance A

Table 12.2

Phosphors commonly used in X-ray scintillators

Phosphor (Key – main material: doping material)	Wavelength at peak emission (nm)	Time constant (decay to 10% intensity)	Afterglow
Gd_2O_2S:Tb	545	1.5 ms	Low
Gd_2O_2S:Eu	627	850 µs	Yes
Gd_2O_2S:Pr	513	7 µs	None
Gd_2O_2S:Pr Ce	513	4 µs	None
Y_2O_2S:Tb	545	1.5 ms	
Y_2O_2S:Eu	627	850 µs	Yes
Y_2O_2S:Pr	513	7 µs	None
Zn(0.5)Cd(0.4)S:Ag	560	80 µs	Yes
Zn(0.4)Cd(0.6)S:Ag	630	80 µs	Yes
$CdWO_4$	475	28 µs	None
$CaWO_4$	Blue (410 nm)	20 µs	None
ZnS:Cu (GS)	Green (520 nm)	minutes	Long

materials/products. However, the options are rather limited; we cannot, for example, build “high-Q” X-ray filters with arbitrary waveband limits. Typical materials used in X-ray filters are:

- To pass high energy X-rays: platinum, gold and lead
- To pass low energy X-rays: molybdenum, silver or tin

Copper, aluminium, zinc and more exotic materials, such as the rare earth elements, are also used [6, 7]. Figure 12.15 sketches the absorption spectra for various materials.

Dual-band systems can be built, using two photo-detector arrays. These are fitted with different filters. ➲ *Figure 12.6* shows the physical layout of such a sensor based on two line-scan photo-detector arrays. The principle of operation is explained in ➲ *Fig. 12.7*. Cobalt, Chromium, Magnesium and Silicon all have very similar spectral characteristics, differing only in the wavelength of a single “step” change in absorption. Graphite has a similar characteristic but does not have a “step” and can therefore be used as an attenuator to match the gains in the two video channels.

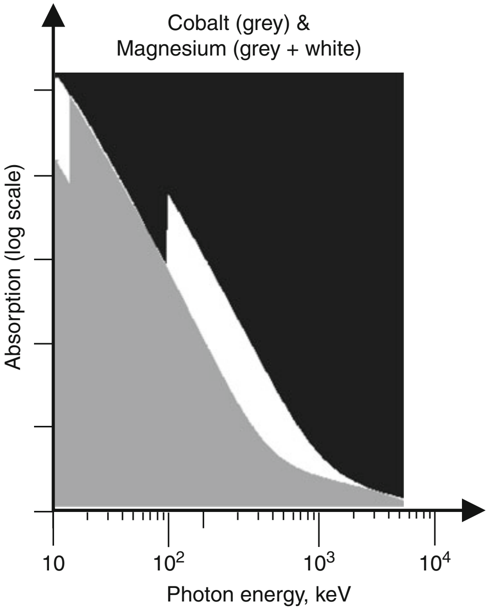


Fig. 12.7

Creating an X-ray band-pass filter by comparing two sensor channels coated with different materials. The absorption spectra for Cobalt (*grey*, Co) and Magnesium (Mg, *grey plus white*) are very similar in the range 30–70 keV. Co is used as a filter in one channel and Mg for the other. The gains in these two channels are carefully adjusted, so that within this energy band, the two cameras produce identical outputs. These two outputs are then subtracted electronically or in software. The combined output is therefore sensitive only to energies in the band 100 keV to about 1,000 keV. Within this waveband, the Co channel has a gain about 4.5 times that of the Mg channel. Thus, the gain of the dual-channel system is very nearly constant over the band 100–140 keV with a sharp cut-off to near zero at about 100 keV. Since Cobalt, Chromium, Magnesium, Silicon and Graphite all have similar absorption spectra, any two of them may be combined in this way. Other pairs of filter materials are possible; the Co:Mg pair is simple to explain

12.2.5 Digitising the Signal

The video signal from the sensor must be digitised before it can be processed by computer. This is achieved in the same way as in any other Machine Vision system and hence will not be discussed further in this chapter.

12.3 Understanding and Analysing the Images

In some applications, standard image processing techniques will suffice to complete the inspection function (➲ *Fig. 12.8*). Hence, for example, a simple processing sequence should suffice to measure the filling level in a tube of tooth-paste, or count the tablets in a blister-pack strip. Although conventional image processing has a role to play, analysing X-ray images sometimes requires special methods, in view of their particular characteristics. As in other areas of Machine Vision, automating the interpretation of AXI images is easier and more effective if we know exactly what to expect when "good" product appears at the point of inspection. Simple existential inspection (i.e., determining whether a part is present/absent) of rigid close-tolerance engineering products/assemblies is straightforward. The task is made even easier if parts are always presented in the same position and orientation, relative to the X-ray source and sensor. ➲ *Figure 12.9* shows an example: an aerosol spray valve. This consists of a number of accurately made plastic and metal components. (Although these are not high-precision parts in the normal engineering sense, they are very much less variable than natural products.) Verifying that such a product is complete and correctly assembled can be achieved very easily, simply by checking the intensity profile along a series of pre-defined scan lines. If there is some small variation in the position/orientation of the assembly, it may be necessary to

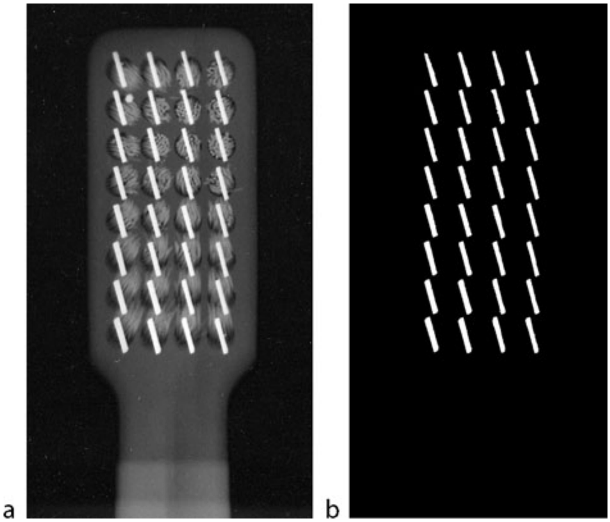


Fig. 12.8

Checking completeness of assembly of a tooth brush. (a) Original image. The bright bars are metal clips needed to retain the bristles. (b) Metal clips detected by simple thresholding

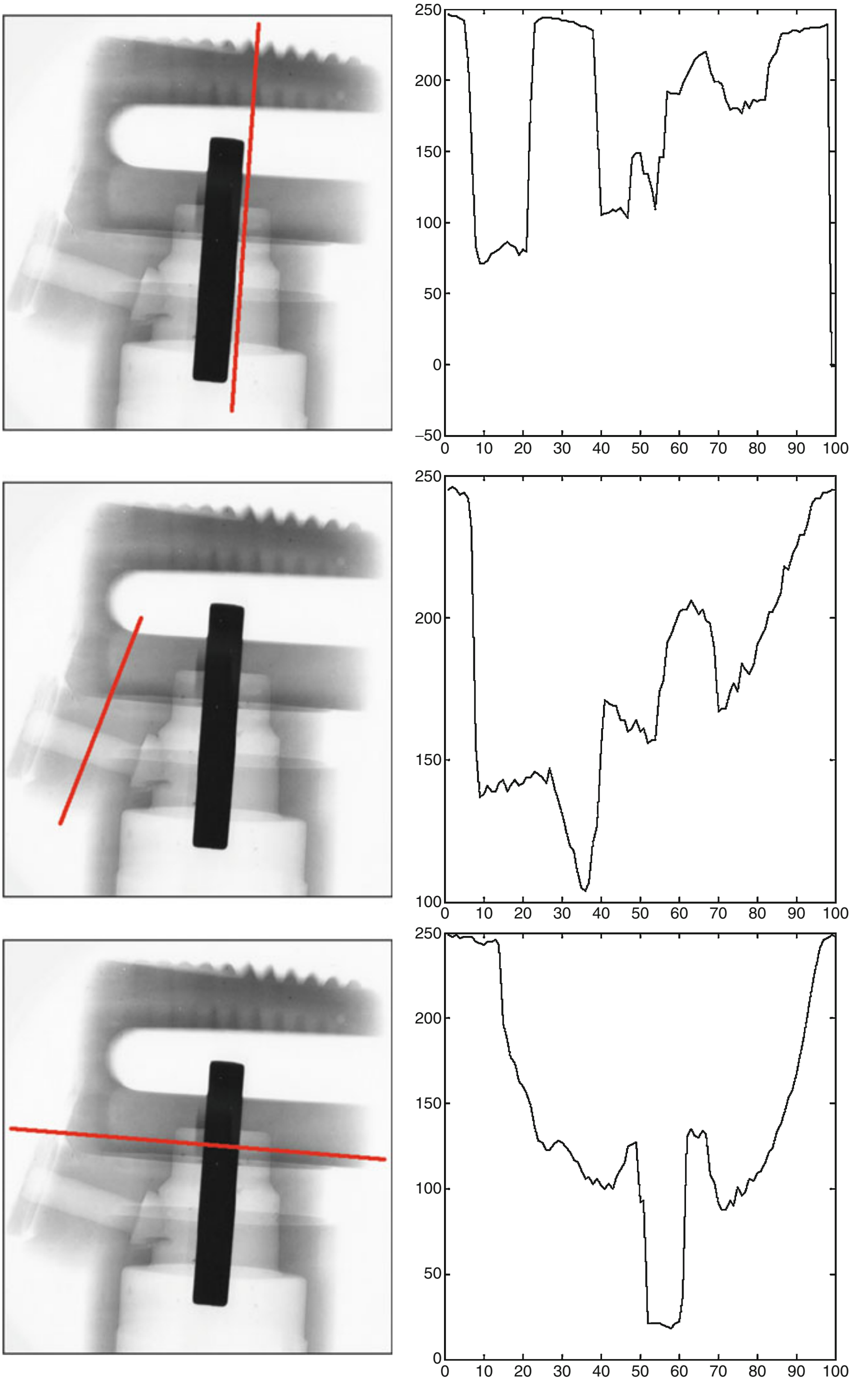


Fig. 12.9
Aerosol spray assembly. Scan lines (*red*) and intensity profiles (*right*)

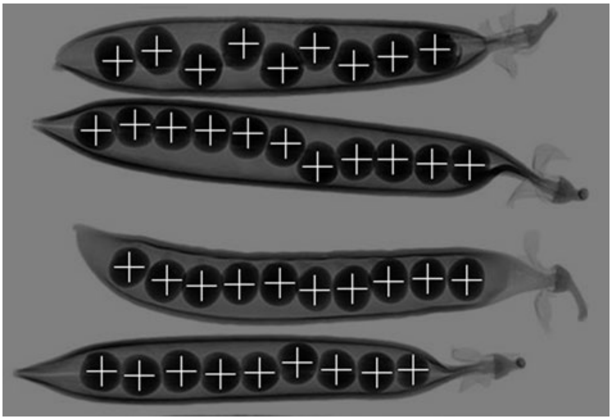


Fig. 12.10

Locating and counting peas in their pods. The algorithm can be adjusted to detect peas within a given range of sizes. [QT: wri, enc, thr, neg, ero(10,0), shk, crs(10), rei, swi,adi, mxi]

adjust the position of these scan lines relative to easily discernable edges, or other features. Binary and grey-scale morphology can also be used to perform fairly crude existential inspection (➋ *Fig. 12.10*). Further applications are illustrated in ➋ *Figs. 12.11–12.13*.

12.3.1 Modelling X-Ray Absorption

In simple situations, such as that illustrated in ➋ *Fig. 12.9*, there is little difficulty in understanding the structure of the X-ray image; each step change in intensity can be attributed to an easily identifiable component edge within the assembly. However, this is not always the case; the image structure may be very complicated indeed. (The reasons for this are listed later.) In this event, it may be helpful to generate a synthesized image that predicts what the X-ray image will look like. Understanding and analysing X-ray images automatically can both be helped considerably by the judicious use of computer graphics. In some cases, synthesising a low-resolution image is sufficient, to help an engineer to understand why the X-ray image appears as it does (➋ *Fig. 12.16*). In some applications, it may be necessary to use a very detailed synthesised image as a "golden sample" that is then compared to the real X-ray images. In this short section, we can only indicate the broad nature of the physical model underlying image synthesis.

Consider a single monochromatic X-ray, of initial intensity I_0, travelling normally through a slab of homogenous material of thickness X and with X-ray *absorption coefficient* μ. The intensity of the emergent ray is given by the *Beer equation*:

$$I_0\exp(-\mu X) \tag{12.7}$$

Notice that, when the ray is travelling through the block at an oblique angle, the path length is bigger, so the attenuation is larger (➋ *Fig. 12.14*). If the angle of incidence is θ, the path length is equal to X/cos(θ). In this case, the intensity of the emergent beam is reduced to

$$I_0\exp(-\mu X/\cos(\theta)) \tag{12.8}$$

In what follows, we shall consider normal incidence, unless otherwise stated.

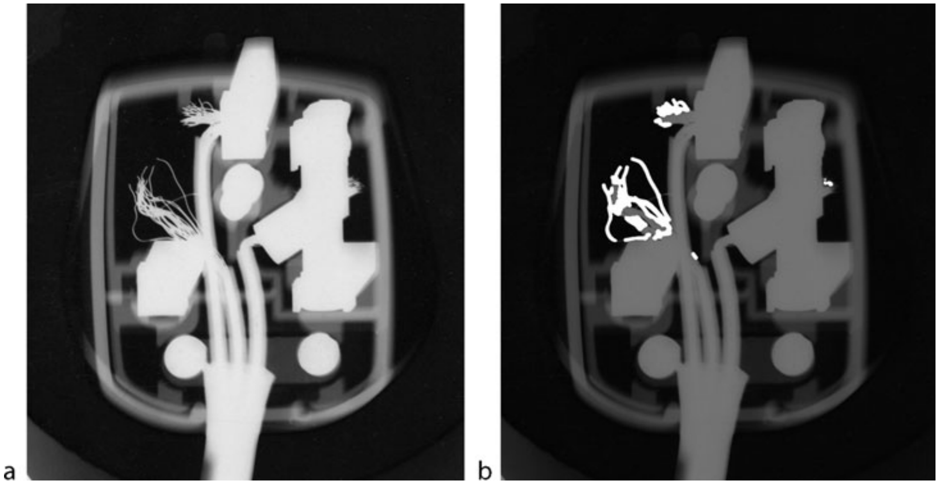


Fig. 12.11
Electrical power plug with faulty wiring. (**a**) Original image. (**b**) Thin bright streaks (strands of bare copper wire) detected using the QT crack detector, *crk* (Chaps. 19 and 21)

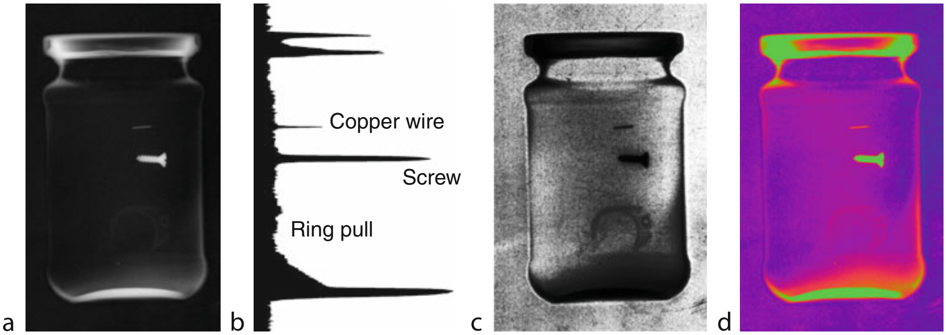


Fig. 12.12
Jar of baby food. (**a**) Original image, negated for clarity. Three foreign bodies have been introduced for testing: short length of copper wire, steel screw and an aluminium ring-pull (almost invisible). (**b**) Intensity profile along the vertical scan line. Notice the high contrast of the copper wire, despite its small diameter. However, the ring-pull does not produce an obvious change in intensity. Notice that this graph also indicates the thickness of the side-wall and, in particular, the distribution of glass around the base and neck-shoulder regions. (**c**) Contrast enhanced to demonstrate that the ring pull really is there! (**d**) Pseudo colour often helps to improve visibility

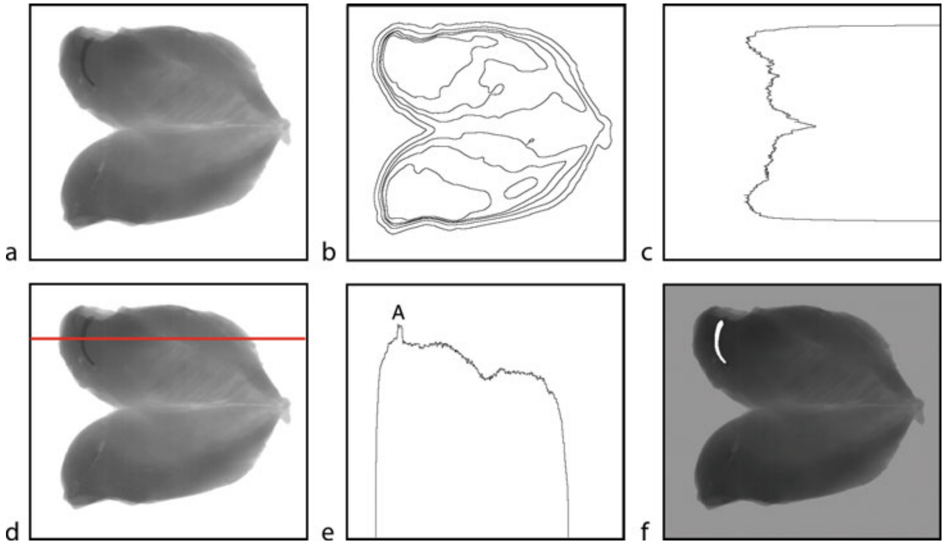


Fig. 12.13

Analysing chicken breast meat. (''Butterfly'') (a) Original image. Source: Spectral Fusion Technologies Ltd. (Birmingham, B46 1JT, UK) (b) Isopohotes. (c) Intensity profile through the centre of the image. (d) Scan line through the crescent-shaped bone. (e) Intensity profile along the red scan line. Notice the low contrast between the bone (A) and the meat. (f) Bone detected using morphology. Although it is effective in this instance, the algorithm used here is not robust enough for practical use

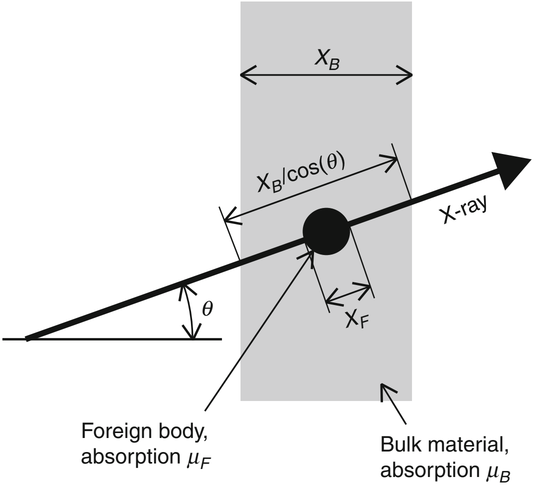

X_B
$X_B/\cos(\theta)$
X-ray
θ
X_F
Foreign body, absorption μ_F
Bulk material, absorption μ_B

Fig. 12.14

An X-ray travelling at an oblique angle through a block of absorbing material containing a spherical foreign body of diameter X_F encounters a path of length ($X_B/\cos\theta$) and will therefore be attenuated more as θ increases. However, the contrast, as defined by Eq. 12.13, depends upon ($\mu_F - \mu_B$) and X_F but is independent of θ

Now, consider a ray travelling through a multi-layer structure, in which the individual layers are of thickness X_i (i=[1,...,N]) and have absorption coefficients μ_i (i = [1,...,N]). The emerging X-ray intensity is given by the following product:

$$I_0 \prod_{i=1}^{N} \exp(-\mu_i X_i) \tag{12.9}$$

or in another form

$$I_0 \exp\left(\sum_{i=1}^{N}(-\mu_i X_i)\right) \tag{12.10}$$

This analysis can, of course, be extended to describe the absorption of a ray travelling through continuously varying medium. However, for our present purposes, ➲ Eqs. 12.9 and ➲ 12.10 will suffice.

Let us now consider a special case: a small piece of material (i.e., a foreign body) of thickness X_F and with absorption coefficient μ_F, is embedded within an homogenous block of overall thickness $X_B(X_B>>X_F)$ and absorption coefficient μ_B. (This is illustrated in ➲ *Fig. 12.14* with the angle θ equal to zero.) The intensity of the emergent beam is

$$I_F = I_0\exp(-\mu_B(X_B - X_F))\exp(-\mu_F X_F) \tag{12.11}$$

If no foreign body is present, the emergent beam intensity is clearly

$$I_B = I_0\exp\left(-\mu_B X_B\right) \tag{12.12}$$

Using ➲ Eqs. 12.11 and ➲ 12.12, we can define a measure of the local contrast (C), caused by the presence of a foreign body. We define C as

$$C = |I_B - I_F| \div I_B \tag{12.13}$$

which reduces to

$$C = |(1 - \exp(\mu_\mathrm{B} - \mu_\mathrm{F})\mathrm{X}_\mathrm{F})| \tag{12.14}$$

The contrast ratio defined in this way is independent of the initial intensity, I_0 and the block thickness (X_B) but varies with the difference in the absorption coefficients of the bulk material and the foreign body ($\mu_B - \mu_F$).➲ Equation 12.14 also shows that larger foreign bodies produce an increased contrast compared to small ones with the same material composition (i.e., same value of μ_F.) The corollary of this is that a wedge-shaped foreign body may not be detectable in its entirety, since the contrast is low at the "thin end". For the same reason, the diameter of a spherical foreign body may be underestimated.

This assumes that the foreign body and the bulk material have the same spectral response. So far, we have assumed that both μ_B and μ_F are both dependent upon the X-ray wavelength, λ. ➲ Equation 12.14 describes the worst-case situation, in which the spectral variation of μ_B and μ_F are exactly the same. To obtain a more precise estimate of the contrast ratio, we should include the following:

- Spectral emission characteristic of the X-ray source, G(λ)
- Spectral absorption characteristic of the bulk material, $\mu_B\left(\dot{\lambda}\right)$
- Spectral absorption characteristic for the foreign body, $\mu_F\left(\dot{\lambda}\right)$
- X-ray-to-VIS conversion efficiency of the scintillator, S(λ,Λ). This is a 2-dimensional function, since there is a different VIS emission spectrum for each incident wavelength λ. Λ is the VIS wavelength.)
- Spectral sensitivity function of the camera (i.e., the VIS image sensor), $C(\Lambda)$

The camera output is then given by

$$\int_{\Lambda}\int_{\lambda} G(\lambda).R(\lambda).S(\lambda,\Lambda).C(\Lambda).d\lambda.d\Lambda \tag{12.15}$$

R(λ) is a generalised version of ➋ Eq. 12.9, viz:

$$\mathrm{R}(\lambda) = \mathrm{I}_0.\exp(-\mu_B(\lambda).(X_B - X_F)).\exp(-\mu_F(\lambda).X_F) \tag{12.16}$$

For most practical purposes, we can estimate the absorption coefficient ($\mu_Z(\lambda)$) for an element of atomic number *Z*, at wavelength λ, using the *Bragg and Beer Equation*:

$$\mu_Z(\lambda) = \mathrm{K}\lambda^3\mathrm{Z}^3 \tag{12.17}$$

K is a constant depending on the element's specific weight. A mixture, or compound, containing elements with absorption coefficients σ_1, σ_2, σ_3,... in proportions P_1: P_2: P_3: ... has an overall absorption coefficient equal to:

$$\sum_i P_i\sigma_i \tag{12.18}$$

Details of X-ray absorption spectra are given for elements 1–92 in [7]. Sample spectral absorption curves are given in ➋ *Fig. 12.15*.

These formulae are, of course, just part of the computation needed to simulate a real X-ray system. Additional calculations are needed to derive the path lengths, X_B and X_F, for rays covering the whole cone occupied by the X-ray beam. These are standard geometric calculations but it must be remembered that the beam intensity diminishes according to the inverse square law.

Examples of synthetic X-ray images are given in ➋ *Figs. 12.16* and ➋ *12.22*. In the former case, they can be compared to the images from an experimental X-ray system. In the second instance, the synthetic X-ray images actually showed that it would not be worthwhile building a system with the particular layout in question because the images are too complicated to analyse.

12.3.2 Why X-Ray Images Are Complicated

Divergence of the X-ray beam causes one of the major complicating features in X-ray systems. This and other factors causing further difficulties are listed below.

- *Non-homogenous beam* The intensity varies across an X-ray beam, is normally greatest in the central region and becomes smaller towards its edge. There are two contributary factors. First, the energy transmitted by the X-ray tube is greatest towards the centre of the beam and decreases towards its periphery. Secondly, the beam divergence spreads the available energy unevenly across the cone (➋ *Fig. 12.17*).
- *Variable magnification* When two identical objects, are placed at different points along the X-ray path, the one closer to the source appears to be larger (➋ *Figs. 12.18*, ➋ *12.23* and ➋ *12.24*).
- *Parallax errors* Parallax distortion occurs as a result of beam spreading (➋ *Fig. 12.19*). For example, two component-lead connecting pads, on opposite sides of a printed circuit board, sometimes create an 8-shaped X-ray shadow. A parallel beam would form a simple

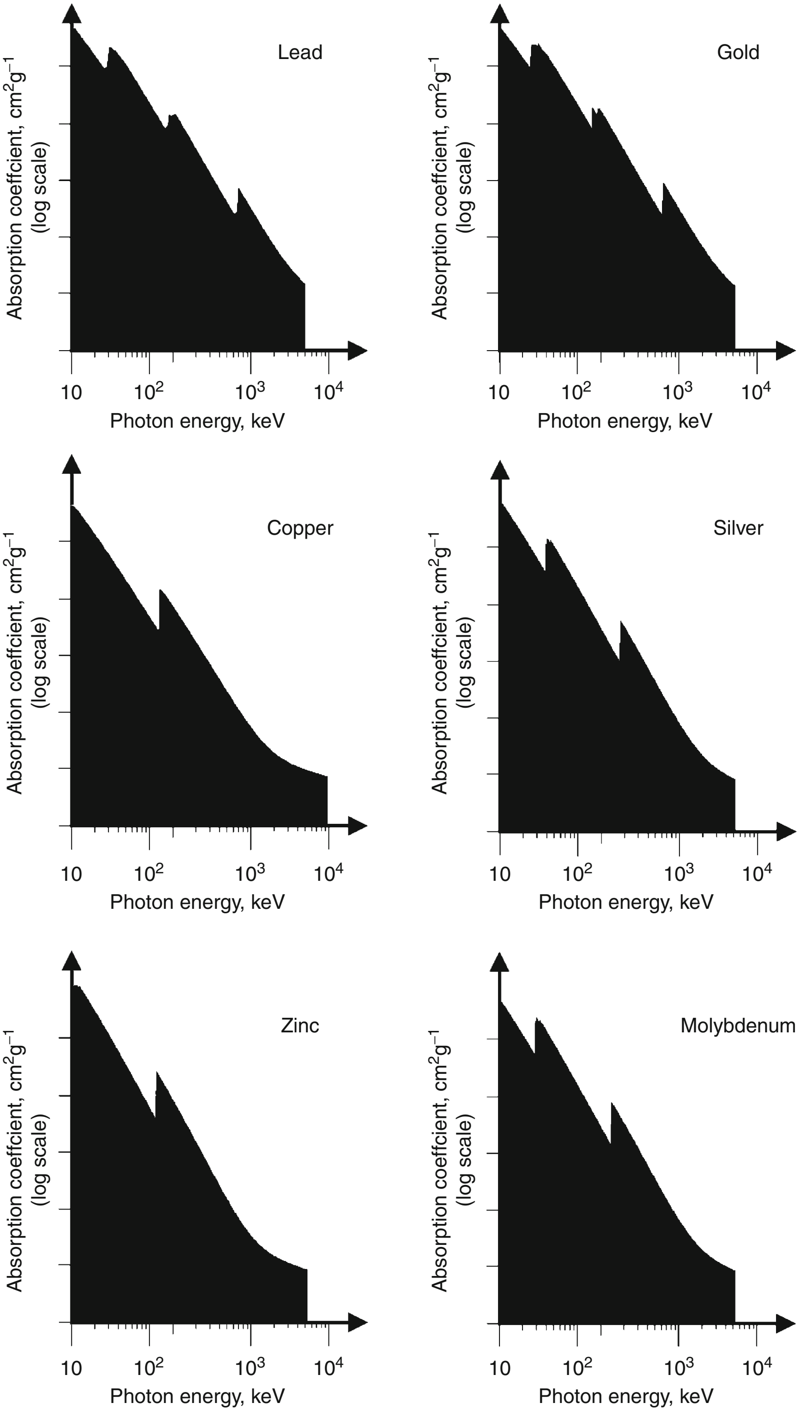


Fig. 12.15 (Continued)

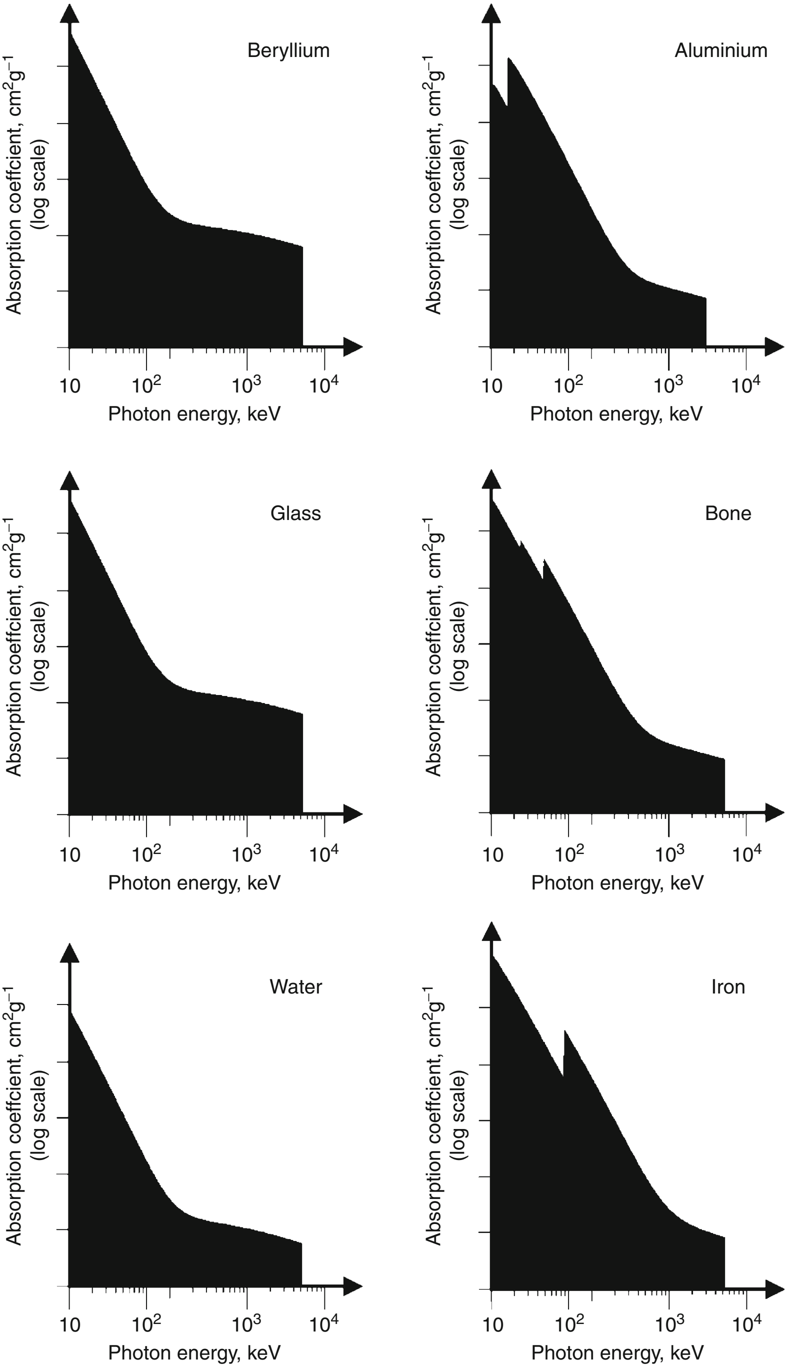


Fig. 12.15

X-ray absorption spectra (two previous pages)

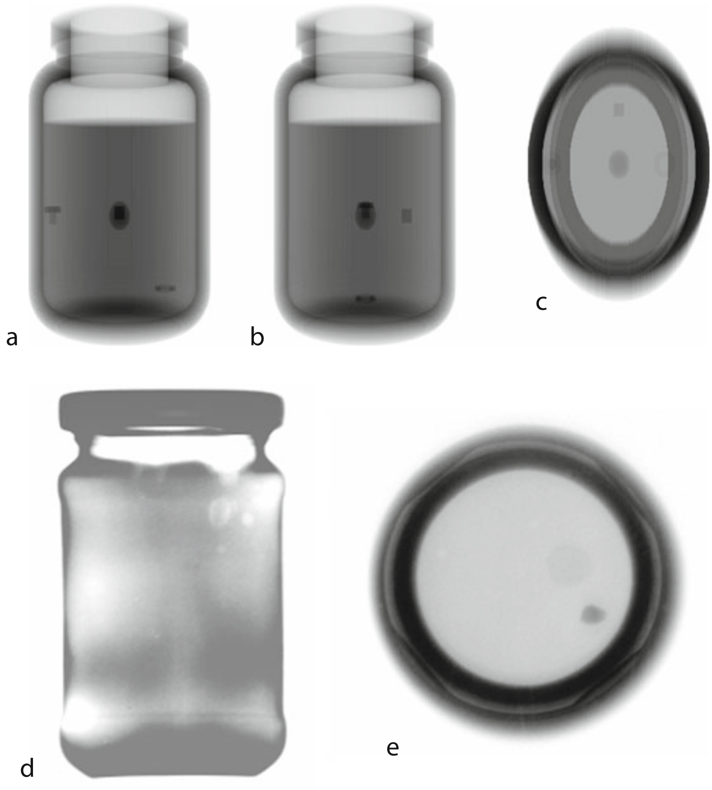


Fig. 12.16

Synthetic X-ray images of a glass jar containing a liquid and four foreign bodies. (a) Side view. (b) Front view. (c) Plan view. The washer, near the base of the jar, cannot be seen easily in the front view. The screw cannot be seen properly in the plan view and is totally obscured in the front view by the sphere. Similarly, the cube is obscured in the side view. (d) and (e) Real jar for comparison. There three foreign bodies. One of them is invisible in the plan view. Variations in the thickness of the glass wall complicate the picture. This jar is fitted with a metal cap

circular ring. Furthermore, conductor tracks that are aligned but on opposite sides of the board become separated, if they close to the edge of the beam.

- *Warping across the image* Consider a grid, or regular array of spots, inclined to the central axis of the beam (➲ *Fig. 12.20*). A distorted grid is projected onto the scintillator screen, since the diverging beam magnifies features close to its edge more than those near the centre. As a result, it may be difficult to register an X-ray image to that from a "golden sample". In addition, it is difficult to measure point-to-point distances precisely.
- *Blurring* Since the X-ray source is not a true mathematical point, images are always blurred to some extent; the larger the spot on the anode targeted by the electron beam, the greater is the blurring effect. So-called *micro-focus* X-ray sources are designed specifically to minimise blurring. Provided the spot size is known and constant, deconvolution software can be used to compensate for blurring but this would not normally be the preferred solution,

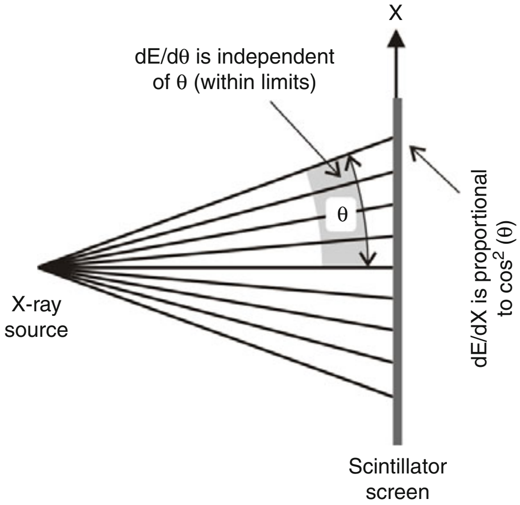

X
dE/dθ is independent of θ (within limits)
θ
X-ray source
dE/dX is proportional to cos² (θ)
Scintillator screen

Fig. 12.17
The diverging beam produces an uneven energy density on the scintillator

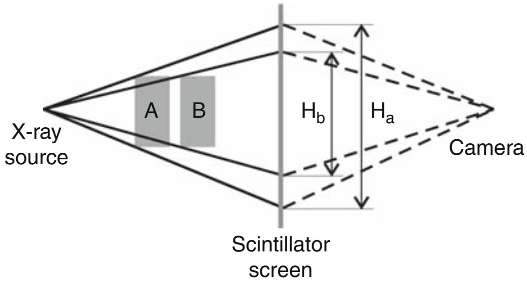


Fig. 12.18
Variable magnification due to beam spreading. H_a is the height of the shadow of object A, while H_b is the apparent height of B. Since A is closer to the point source of X-rays, $H_a > H_b$

since it is computationally expensive and rarely achieves the precision needed for automated analysis.

- *Superimposition of feature shadows* Information about all of the features lying along the path of a given ray is reduced to a single number. This is the implication of Eqs. 12.9 and 12.10. The 3-dimensional structure of the sample is effectively compressed into a single planar image. For example, in the case of the aerosol spray assembly (*Fig. 12.9*), there are several plastic components and a metal pin contributing to the image. Their shadows are superimposed. Notice that any features lying behind the metal pin, which is effectively opaque to X-rays, are lost.
- *Back-scatter* X-rays do not behave exactly in the simple manner described so far. A small proportion of the X-ray energy is scattered, rather than being transmitted or absorbed. This process is akin to the scattering of VIS radiation within a transparent medium such as glass. Scattering of X-rays can lead to degradation of the image quality.
- *Noise* Camera and photon noise both contribute to degradation of image quality.

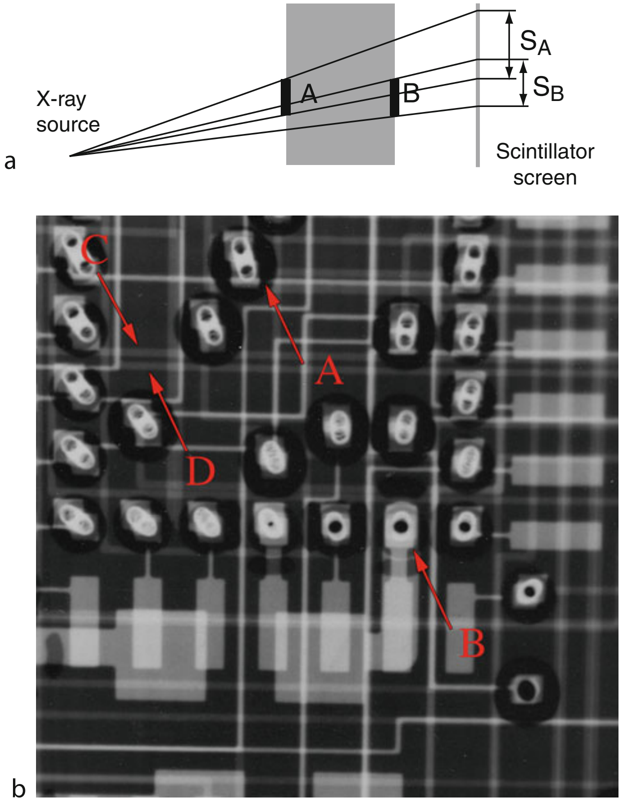


Fig. 12.19

Misalignment of X-ray shadows as a result of parallax. (a) Objects A and B are aligned but are separated along the central axis. S_A is the shadow of A. Similarly, S_B is the shadow of B. S_A and S_B do not coincide exactly and possibly not at all. (b) Printed circuit board showing marked parallax effects: the shadows of two ring-like pads are not aligned near the edge of the image and produce a combined shadow in the form of an 8 (A) However, near the centre the pads overlap to resemble an O. (B). In the same way, overlapping conductor tracks do not appear to be aligned near the edge (C and D)

12.4 Dual Axis X-Ray System

In this section, we describe a *Twin Orthogonal Fan-beam* (TOF) x-ray system for detecting foreign bodies in multi-package food products. The geometric organisation of one such system is explained in *Fig. 12.21*. Other physical arrangements were considered but were quickly discounted, as they were not found to confer any significant advantages. (See for example *Fig. 12.22*.) The work reported in this section was conducted as part of an experimental project funded by the European Union [8]. The objective was to advance AXI system technology across a broad front: from sensors to algorithms. We shall limit our discussion here on the overall concept development and how image processing algorithms were developed specially for the particular type of data that a TOF system generates. This is, once more, an

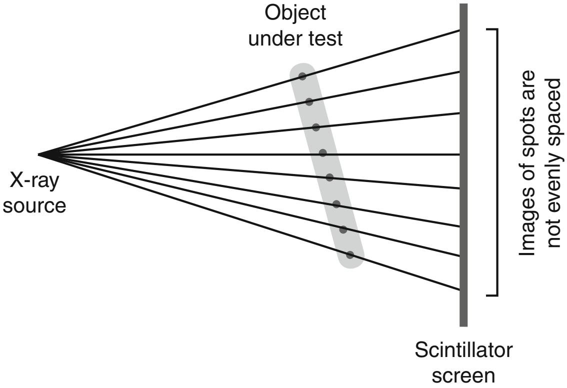


Fig. 12.20
Distortion due to beam divergence

exercise in systems thinking; the physical layout of the image acquisition sub-system creates distinctive data types that require special algorithms to extract the maximum amount of information.

12.4.1 Image Acquisition

In a certain food-processing plant, six empty glass jars are delivered simultaneously, on a cardboard tray, to a multi-nozzle filling station. At no time are the jars ever available separately for filling, capping, or inspection; they must be examined in situ, alongside the other jars on the tray. In the TOF system, two images are digitised simultaneously from line-scan sensors and are then pre-processed separately. These line-scan sensors are aligned precisely, the significance of which will be explained later. The image pair is then processed to locate foreign bodies and operate an *accept/reject* mechanism that diverts the whole tray appropriately. Information about the location (in 3D space) of any foreign bodies is also calculated.

In the side view (*Figs. 12.23* and *12.24*), approximately 90% of the volume of the product is clearly visible, unobscured by the rim and shoulders of the jars. These regions of the side-view image are characterised by having a smoothly varying intensity profile. That part of the product that overlaps the heavy shadows caused by the jar is much more difficult to analyse and therefore requires quite sophisticated image processing techniques. The plan view generates an even more complex image and only 60% of the product volume remains unobscured. The cardboard tray is effectively transparent to the x-rays and, hence, can safely be ignored during the inspection process.

Understanding the Images When we obtained the first TOF image pairs, we were puzzled by their complex form. To gain a greater understanding of their structure, several pairs of x-ray images were synthesised, using a ray-tracing program based on the idea implicit in Sect. 12.3.1. This exercise provided some valuable insight, enabling us to:

- Advise our collaborators who were building a TOF x-ray system.
- Predict how various alternative TOF systems would behave, without ever building them (*Fig. 12.22*).

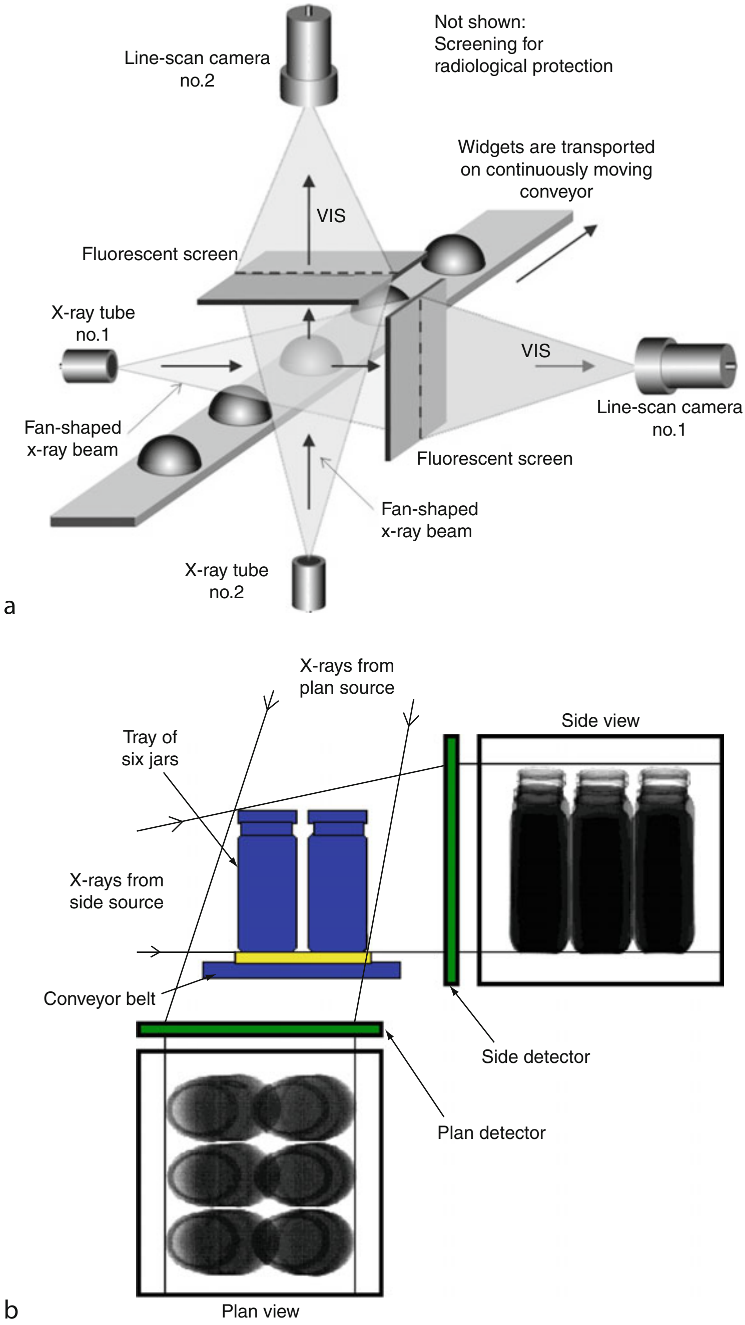


Fig. 12.21

Twin orthogonal fan-beam x-ray inspection system. (a) Layout. (b) Image formation. The system was designed to inspect six glass jars simultaneously. These are held on a cardboard tray. The jars contain tomato sauce. Sample image pairs generate by such a system are presented in *Figs. 12.16* and *12.23*

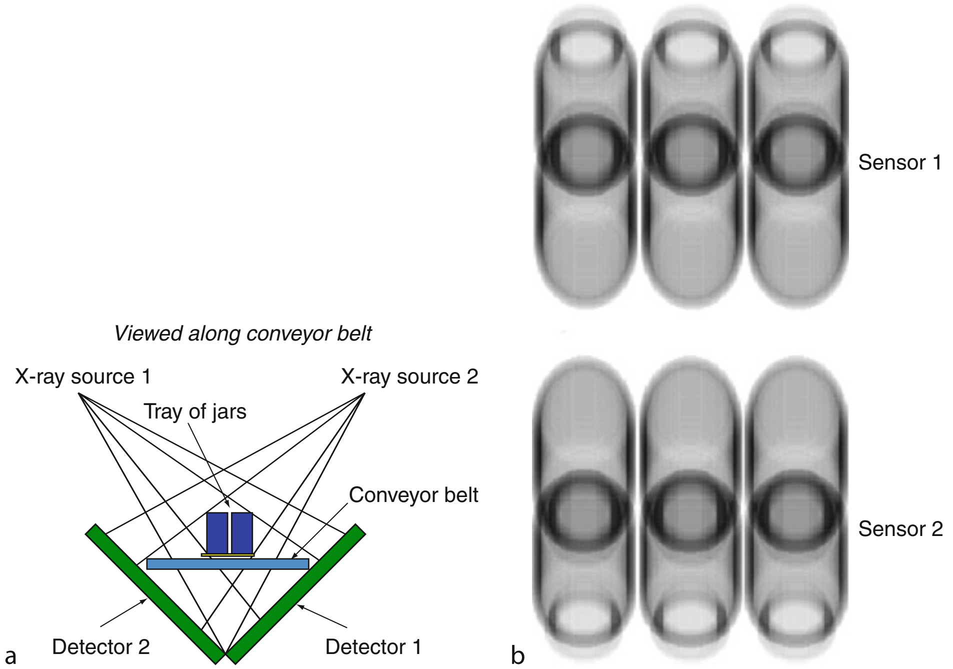


Fig. 12.22
Alternative design for a TOF x-ray inspection system. (a) Layout. (b) Sample image pair (synthesised images). As a result of seeing these images, the idea of building an x-ray system like this was abandoned

- Identify that part of the product that is clearly visible to just one, or neither, of the x-ray sensors.
- Devise procedures for analysing "real" images (i.e., those derived from the TOF x-ray rig).
- Assist in the debugging the x-ray rig.

Since this is a line-scan system, beam divergence affects only one axis. (Vertical axis in the plan view: across the conveyor. Vertical axis in the side view: normal to the conveyor.) In the side view, superposition and beam divergence cause particular problems in the neck/shoulder region. This complicates the image analysis, so that new techniques had to be devised [9]. The plan view shows a composite shadow of three pairs of jars. In this case, beam divergence makes the mouth of each jar appear ovoid.

Interpreting Image Pairs An essential feature of the TOF x-ray systems illustrated in *Figs. 12.21* and *12.22* is that the side- and plan-images are aligned precisely along the X-axis. This makes their joint interpretation much easier. (*Fig. 12.23*) Hence, the presence of a foreign body at position (X, Z) in the side view image suggests that there should be a foreign body evident at a point (X,Y), for some value of Y, in the plan view, and vice versa. However, certain regions of both images are problematical, notably the neck-shoulder regions in both images and the base regions in the side view. Standard image filtering methods, such as convolution and morphology, can be used to look for intensity anomalies due to foreign bodies but the results must be interpreted in an intelligent way. Making inferences about one

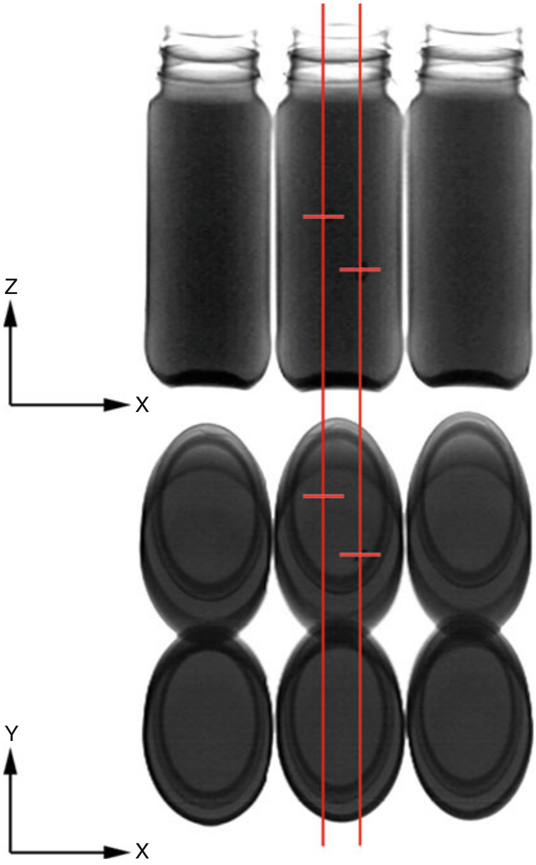


Fig. 12.23
The plan and side view images of a TOF pair are aligned exactly. Hence, a suspected foreign body at position (X, Z) in the side view indicates that there should be a foreign body at (X,Y) in the plan view, for some value of Y. This allows suspicions gained from one image to be confirmed/refuted with reference to the other. The jars move along the X-axis, so beam divergence has no effect in this direction

picture based on what has been found in another image requires a level of intelligence that cannot be represented easily in an image processing package such as QT (Chap. 21) or NeatVision (Chap. 22). This type of task is well suited to a language that combines image processing and Prolog-style reasoning (Chap. 23)

In view of the complexity of the TOF images and the low contrast of many foreign bodies, the author and his colleagues decided to investigate the use of template matching. This technique is usually derided because it is sensitive to position, orientation and scale changes. *Figure 12.24* shows the result of simply subtracting a reference image, obtained by shifting the test image, from the test image before shifting. Provided the difficulties just mentioned can be overcome, image subtraction can provide a useful way to improve the visibility of foreign bodies. It is effective even when processing images of a complex container, provided it is of fixed

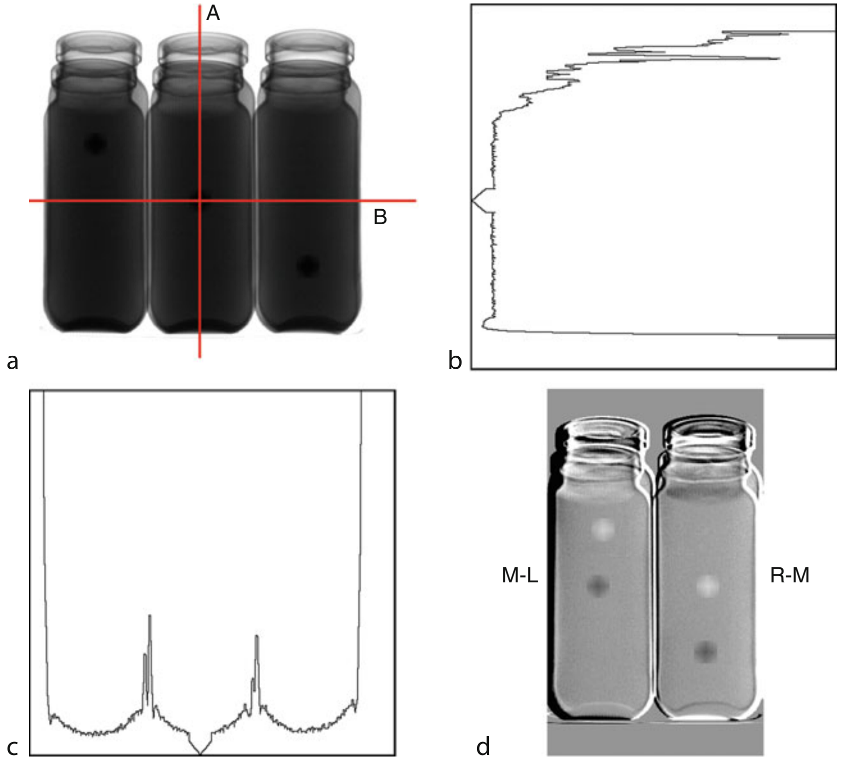


Fig. 12.24

Improving the visibility of foreign bodies by subtracting a reference image. (a) Original image. Six jars are arranged in three pairs, each pair aligned with the system's central axis. Three low-contrast foreign bodies have been simulated, one for each pair of jars. (b) Intensity profile, along the red vertical line (A) through the simulated foreign body in the middle pair M. (c) Intensity profile, along the red horizontal line B. (d) For the purposes of illustration, the middle pair (M) provides the reference image. The original image was shifted to the left by a distance equal to the width of one jar. This result and the original image were then subtracted and the contrast increased by histogram equalisation. M–L: the image of the left-hand pair (L) subtracted from that of M. R–L: the image of M subtracted from that of the right-hand pair (R)

material composition, and has known shape, size, position and orientation. Image registration might involve some complicated processing. It may even be necessary to perform local feature matching repeatedly, to accommodate image variability. In their research, the author and his colleagues used a deformable reference image, which was subtracted from the test image. The warping process was guided by contours extracted by following edges with a high intensity gradient. A degree of intelligence is needed to do this, because edge contours cross one another, due to superposition. The deformable model was required to accommodate the fact that the images may be warped, because the jars are examined in a different part of the diverging beam. By subtracting the carefully deformed reference image, it is possible to increase the contrast of foreign bodies and hence make their detection more reliable. However, the computational process is complicated and slow. This was a research study and resulted in new ideas but, so far, has not lead to a factory installation.

12.5 Practical Issues and Future Trends

So far, we have concentrated on the fundamental technical and conceptual features of an x-ray inspection system. In addition, there are many other systems issues that must be taken into account; no machine would be truly successful if these were neglected. An x-ray system that is designed with only theoretical performance in mind would almost certainly be unreliable, unsafe, and unhygienic. The fact that theory, or laboratory-based experimentation, predicts that it is possible to detect a specific foreign body, does not necessarily mean that a practical x-ray inspection system can be built that is able to do so reliably in a food processing factory. A wide range of factors affect the technical feasibility and commercial viability of building and operating an x-ray system see ➲ Sect. 12.6.

12.5.1 Redirecting X-Rays

It would be a great advantage if x-rays could be focussed and reflected like visible light, to avoid beam divergence. This would make image processing in AXI systems very much simpler. Building large detectors would also be very useful but depends on our ability to construct large lenses or mirrors for x-rays.

A highly polished metal surface will reflect an x-ray beam if the angle of incidence is above about 89.7°. A very smooth, very narrow gently-curved groove in a block of copper will act as an x-ray wave-guide. Lenses and collimators can be built on this principle to redirect an x-ray beam but they require very precise machining and alignment. Also see Ref. [10].

Diffraction focussing of x-rays is possible but will only work properly for narrow-band sources. A third method, is to use a very pure curved crystal as an x-ray lens. Recent work has shown that it possible is to build a refracting lens using an array of extremely small lenses. These might be hollows in a block of aluminium [11]. Using ideas like these it may, one day, be possible to generate parallel, or even convergent, beams. However, at the moment, it is too difficult/expensive to incorporate them in a practical shop-floor AXI system.

12.5.2 X-Ray Lasers

Designing an AXI system would be very much easier if we could build x-ray lasers cheaply. A monochromatic x-ray laser with a very narrow "pencil" beam would be very useful, as it could be incorporated into a flying-spot scanner (see ➲ Chap. 9) A laser source would be a great advantage for another reason, it would have a well-defined and possibly controllable wavelength capable of discriminating between different materials very much more easily than a broad-spectrum source can. If the beam intensity is high enough, there are two additional possibilities for creating images: using x-ray back-scatter and x-ray fluorescence. If it were possible to build an array of tiny x-ray lasers, AXI would truly be revolutionised.

12.5.3 Improving Image Processing

The ability to analyse x-ray images is not as well developed as we would like. A high level of intelligence is needed to detect foreign bodies because x-ray images are inherently complicated. Moreover, they are often unpredictable, particularly when inspecting natural products/materials. As we have seen in the case of the TOF images, new algorithms had to be invented. This is also the case for applications where image subtraction is appropriate, since image registration algorithms are problem specific. The development of image processing algorithms that are guided by a CAD model would also be beneficial, particularly for applications in high-precision manufacturing. This may involve image synthesis, as we have illustrated, or automatically generating "driving instructions" for an image proccessor.

12.5.4 Reducing Costs

The high cost of present-day x-ray inspection system is due, in large part, to the engineering effort needed to design a special system for every individual application. The cost could, of course, be reduced significantly, if large numbers of identical machines could be built. Unfortunately, the market has not yet reached the critical point at which repeat orders make mass-production of x-ray systems a commercial reality. The high cost inhibits investment in the technology, which limits the savings that can be made by copying existing designs. The principal equipment costs are incurred for mechanical handling, environmental protection, x-ray beam generation, x-ray shielding, x-ray sensing and image processing. It is difficult to envisage major reductions in cost for any of these. However, if x-ray lasers became a practical option, it might be possible to reduce the cost of shielding and the physical size of a system considerably, since the beam would be better controlled.

12.5.5 Final Remarks

Industry has not really exploited AXI to its potential. While it is somewhat limited by existing technology, many more applications could benefit from its use. Lack of understanding and confidence in AXI technology are probably two of the most important reasons for this. There is a need for better education about the capability, achievements and potential of AXI. There is also a need to reassure managers and shop-floor personnel about the safety of properly engineered x-rays systems.

12.6 Supplement

12.6.1 Factors Affecting Performance and User Acceptance

The following list indicates just some of the major factors that influence the success of an x-ray inspection system. There are many subtle interactions between the various parts of a system and neglecting these may mean that it will fail to satisfy the customer.

Speed: An automated x-ray system must be fast enough to match the product throughput rate. This is normally dictated by the food-processing machinery and cannot be reduced merely to accommodate the needs of the inspection system. Speed is defined using two parameters: latency (i.e., the time between capturing an image and calculating a decision about it.) and throughput rate (measured by the number of objects inspected in unit time, i.e., seconds or minutes). In addition, the speed that the product is moved past the x-ray source and sensor is determined by that of the conveyor belt, which is, in turn, fixed by the production process.

X-ray beam intensity: It may be necessary to increase the system sensitivity, to improve its operating speed. This can be achieved by increasing the beam intensity but this reduces the tube life and may generate an unacceptable and dangerously high level of radiation. (This parameter is controlled by the tube beam current.)

Beam energy: This parameter is measured by the accelerating voltage (i.e., anode-cathode voltage) applied to the tube. It effectively defines the peak x-ray wave-length.

Mechanical sub-system: Must be robust and easy to maintain and clean. This is, of course, particularly important in the food and pharmaceutical industries. "Strip-down" times of less than one minute are needed in some industries, to meet stringent hygiene standards, without causing undue disturbance to the production process.

Robustness: Care must be taken to design a rugged machine, as a food factory can be a very hostile environment for electronic and mechanical equipment.

Human factors: Workers who will be required to operate the x-ray equipment must be considered to be an integral part of the inspection process. If a machine is too complicated to operate, or it requires continuous adjustment or supervision, it will not be effective and won't be used for long. Equipment operating in a food-handling environment must be easily cleaned. There is no point in eliminating a foreign body problem, only to create a microbiological hazard.

Foreign body: Material (determines the x-ray density), size, shape, location within the inspected product, type (often varies in a predictable way within the product), orientation.

Product: Size, shape, orientation, materials (Determines the x-ray density. Moisture or fat content may vary), internal structure, relative positions of internal features (may vary from one sample to another, e.g., bones within a carcass).

Maintenance: The system should be easy to maintain and repair, with comprehensive self-diagnosis software.

Calibration: There should be a well-defined calibration procedure. Any calibration targets must be easy to keep clean and be stable over a prolonged period of time. Moreover, they must thoroughly test those functions of the machine that are critical to its proper operation.

Hygiene: It should be possible to strip down and clean all parts of the inspection machine that are likely to come into contact with food materials/products, within a few seconds. It should be completely waterproof, so that it can withstand regular hosing down. Antibacterial sprays and washing agents should not damage the machine.

Safety: X-rays are dangerous. Legal safe-guards exist to protect workers from exposure to penetrating radiation. This imposes certain constraints on the design, such as fail safe operation, restrictions on access when the system is generating x-rays, as well as operator training and health checks.

Environmental: The system should be able to withstand the hostile environment found in a factory environment, which may be hot, cold, damp, or dusty. It should be able to withstand wide variations in temperature, as well as electrical noise generated within the factory.

Packaging: Product packaging can have a significant effect on detection performance. For example, a small splinter of glass that is easily detectable in a loosely packed product may be much more difficult to identify inside a glass container. There are two reasons for this: (a) Image contrast is reduced because the container absorbs x-rays. (b) The complex geometry of the container makes the analysis of x-ray images more difficult.

Process requirements: A system must be designed for a specific purpose, taking into account all of the requirements imposed by the customer company's working practices, existing equipment, legislation, etc.

Commercial/economic: The perceived pay-back period for any capital outlay is critical in deciding the success of a project.

Work practices: The system should complement, not radically change existing inspection and manufacturing protocols.

Resolution: The smallest object that can be resolved is, in practice about 3–5 times larger than a pixel. For a system using a line-scan camera and a conveyor belt, the resolution along the web is controlled by the camera clock speed and the belt speed. The resolution across the belt is dictated by the physical spacing of the photo-detector elements. The resolution along the direction of travel is dictated by the sampling speed of the array read-out electronics. The resolution in these two directions should be the same, to avoid changes in the apparent size of an object as it is rotated.

Conveyor belt speed: The speed of the conveyor belt should be regulated, to avoid altering the image resolution.

Performance: See *Balancing rejection rates.*

Production run: This affects the economics of the building and operating an x-ray inspection system.

Wash-down procedure: For reasons of hygiene, many x-ray systems are designed to withstand washing with a high-pressure water jet, as well as cleaning chemicals. Hence, both mechanical and electrical elements must be enclosed in fully water-proof housings.

Temperature variations: The x-ray power supply generates considerable heat. Large temperature variations can occur inside the cabinet, especially if the equipment is sealed inside a water-proof cabinet. This can lead to drift in the x-ray detector output. To stabilise the internal temperature inside the cabinet, it may be necessary to provide air conditioning.

Balancing rejection rates: In any inspection process, there is a trade-off between false-acceptance and false-rejection levels. For some products, such as finished packaged goods, the false-rejection rate must be limited (perhaps less than 1%), to avoid discarding high-value product that is perfectly acceptable to the end customer. In these circumstances, it may be worthwhile manually inspecting all product that the x-ray system has rejected. The x-ray system can then be tuned in favour of rejecting more "suspicious" product; much higher false-rejection levels (perhaps 3–5%) might then be acceptable.

Reliability of the rejection mechanism: Mechanical accept/reject devices have a limited reliability, particularly when they are used to move products that are sticky, fibrous, gritty, dusty, or for some other reason, likely to cause jamming. Accept/reject actuators are exposed to an uncontrolled, dirty environment, so they must be designed with great care and serviced regularly. It is foolish to spend a great deal of effort to improve the performance of an inspection algorithm only to neglect the mechanical handling sub-system.

Acknowledgements

Much of the material in this chapter was supplied by Dr. Mark Gravess, previously of Spectral Fusion Technologies, Birmingham, B46 1JT, UK, and was included in our joint book [13].

Much of the work on Twin Orthogonal Fan-beam (TOF) x-ray inspection systems was conducted by Stephen C. Palmer, working with the author on the project entitled "*Automated 3D X-ray Inspection System for On-line Foreign Body Detection in Food Products*". This was funded under the Agro Industries Research (AIR) programme of the Commission of the European Communities – Directorate General XII. Reference number AIR2-CT93-1054.

References

1. Continuum X-rays, Wittke JH (2009) http://www4.nau.edu/microanalysis/Microprobe-SEM/Signals.html. Accessed 8 July 2009
2. High Power X-ray Tube, Student Britannica (2009), http://student.britannica.com/eb/art/print?id=6614. Accessed 7 July 2009
3. Microfocus X-ray (2009) http://www.phoenix-xray.com/en/company/technology/principles_of_operation/principle_025.html. Accessed 7 July 2009
4. X-ray Phosphors, Phosphor Technology (2009) http://www.phosphor-technology.com/products/x-ray.htm. Accessed 21 July 2009
5. Scintillators as X-ray Detectors, NASA, USA (2009) http://imagine.gsfc.nasa.gov/docs/science/how_l2/xray_scintillators.html. Accessed 21 July 2009
6. Rare Earth elemenet, Wikipedia (2009) http://en.wikipedia.org/wiki/Rare_earth_element. Accessed 17 Dec 2009
7. National Institute of Standards and Technology (2009) http://physics.nist.gov/PhysRefData/XrayMassCoef/ tab3.html. Accessed 17 Dec 2009
8. Automated 3D X-ray Inspection System for On-line Foreign Body Detection in Food Products. Agro Industries Research (AIR), Commission of the European Communities, Ref. AIR2-CT93-1054
9. Palmer SC, Batchelor BG (1998) Model-based inspection of multi-packaged food products using a twin-beam X-ray system. In: Proceedings of the SPIE Conference on Machine Vision Systems for Inspection and Metrology VII, Boston, November 1998, vol 3521, pp 73–85, ISBN 0-8194-2982-1
10. X-ray Imaging Systems, NASA (2009) http://imagine.gsfc.nasa.gov/docs/science/how_l2/xtelescopes_systems.html. Accessed 17 Dec 2009
11. Refractive X-ray Lenses (2009) Technische Universität, Dresden, http://tudresden.de/die_tu_dresden/fakultaeten/fakultaet_mathematik_und_naturwissenschaften/fachrichtung_physik/isp/skm/research/xray_lenses, accessed 23rd July 2009
12. Graves M, Batchelor BG (2003) Machine vision for the inspection of natural products. Springer, London, ISBN 1-85233-525-4

Further Reading

Aolong Radiative Instrument Co. Ltd, Dandong City, Liaoning, PRC, URL: http://www.aolongndt.com/pcd.html. Accessed 29 July 2009

Cambier JL, Pasiak DC (1986) On-line X-ray inspection of packaged foods. Society of Manufacturing Engineers, Conference on Vision, Detroit 1986

Chan J (1988) Automated X-ray Inspection of foodstuff. MSc Thesis, University of Wales College of Cardiff, Cardiff, Wales

CXR Company, Inc., Warsaw, IN, USA, http://www.cxrcompany.com/. Accessed 29 July 2009

Davies ER (1997) Machine vision: theory, algorithms, practicalities, 2nd edn. Academic, London

Davies ER (2000) Image processing for the food industry. World Scientific, Singapore

Davies ER, Patel D (1995) Sensitive X-ray detection of contaminants in food products. The Institution of Electrical Engineers, Conference on Applications of Machine Vision, Savoy Place, London 2/1–2/6

Dykes GW (1985) Automated inspection of food jars for glass fragments. In: Proceedings of Society of Manufacturing Engineers, Conference on Vision, 6/21–6/30

EG&G (1991) X-ray meat inspection scans up to nine tons per hour. Prepared Foods 160, 87 (March 1991)

Focal Spot, Inc., San Diego, CA, USA, http://www.focalspot.com/. Accessed 29 July 2009

Graves M, Marshall A, Ahmed, S (1996) X-ray quality control in the Chicken Processing Industries. In: Proceedings of Campden and Chorleywood Conference on Applications of Image Analysis in the Food Industry, 16 July 1996

Graves M, Smith A, Batchelor B, Palmer S (1994) Design and analysis of X-ray vision systems for high-speed detection of foreign body contamination in food. In: Proceedings of SPIE Conference on Machine Vision Applications, Architectures, and Systems Integration III, vol 2347

Hunt, R, Samples of X-ray images (2009) http://www.rhunt.f9.co.uk/X-Ray_Gallery/X-Ray_Gallery_Page1.htm. Accessed 15 July 2009

Hunt R, Sample X-ray Images (2009) http://www.rhunt.f9.co.uk/X-Ray_Gallery/X-Ray_Gallery_Page1.htm. Accessed 29 July 2009

Keagy PM, Schatzki TF (1991) Effect of image resolution on insect detection in wheat radiographs. Cereal Chem 68(4):339–343

Kehoe A, Parker GA (1990) Image processing for industrial radiographic inspection: image enhancement. Br J NDT 2(4):183–190

Kehoe A, Parker GA, LeBlanc A (1989) Image processing for automatic foreign bodydetection in industrial radiographic images. In: IEE 3rd International Conference on Image Processing and Its Applications, Coventry (July)

Mettler-Toledo Ltd., Leicester, UK, http://uk.mt.com/gb/en/home.html. Accessed 29 July 2009

National Institute of Standards and Technology Physics Laboratory, Gaithersburg, MA, USA, X-ray Absorption Data (2009) http://physics.nist.gov/PhysRefData/XrayMassCoef/tab1.html. Accessed 29 July 2009

ORS, Inc., Whitesboro, NY, USA (2009) http:// www.ors-labs.com/RealTimeXrayInspection.html. Accessed 29 July 2009

Penman D, Olsson O, Beach D (1992) Automatic X-ray inspection of canned products for foreign material. SPIE 1832:342–347

Philips1 (1985) Foreign bodies in prepared foods detected by X-ray techniques. Philips Industrial X-ray Inspection Technical Bulletin 17.7350.38.0011.11

Palmer SC, Batchelor BG (1998) Model-based inspection of multi-packaged food products using a twin-beam X-ray system. In: Proceedings of SPIE Conference on Machine Vision Systems for Inspection and Metrology VII, Boston, MA, November 1998, vol 3521, pp 73–85, ISBN 0-8194-2982-1

Ramsay JD, Del Rossi G (1976) Method and apparatus for the detection of foreign material in food substances. United States Patent 3,995,164

Schatzki TF et al (1997) Foreign body detection in apples by means of X-ray imaging. SPIE 2907:176–185

Schatzki TF, Young R, Haff RP, Eye JG, Wright GR (1996) Visual detection of particulates in X-ray images of processed meat products. Opt Eng 35(8):2286–2291

Teradyne, Inc., North Reading, MA, USA (2009) http://www.teradyne.com/axi/. Accessed 29th July 2009

VJ Technologies, Inc., CT, USA (2009) http://www.vjt.com/. Accessed 29 July 2009

Wikipedia (2009) http://en.wikipedia.org/wiki/Automated_x-ray_inspection. Accessed 29 July 2009

X-Tek Systems Ltd., Tring, UK (2009) http://www.xtekxray.com/applications/library_blade.html. Accessed 29 July 2009

YXLON International GmbH, Hamburg, Germany (2009) http://www.yxlon.com/. Accessed 29 July 2009

Zwigelaar R et al (1996) X-ray simulations for imaging applications in agriculture and food industries. J Agr Eng Res 63(2):161–170

13 Illumination-Invariant Image Processing

John W. V. Miller · Malayappan Shridhar
The University of Michigan-Dearborn, Dearborn, MI, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_13,

Abstract: Illumination-invariant image processing is an extension of the classical technique of homomorphic filtering using a logarithmic point transformation. In this chapter, traditional approaches to illumination-invariant processing are briefly reviewed and then extended using newer image processing techniques. Relevant hardware considerations are also discussed including the number of bits per pixel required for digitization, minimizing the dynamic range of the data for image processing, and camera requirements. Three applications using illumination-invariant processing techniques are also provided.

13.1 Introduction

Scene illumination is one of the most important factors that determine success or failure for many imaging and machine vision applications. With appropriate lighting, irrelevant information can be eliminated, contrast of significant image features enhanced, and the sensor signal-to-noise ratio improved. Stringent requirements, however, may be placed on the lighting with regard to the need for very uniform and temporally constant levels of illumination unless suitable low-level processing is performed. Frequent lighting adjustments, tightly regulated power supplies or special feedback circuitry for controlling light level may be needed in these cases which can increase system cost and complexity significantly.

Even with ideal illumination, deficiencies in the optics and imaging hardware can introduce apparent variations in scene illumination that reduce the robustness and reliability of vision systems. Optical losses increase significantly from the centre of the field to the edge due to the $\cos^4$ law and vignetting [12]. Imagers often exhibit sensitivity variations (shading) as a function of spatial position. While higher-quality system components reduce these problems, the added cost can be excessive.

Real-time gray-scale image processing hardware which is generally available in newer vision systems can alleviate the effects of these imperfections economically. Traditionally, linear high-pass filtering prior to thresholding has often been used to reduce the effects of these variations since they are most significant at low spatial frequencies. Because image features of interest such as edges predominate at high spatial frequencies, the effects of illumination variations can be reduced prior to further processing [8, 21].

Morphological processing also has been used to correct for illumination variations [4, 22]. In some applications, a closing operation will generate an image that is a good estimate of background illumination by suppressing dark features at higher spatial frequencies. Calculating the difference between the background image and the original image generates a high-pass-filtered image.

High-pass filtering alone, however, does not completely eliminate these effects because most images are approximately a function of the product of the illumination and reflectance properties of a given scene [23]. An image feature in a dim area of an image will have poorer contrast than the same feature in a bright area as a result of this modulation-like property. A number of techniques have been used to remove the gray-level distortion introduced by this effect including shading correction and homomorphic filtering.

The term “illumination invariance” will be used to describe these techniques since they attempt to generate an image in which gray-scale errors associated with illumination, optics and the imager have been eliminated. Errors associated with illumination and optics are presumed to be purely multiplicative while the imager may require both an additive and a multiplicative correction factor [5]. Compensation for the multiplicative effects from all three sources can be provided simultaneously after imager offset errors have been removed.

Shading correction can directly compensate for these error sources if an “empty” scene is available, that is, a scene that is only a function of the illumination, optics and imager. For purposes of this chapter, such a scene is designated as the background. Scene elements of interest, such as objects or image features, comprise the foreground. Hence, if an image of the background can be acquired directly and stored, it can be divided into the foreground image data to compensate for existing nonuniformities [5, 12]. If imager offset errors are significant, a completely dark scene can be captured and subtracted from the imager data prior to multiplicative correction. Shading correction has been used in a variety of applications including thin-film disk inspection and the evaluation of lumber [3, 20].

Homomorphic filtering represents another approach to illumination-invariant processing [17, 23]. Here, gray-scale values of a given image are nonlinearly transformed, the resulting image is linearly filtered, and the inverse of the original nonlinear transformation is performed on the filtered image. Homomorphic filtering using a logarithmic transformation to linearize the multiplicative effect of illumination is a classical and well established image and signal processing technique [17]. Typical applications include enhancement of photographs containing scenes with wide illumination variations and edge detection [19, 21]. Homomorphic Wiener filtering has also been used for restoration [8].

In this chapter, classical techniques for illumination-invariant processing are reviewed and a number of enhancements are proposed. Homomorphic filtering, for example, has very advantageous properties. However, obtaining linear filters that remove or pass specific scene elements in a given image can be very difficult to implement. Nonlinear filtering operations such as morphological closings can often be generated with the desired characteristics. By logarithmically transforming a given image prior to morphological filtering, illumination-invariance may also be achieved.

Implementation issues are also examined here. A detailed discussion regarding the minimization of quantization errors with applications that require large dynamic range is presented. The use of an analog logarithmic amplifier prior to digitization is shown to provide significant benefits. Additional techniques to minimize A/D resolution requirements are discussed here also.

A discussion of various aspects of illumination-invariant processing has been divided into four sections in this chapter. In the next section, a basic approach to illumination-invariant processing based on background estimation with very general assumptions is developed along with a brief discussion of classical illumination-invariant processing approaches. A presentation of new techniques is given in ➋ Sect. 13.3 and hardware considerations for minimizing quantization errors are given in ➋ Sect. 13.4. In the last section of the chapter, three applications using illumination-invariant processing techniques are discussed.

13.2 Classical Illumination-Invariant Processing Techniques

The basic concept for illumination invariance follows from the image formation model in which light reflected from an object is approximately the product of the incident light and surface properties [12, 17]. If the background illumination is known, it can be used to normalize the light reflected from the object and minimize errors caused by illumination variations.

The same approach also can be applied to back-illuminated translucent objects or with an image created by passing X-rays through an object. In either case, the observed radiation is the product of the physical properties of the objects and the incident radiation. In this chapter, a simple one-dimensional model will be presented that is suitable for both reflectance and transmittance. If desired, this model can be generalized to two dimensions easily.

13.2.1 A Simple Image Formation Model

Consider the expression for the incident light $I[n]$ at the nth pixel of a sensor,

$$I[n] = I_0[n]s[n] \tag{13.1}$$

where $I_0[n]$ is the illumination for an empty scene and $s[n]$ is the value of the transmission or reflectance for the object in the scene at that point. The value of $s[n]$ can range from zero when no light is reflected or transmitted to one for total light reflectance or transmittance. Note that $s[n]$ is rarely smaller than 1% for the reflectance case [23].

This simple model does not take into consideration the complex manner in which objects with both specular and diffuse components reflect light. Other considerations such as the angle of incident light upon the observed surface, light polarisation, spectral properties and angle of the camera to the observed surface can also affect the detected light pattern markedly [12]. However, for purely transmissive cases and many machine vision applications employing reflected light, this model is appropriate.

If the assumption is made that the sensor linearly converts the illumination it receives into a voltage $v[n]$, then

$$v[n] = b[n]s[n] \tag{13.2}$$

where $b[n]$ is proportional to $I_0[n]$.

The maximum illumination signal possible at the nth pixel, $b[n]$, is a function of both the light source and the optics. Typically, this function is slowly varying such that

$$b[n] \approx b[n+k], 0 < k < d \tag{13.3}$$

in which d represents the number of sensor pixels with relatively little change in illumination. If the background illumination has significant high-frequency components, then d is correspondingly small.

To illustrate this concept further, consider one scan line of data from a simulated sensor plotted in ➋ *Fig. 13.1*. Here, small dark objects cause sharp drops in light level against the slowly varying background illumination. These objects have a transmission or reflectance value of 0.5 so that $s[n]$ is unity everywhere except in the presence of these objects. The background illumination level falls to zero at the edges of the scan. While this rarely happens in actual applications, similar but less severe illumination patterns are observed in typical applications due to optical and illumination properties [11, 12]. Note that no linear threshold exists that can differentiate between the background and all of the dark objects simultaneously.

13.2.2 High-Pass Filtering

As mentioned in the introduction, high-pass filtering is often used to reduce the effects of such nonuniformities. The high-pass filter will remove DC and low-frequency components while

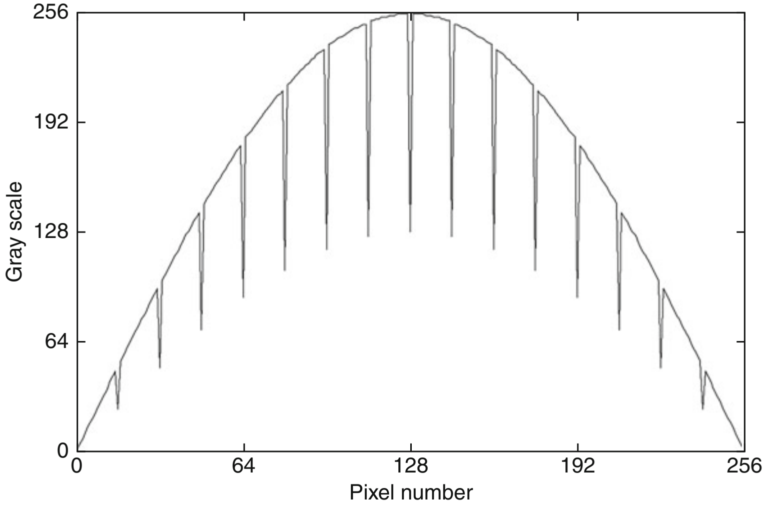


Fig. 13.1
Simulated scene containing small objects with *s*[*n*] = against a slowly varying background

retaining the high-frequency features of interest. In essence, the high-pass filter "subtracts" the low-frequency background from the image. Consider a high-pass filter which is implemented as the difference between a given signal and its filtered output obtained with a low-pass filter. Under the assumption that the low-pass filter will recover $b[n]$ (as shown in the upper trace of ➜ *Fig. 13.2*) from the expression $b[n]s[n]$ exactly for all n, the high-pass filter $y_{hp}[n]$ is defined as

$$y_{hp}[n] = b[n]s[n] - b[n] = b[n](s[n] - 1). \qquad (13.4)$$

Since $s[n]$ is unity everywhere except in the presence of the dark objects, $y_{hp}[n]$ is nonzero only for pixels covered by these objects. The output of the filter will be zero except in the presence of the dark objects where its output will be positive. Negative output values imply that the foreground image is brighter than the background illumination which is impossible for backlit objects. Specular reflections with front illumination can be brighter than the background estimate if it was obtained with a diffuse reflector. Note that the assumption for the existence of a filter $f(\cdot)$ such that

$$b[n] = f(b[n]s[n]) \qquad (13.5)$$

may prove to be very difficult to obtain in practice. If $f(\cdot)$ is to be a linear filter, serious implementation problems arise unless the power spectrums of the background and foreground are widely separated. Certain types of nonlinear filters, however, are likely to be much more suitable for estimating $b[n]$ accurately which are presented in ➜ Sect. 13.3.2.

Using the filter given in (➜ Eq. 13.4) to process the data of ➜ *Fig. 13.1* provides the results shown in the lower trace of ➜ *Fig. 13.2*. Although low threshold values exist that will detect the features of interest, the amplitudes of the dark objects vary significantly in proportion to the background light level. Low threshold values, however, will be very susceptible to noise and irrelevant features when $b[n]$ is large [11].

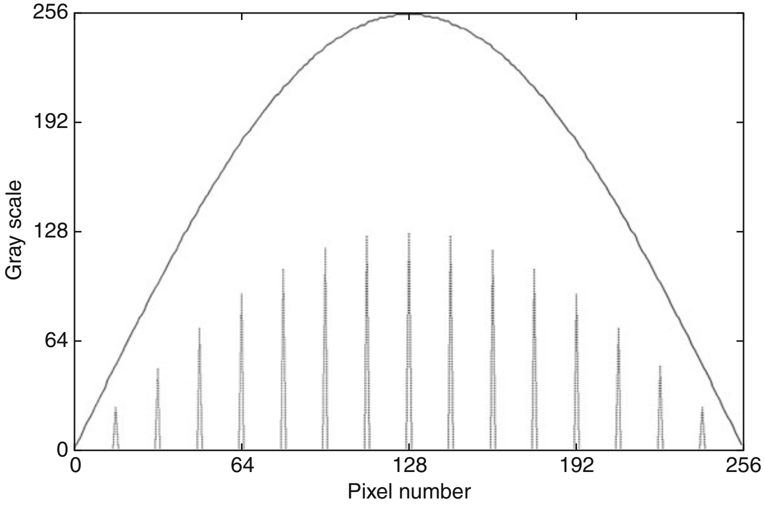


Fig. 13.2

By applying an ideal low-pass filter to the data of *Fig. 13.1*, the background illumination can be obtained as shown in the top trace. Subtracting the unfiltered data from it generates the lower trace

13.2.3 Illumination-Invariant Processing

Consider modifying (Eq. 13.4) so that the low-pass filtered signal is divided into the original signal.

$$y_{ii}[n] = \frac{b[n]s[n]}{b[n]} = s[n] \qquad (13.6)$$

Here, $y_{\mathrm{ii}}[n]$ is unaffected by illumination variations and thus is illumination invariant. This technique is often referred to as *shading correction* and may include correcting for sensor offset errors prior to the division operation [5, 20]. From this simple model, it would appear that any value of illumination is satisfactory. However, because of sensor characteristics, random noise, and quantization effects, the acceptable range of light variations will always be limited. Typically, illumination variations of up to 16 to 1 can be tolerated for many vision applications with 8-bit uniform quantization since at least 4 bits are still available in the darkest portion of the scene to represent $s[n]$ digitally. Further quantization considerations are discussed in Sect. 13.4.1.

13.2.4 Processing Using a Logarithmic Gray-Scale Transformation

Illumination-invariant processing can also be obtained by logarithmically transforming the gray-scale data from the sensor prior to high-pass filtering. The basic approach here is to replace the division operation in (Eq. 13.6) by the subtraction of two logarithmically transformed values. While the results are theoretically identical for the two methods, the logarithmic implementation has significant advantages with low-noise cameras which are considered in Sect. 13.4.2.

A linearly-digitized image is represented by a matrix of numbers which correspond to the light intensity of an image. Calculating the logarithm of each pixel results in a new image that is proportional to the density of a photograph of the same scene [23]. As will be seen, there are significant advantages to working with a density representation for achieving illumination-invariant processing.

To begin, consider a modification of (Eq. 13.4)

$$y_{\log hp}\,[n] = log(b[n][s[n]) - log(b[n]) = log(s[n]) \tag{13.7}$$

If desired, exponentiation can be performed so that results identical to equation (Eq. 13.5) are obtained. However, since the output of the filter is often only used for thresholding purposes, this additional operation is not required for many vision applications.

Applying (Eq. 13.7) to the simulation data plotted in *Fig. 13.1* gives the results plotted in *Fig. 13.3*. Here, the top trace is the logarithmically-transformed data from *Fig. 13.1* and the bottom trace is the result of background subtraction after logarithmic transformation. Note that all amplitudes are equal regardless of the illumination level.

If the low-pass filter used to estimate $\log(b[n])$ is linear with a spatial (not temporal) impulse response $h_1[n]$, the resulting high-pass filtering operation will be homomorphic after its output is exponentiated. This can be demonstrated by convolving $h_1[n]$ with the density image to obtain the low-pass-filtered output signal $y_1[n]$,

$$\begin{aligned} y_1[n] &= h_1[n] * \log(b[n]s[n]) \qquad (13.8)\\ &= h_1[n] * \log(b[n]) + h_1[n] * \log(s[n]) = \log(b[n]). \qquad (13.9)\end{aligned}$$

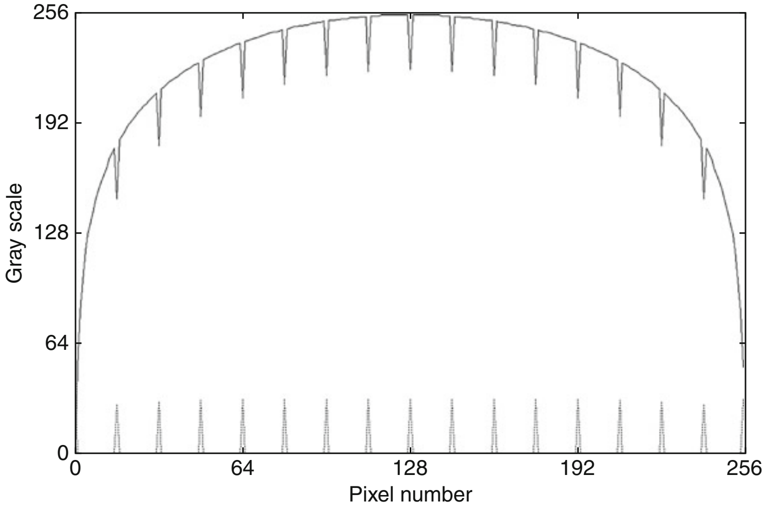


Fig. 13.3
The top trace has been generated by performing a logarithmic transformation of the data in *Fig. 13.1*. Low-pass filtering this trace and subtracting the top trace from it creates the bottom trace

A new impulse response can be defined as

$$h_2[n] = \delta[n] - h_1[n] \tag{13.10}$$

Convolving $h_2[n]$ with $\log(b[n]s[n])$ generates the high-pass filtered output $y_2[n]$,

$$\mathrm{y}_2[n] = h_2[n] * \log(b[n]s[n]) \tag{13.11}$$

$$= \log(b[n]s[n]) - h_1[n] * \log(b[n]s[n]) = \log(s[n]) \tag{13.12}$$

which is the desired information. As previously stated, $y_2[n]$ can be exponentiated to obtain a homomorphic filter.

13.2.5 Illumination-Invariant Filtering Using an FIR Filter

Values for $s[n]$ can be recovered very well from a scene if a good estimate of the background illumination $b[n]$ is available. The ability to generate this estimate is very application dependent and is essential if $s[n]$ is to be recovered accurately. For other applications, however, it is only necessary that the filter remove the DC component in a local region of the image. The gradient operator is one example of a commonly used *finite-impulse response* (FIR) filter in which $s[n]$ cannot be recovered although useful information is extracted from it.

Consider a high-pass FIR filter that removes the DC component over the extent of the filter and assume that it is small relative to the variation of the background illumination. If a density image is filtered by it, the resulting image will be illumination invariant. To show this, consider the basic FIR convolution operation

$$y[n] = \sum_{k=-K/2}^{K/2} C_k \log\left(x[n-k]\right) \tag{13.13}$$

in which C_k is the kth filter coefficient, $x[n]$ is the data to be filtered, and K is the kernel size. Since the output of the filter must be zero when the input signal does not vary over the extent of the filter, we will use the constant α to represent $x[n]$ over the extent of the filter.

$$y[n] = \sum_{k=-K/2}^{K/2} C_k \log\left(\alpha\right) = log(\alpha) \sum_{k=-K/2}^{K/2} C_k = 0 \tag{13.14}$$

From this, it can be inferred that

$$\sum_{k=-K/2}^{K/2} C_k = 0. \tag{13.15}$$

This condition can now be used to illustrate that the output of the filter is independent of the background illumination. Since $v[n] = s[n]b[n]$,

$$y[n] = \sum_{k=-K/2}^{K/2} C_k \log\left(s[n-k]b[n-k]\right) \tag{13.16}$$

$$= \sum_{k=-K/2}^{K/2} C_k \log\left(s[n-k]\right) + \sum_{k=-K/2}^{K/2} C_k \log\left(b[n-k]\right) \tag{13.17}$$

Using the requirement for a DC removing filter from (Eq. 13.14) and letting $\alpha = b[n]$ over the extent of the filter (recall that $b[n] \cong b[n \pm k]$ for k small),

$$y[n] = \sum_{k=-K/2}^{K/2} C_k \log\left(s[n-k]\right) \tag{13.18}$$

Clearly, $y[n]$ is independent of the background illumination. Note that in general $y[n] \neq \log(s[n])$.

13.2.6 Shading Correction

Shading correction is another technique that may be used if background illumination levels can be obtained directly. The inspection of transparent or translucent objects which are backlit is a typical class of applications that can be addressed with this approach. Often, an object will not be in the field of view continuously so that an image of the empty scene is available. In some applications, this will provide a direct measurement of the effective scene illumination. If the illumination is temporally invariant (Eq. 13.6) can be used to obtain illumination invariance.

Alternatively, background subtraction can be performed between the background density image and a density image containing both background and foreground information as in (Eq. 13.7). This approach is appropriate for both linear and matrix cameras. Implementation is usually easier with linear-camera systems since data from only one scan of the empty scene is required. For matrix cameras, a frame buffer can be used to store the background scene.

Note that no special assumptions about the spatial properties of the background relative to features of interest are required since spatial filtering is not used either to estimate the background illumination as in (Eq. 13.5) or to remove its effects as in (Eq. 13.12). If (Eq. 13.7) is used for shading correction, a time-domain filtering operation is being performed by the subtraction of two sample images acquired at different times.

However, sampling the background only once, and subtracting that image from more recent images results in a time-varying filter. As time progresses, the temporal frequency response of the filter changes because the effective sampling interval increases. In general, there is no loss of effectiveness provided that the background illumination level does not change significantly. It can be shown that adjusting for illumination changes is relatively straightforward if spatial and temporal illumination functions vary independently. A suitable approach for exploiting this is presented in the next section.

13.3 New Approaches for Illumination-Invariant Processing

While traditional approaches to illumination invariance are quite effective in many applications, significant enhancements can still be realized. In this section, extensions and improvements for realizing a more effective illumination invariance are described. To begin, a method for increasing the robustness of the direct background measurement approach is presented based on the assumption that spatial and temporal illumination variations are independent. The basic objective here is to correct for short-term lighting variations without having to capture a new background image. The basic assumption is made that illumination variations occur proportionally and simultaneously over the entire scene.

13.3.1 Compensating for Short-Term Temporal Illumination Variations

While both long- and short-term illumination variations can occur, long-term variations due to darkening and similar ageing effects can be ameliorated with relatively infrequent (daily or weekly) periodic sampling. However, short-term illumination variations associated with power-line fluctuations require a different solution. These variations can occur over periods on the order of 10 ms. which makes background resampling impractical in most applications. Incandescent sources exhibit a pronounced 100 or 120 Hz ripple when powered by 50 or 60 Hz. sinusoidal power sources. This ripple is caused by lamp filament cooling during zero crossings. With DC voltage sources, even modest voltage changes cause significant light output variations due to the great sensitivity of black-body radiators to temperature changes [15].

A variety of techniques can be used to deal with short-term deviations. Regulated DC sources can provide power for the illumination sources although this solution can be quite expensive if power requirements are substantial. In some applications, cameras can be scanned synchronously with the power line although AC power regulation may still be required. Television-type cameras are often synchronized with the 50 or 60 Hz power source using phase-locked loop circuitry to minimize light-source ripple. This approach, however, is unsuitable for high-speed matrix and line-scan camera applications.

More robust compensation can be provided using an extension of concepts previously discussed here. To begin, consider modifying (➋ Eq. 13.1) so that the effective illumination reaching the sensor is a function of both time and spatial position,

$$I[j,n] = I_0[j,n]s[j,n] \tag{13.19}$$

where $I[j, n]$ is the effective radiation for the jth time sample impinging on the nth camera pixel. Since illumination sources are usually driven from a common power source, the light output of each illumination source will vary temporally in exactly the same manner.

For this reason, $I_0[j, n]$ can be decomposed into the product

$$I_0[j,n] = I_t[j]I_s[n] \tag{13.20}$$

where $I_t[j]$ is the relative illumination level for the jth sample and $I_s[n]$ is the value of the illumination for the nth pixel at a specific time sample, such as $j = 0$. Under the same assumptions as (➋ Eq. 13.2), the voltage output of the sensor becomes

$$v[j,n] = I_t[j]b[n]s[j,n] \tag{13.21}$$

The density representation for (➋ Eq. 13.21) is

$$d[j,n] = \log(I_t[j]b[n]s[j,n]) = \log(I_t[j]) + \log(b[n]s[j,n]) \tag{13.22}$$

Assume that some reference region R exists in the image that is temporally invariant except for variations due to $I_t[j]$. This region can be a fixed painted white strip that is always present in the field of view of the camera. The light reflected into the camera from this region will vary directly as a function of the illumination. Let $a[j]$ be the average value of the N density-image pixels in this region for the jth sample,

$$a[j] = \frac{1}{N}\sum_{n\varepsilon R}\Big(log(I_t[j]) + log(b[n]s[j,n]\Big) \tag{13.23}$$

$$= \frac{1}{N}\Big(\sum_{n\varepsilon R} log(I_t[j]) + \sum_{n\varepsilon R} log(b[n]s[j,n]\Big) \tag{13.24}$$

Note that $\log I_t[j]$ is constant for all pixels in region R since it only varies as a function of the sample number. Similarly, a constant k_R can be defined as

$$k_R = \frac{1}{N}\sum_{n\varepsilon R} log(b[n]s[j,n]) \tag{13.25}$$

since region R is temporally invariant by definition except for $I_t[j]$. The average of region R becomes

$$a[j] = \log(I_t[j]) + k_R \tag{13.26}$$

This value can be used to generate a normalized version of the jth image, $d_n[j, n]$

$$d_n[j,n] = \log I_t[j] + \log(b[n]s[j,n]) - a[j] \tag{13.27}$$

$$= \log I_t[j] + \log(b[n]s[j,n]) - log(I_t[j]) - k_R \tag{13.28}$$

$$= \log(b[n]s[j,n]) - k_R \tag{13.29}$$

Notice that the amount of processing required is relatively small. The region R need only be large enough to minimize errors due to camera signal noise. The image normalization operation simply requires that a constant value be added to each pixel which could be done easily with a lookup table.

There are a number of ways in which this approach can be applied. Consider a simple case in which spatial uniformity of the illumination source is relatively good and image features have sufficiently high contrast. Assume simple thresholding is sufficient for segmentation or identifying features of interest with a temporally constant light source. Given some initial image, the average gray-scale value of the reference region can be found and used to normalize the image as given in (➲ Eq. 13.29). A threshold value can be found using a suitable technique and used on all succeeding images after they are normalized. Alternatively, the threshold T_0 could be found for the initial image and a new threshold value T_j calculated for the jth succeeding image as

$$T_j = T_0 - a[0] + a[j]. \tag{13.30}$$

Depending on the hardware available, this operation may be faster than normalizing each image and using a constant threshold.

Illumination-invariant template matching can be performed using this approach also. Assuming that the 0th image is the reference image, the difference image $\mathrm{d}[j, n]$ is calculated as

$$d[j,n] = d_n[j,n] - d_n[0,n] \tag{13.31}$$

$$= \log(b[n]s[j,n]) - k_R - (\log(b[n]s[0,n] - k_R) \tag{13.32}$$

$$= \log(s[j,n]) - \log(s[0,n]) \tag{13.33}$$

Background subtraction results if the 0th image is empty, that is $s[0, n] = 1$ for all n. Substituting this into (➲ Eq. 13.33) gives

$$\mathrm{d}[j,n] = \log(s[j,n]) \tag{13.34}$$

which is only a function of the foreground image.

13.3.2 Morphological Illumination-Invariant Processing

When the background cannot be measured directly, some means of background estimation by spatial filtering is very desirable. One effective technique for accomplishing this in many applications is gray-scale morphology. Given an image in which features of interest have limited spatial extent relative to a bright background, gray-scale morphology can provide an excellent estimate of the background.

Traditionally, the estimate of the background image obtained by morphological filtering is linearly subtracted from the original image prior to thresholding or some other segmentation operation. As discussed in Sect. 13.2, the result will not be illumination invariant since the modulating effects of the background illumination will still be present. However, using the background estimate with either (Eq. 13.6) or (Eq. 13.7) will provide illumination-invariant processing.

The basic approach advocated here is to use the closing operation on a density image. The closing operation consists of the sequential application of dilation and erosion using the same structuring element [4, 22]. The dilation operation fills in darker regions of an image as the brighter regions enlarge. If a given structuring element spans a dark region such as the narrow dips in *Fig. 13.1*, the dark region will disappear.

The inverse operation is then applied to remove biases caused by the dilation operation. Since the small dark objects have disappeared, this operation cannot restore them but instead will provide an estimate of the background illumination in the region of these objects based on adjacent pixels (*Fig. 13.4*).

One word of caution should be stated with respect to the ability of a given morphological operation to extract the background successfully. Unlike homomorphic filtering, in which

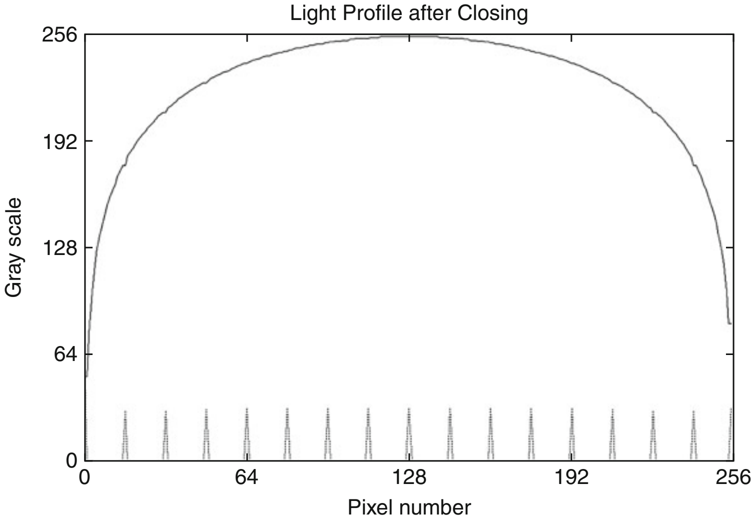


Fig. 13.4
The application of a gray-scale closing operation to the logarithmically transformed data of *Fig. 13.1* is shown in the top trace. The difference between the closed data and the logarithmically transformed data is given in the bottom trace

a linear filter is used, gray-scale morphological operations are highly nonlinear since they are based on min- and max-type functions. In addition, their output is generally biased so they do not preserve DC. Thus, no corresponding relationships exist comparable to the expressions in (➋ Eqs. 13.13–13.18) for morphological filters unless the background can be estimated with relatively little distortion.

13.3.3 Illumination-Invariant Adaptive Thresholding

Like morphological filters, adaptive thresholding can also be modified for illumination invariance. As in the previous approach, working with a density representation allows a straightforward means of decoupling the illumination and reflectance components in an image under a variety of conditions. Here again, the objective with illumination-invariant adaptive thresholding is to ensure that if a given feature in a bright area exceeds the threshold, then an equivalent feature in a dim area will also exceed it. With adaptive thresholding, the threshold at any given point in the image is based on some local average which is usually obtained by low-pass filtering the original image [2].

Using a density representation as the starting point for adaptive threshold processing, consider a low-pass filter that can provide a good estimate of the background $b[n]$. The threshold function $t[n]$ can be given as

$$t[n] = y_{loglp}[n] + T = \log(b[n]) + T. \tag{13.35}$$

Here, T is a sensitivity parameter. If T is large, a given gray-scale value will have to deviate substantially from the local average before the threshold is exceeded. The threshold decision function is *true* if

$$t[n] - \log(b[n]s[n]) < 0 \tag{13.36}$$

and substituting using equation (➋ Eq. 13.35)

$$\log(b[n]) + T - \log(s[n]) - \log(b[n]) < 0 \tag{13.37}$$

results in the following inequality:

$$T - \log(s[n]) < 0 \tag{13.38}$$

Whether or not the decision function is *true* depends only on T and $s[n]$ and is independent of the background illumination.

This operation produces the same bilevel image that is obtained using a constant threshold Ton an image generated by (➋ Eq. 13.7). However, an image obtained by applying (➋ Eq. 13.7) can provide more useful information for setup and adjustment. The high-pass filtered image can be viewed directly and the effect of threshold modification visualized immediately with standard image display hardware.

13.4 Hardware for Illumination-Invariant Processing

As can be seen from the previous discussion, there is a variety of ways to achieve illumination invariance. Some are significantly easier to implement in hardware than others or may have other advantages. Consider, for example, a hardware implementation of (➋ Eq. 13.7) in which the analog signal from the sensor is digitized by an analog-to-digital (A-D) converter and then filtered using a low-pass filter. Logarithmic conversion can be performed easily after digitization by means of a lookup table (LUT) using a memory chip.

13.4.1 Logarithmic Transformation Techniques

A potential disadvantage, however, exists with this approach. The digital logarithmic transform results in a large number of unrealizable values for low input-signal levels while multiple values at high input-signal levels are assigned to the same transformed value. Hence, the quantization is very coarse for low-amplitude signals and too fine at higher amplitudes with a corresponding decrease in dynamic range.

To illustrate this problem more fully, consider ➲ *Fig. 13.5*. The two ramp functions shown represent background illumination (higher ramp) and the foreground scene (lower ramp) with $s[n] = 0.05$. The other two traces represent density difference images. The nearly uniform horizontal trace is the result of performing an analog logarithmic transformation prior to digitizing the signal to obtain $y_a[n]$

$$y_a[n] = round(k_s \log(b[n])) - round(k_s \log(s[n]b[n])) \qquad (13.39)$$

The rounding function rounds its argument to the nearest integer to simulate the quantization error that results with A/D conversion. The scaling factor k_s for 8-bit quantization is calculated as

$$k_s = \frac{255}{\log(b_{\max})} \qquad (13.40)$$

where $b_{\max}$ is the maximum value of the background signal.

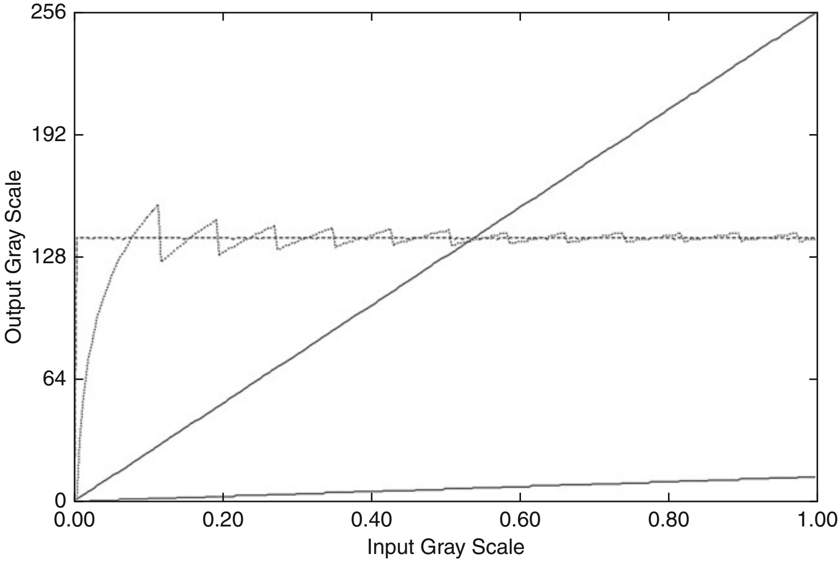


Fig. 13.5
Effects of quantization error when logarithmic transformation is performed before (nearly horizontal line) and after (approximately triangular waveform). The lower and higher ramps represent background and foreground data, respectively

The trace that oscillates in an approximately triangular fashion about the horizontal trace represents the results of digitization prior to logarithmic conversion. The output $y_d[n]$ is calculated as

$$y_d[n] = k_s \log(round(b[n]) - k_s \log(round(s[n]b[n])) \tag{13.41}$$

Since $s[n]$ is constant for all n, the output signal should be constant. Clearly, (Eq. 13.39) comes much closer to achieving this goal. Ignoring the meaningless result when the illumination is zero, a maximum error of two counts from the correct (rounded) value is observed. As indicated by the oscillating trace, however, digitization followed by logarithmic conversion causes substantially greater quantization errors, especially for low light-level conditions.

There are at least two ways to implement (Eq. 13.39) in hardware. Use of an analog logarithmic amplifier prior to digitization represents one approach. Here, the logarithmically-transformed signal is uniformly digitized over the range of acceptable voltages to the A/D converter. Sufficient resolution can be obtained for some applications with this approach using only 6-bit conversion [14].

The second approach is to use a nonlinear quantizer whose output values correspond to a logarithmic representation of the input. Theory covering nonlinear and optimal quantizers is well represented in the literature [10, 18]. However, a review of various manufacturers' data sheets indicates that nearly all A/D converters are implemented with uniform quantization including flash converters that are commonly used in real-time imaging applications.

Given the limitations with existing hardware, the best option is to use an analog logarithmic amplifier with a uniform quantizer. Analog adjustments can be performed to control the resolution per decade of the input signal and the range of input signals that can be digitized. The degree of amplification of the signal after logarithmic conversion determines the voltage range that will be digitized. Offsetting the transformed signal by a certain amount establishes the actual voltage endpoints that correspond to the minimum and maximum outputs of the quantizer.

This can be illustrated by considering the output voltage V_{out} of a logarithmic amplifier,

$$V_{out} = A \log(V_{in}) + V_{offset} \tag{13.42}$$

where A is the amplification, V_{in} is the input voltage and V_{offset} is a constant offset voltage. As the gain is increased, the resolution of the input signal is increased while the range of input voltages that can be converted is reduced. Generally, the offset voltage is adjusted so that the maximum camera signal will correspond to the maximum output value of the A/D converter.

Logarithmic conversion can be performed with standard op-amp circuits using a diode or transistor in the feedback circuit along with temperature compensation circuitry [16]. Alternatively, a monolithic logarithmic amplifier may also be used [9]. The dynamic range required for vision applications is usually not large since most video cameras have dynamic ranges less than 80 dB [6]. Therefore, except for bandwidth, performance requirements of logarithmic amplifier are fairly modest.

13.4.2 Camera Considerations

There are several factors that must be considered when designing a system with a logarithmic amplifier. Perhaps the most basic and obvious factor is the importance of minimizing DC offset errors. If very wide dynamic range is important in a given application, even relatively small DC offset errors can substantially reduce the accuracy of the results.

Video cameras often have substantial offset errors which can be measured by blocking all light from the sensor. The digitized results can be used to eliminate the offset errors by subtracting them on a pixel-by-pixel basis from the camera data [5, 12]. However, these techniques require the image to be digitized before correction is performed which is incompatible with analog logarithmic transformation. One approach that overcomes this limitation is to store the correction signal digitally but convert it back to analog format prior to subtraction and quantization.

Quantization errors can be minimized by increasing analog gain substantially when the offset errors are being acquired and rescaled for analog subtraction. By acquiring multiple images and averaging, the resulting dithering effect will also increase the effective resolution of the offset error image. Note that offset errors can be very temperature dependent with CCD cameras, especially if long integration times are used since correspondingly greater errors due to dark signal effects occur [6].

Background subtraction can be implemented in the same manner to increase the effective resolution of the system for detecting features with very low contrast [14]. The background image can be stored digitally in memory and converted back to analog format for subtraction and amplification prior to digitization. This provides even smaller quantization errors than digital subtraction after an analog logarithmic conversion. The background signal can be stored with substantially more resolution and converted back to analog form very cost effectively. The cost of a high-resolution D/A converter is relatively small compared to a flash converter with the same number of bits.

Note that this type of processing can only be justified with low-noise cameras. Typical television cameras have signal-to-noise ratios on the order of 46 dB which do not justify more than 8-bit representation [7]. With these cameras, lower amplitude signals are lost in the noise. The SNR for low-noise cameras, however, is on the order of 60–80 dB which justifies the additional analog processing [6].

One other caveat is worth noting here. Many cameras provide gamma correction to compensate for the nonlinear (power-law) brightness response of typical CRT displays with increasing grid-drive voltage [1]. The brightness B is given as

$$B = KE^{\gamma} \tag{13.43}$$

where K is a constant, E is the grid-drive voltage and γ is the value of the gamma parameter. Cameras with gamma correction provide the inverse operation

$$E' = \mathrm{E}^{1/\gamma}. \tag{13.44}$$

Values for γ range from 2.0 to 2.4 for typical CRT displays and flat-panel displays model this behaviour so cameras often use 1/2.2 = 0.45 as the correction factor.

Generally, gamma correction should be disabled if it is available on a given camera since a density representation can be obtained from a linear camera response as previously discussed. Also, the precision and stability of gamma correction is not necessarily very high. For typical video applications such as surveillance operations, where a person will observe the resulting image, the exact camera response is relatively unimportant [1]. Gamma correction is generally achieved with simple analog circuitry using PN junction properties to generate the desired function and consequentially is error prone.

In certain non-critical vision applications, however, it may be appropriate to use the gamma-corrected camera signal in place of a logarithmic transformation. Its general shape is similar to the logarithmic transformation as illustrated in ➋ *Fig. 13.6*. Performing background

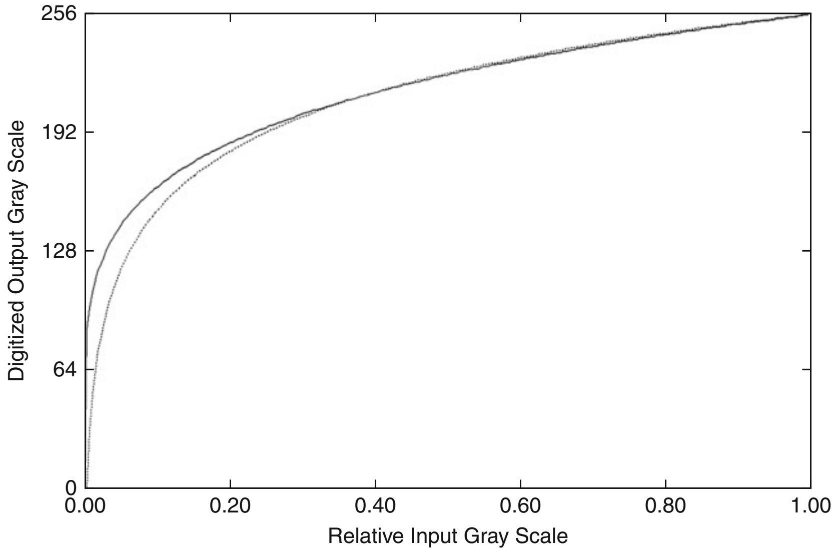


Fig. 13.6
Scaled gamma correction with an exponent of .45 (*top curve*) and logarithmic transformation function (*bottom curve*) are very similar except for low gray levels

subtraction with the original data of ➲ *Fig. 13.1* after gamma correction provides the results in ➲ *Fig. 13.7*. While there is some variation as a function of background illumination, the results are significantly better than the linear differencing results illustrated in ➲ *Fig. 13.2*.

It is also possible to remove the gamma correction digitally [5]. Although traditionally the objective has been to obtain a linear response to light intensity, a logarithmic response can be obtained in the same manner. The similarity of the curves plotted in ➲ *Fig. 13.6* indicate that relatively minor changes in signal levels are required. Quantization errors with this approach are almost as small as with a direct logarithmic transformation because of the similarity of the two curves. However, the problems noted earlier with regard to accuracy and stability may limit the usefulness with many cameras. Individual camera calibration may be performed if stability is not a problem [5].

13.5 Choice of Illumination-Invariant Techniques

A variety of approaches for achieving illumination invariance have been presented in this chapter. In the following section, considerations and tradeoffs for making appropriate choices are discussed for three applications. The first two applications deal with glass inspection.

13.5.1 Plate Glass Inspection

Glass is used in a variety of applications from windows to packaging and quality is often very important. The quality requirements for plate glass in automotive and architectural applications are high because of safety considerations which justify automatic flaw-detection

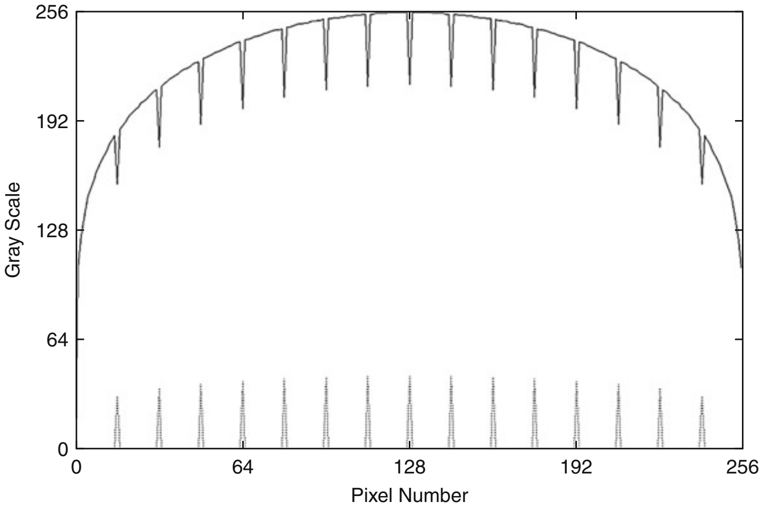


Fig. 13.7
If a camera provides gamma correction, the results are similar to those obtained with a logarithmic transformation shown in ➔ *Fig. 13.3*

techniques. Defects such as scratches, embedded particles, and bubbles need to be detected so that defective glass can be rejected. Glass bottles used for carbonated beverages represents another area where safety is a primary concern. Bottles with defects may explode when handled by the consumer with severe liability consequences.

The generally uniform nature of glass makes it a relatively easy material to inspect since only significant gray-scale changes from the normal background need to be evaluated for defect detection. If defects have sufficient contrast and lighting is reasonably uniform, simple thresholding will provide adequate results and no illumination compensation is required. If the illumination source exhibits temporal variations, the use of a reference region in conjunction with (➔ Eq. 13.29) is appropriate.

Many defects do not have high contrast relative to spatial illumination variations so more sophisticated processing is required. Under these conditions, providing illumination compensation becomes highly advantageous. As stated earlier, the primary objective is to estimate the background illumination by some means or measure it directly.

Direct measurement of the background is very compatible with plate glass because of the highly uniform nature of the material with no defects. Since transmission through a normal glass panel is spatially invariant, the background illumination may be measured directly with a defect-free piece of glass in place.

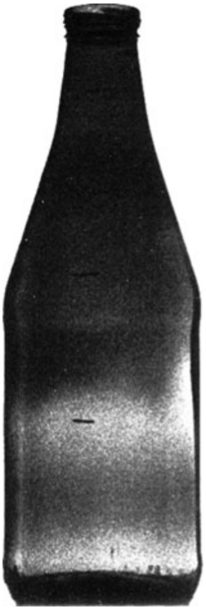
One problem with this approach is that the presence of dust and similar contaminants will introduce inaccuracies in the background measurement. A number of different good panels may be used and combined to create an improved estimate of the background illumination. Note that averaging the measurements together on a pixel-by-pixel basis may not provide the best results. Selecting the maximum gray-scale value at each pixel location will eliminate the attenuating effects of any contaminants present.

Alternatively, some kind of low-pass filtering may be used to minimize the effects of contaminants, especially if they are likely to be small. Since processing the background image will only need to be performed infrequently, more sophisticated off-line algorithms may be implemented for better background estimation. Note that an estimate based on multiple sampling may be spatially low-pass filtered if desired. Low-pass filtering of the estimate should not be performed however, if the sensor exhibits significant pixel-to-pixel sensitivity variations. Once a good estimate of the background has been obtained, it may be used in conjunction with (➲ Eqs. 13.30 and ➲ 13.34) to compensate for short-term illumination variations that may occur.

13.5.2 Amber Bottle Inspection

While many glass-inspection applications should be compatible with the direct background measurement approach, certain circumstances may require a different technique. Consider the amber bottle illustrated in ➲ *Fig. 13.8*. Large variations in light transmission are very evident due to normal sidewall glass-thickness differences that occur during manufacturing. Bottle inspection is typically performed by rotating the bottle in front of a linear camera to provide complete circumferential inspection [13]. As with plate glass, embedded defects are a serious concern and can be detected because they cause light levels to deviate from the normal background significantly. However, directly measuring the background is of no value here because glass thickness varies substantially in a random manner. Therefore, some type of real-time filtering is required to distinguish between normal background variations and defects.

Since the background variations occur at much lower spatial frequencies than defects do, linear high-pass filtering is very useful and effective. While a variety of filters may be useful



◘ Fig. 13.8

Significant variations in light transmission through amber bottles occur due to nonuniform glass thickness

here, consider the simple illumination-invariant filtering operation based on pixel differencing of a density image:

$$y_{diff}[n] = \log(b[n]s[n]) - \log(b[n-1]s[n-1]) \tag{13.45}$$

$$= \log(b[n]s[n]/(b[n-1]s[n-1])) \tag{13.46}$$

Since $b[n] \approx b[n-1]$ if the background is slowly varying,

$$y_{diff}[n] \approx \log(s[n]/s[n-1]) \tag{13.47}$$

which is illumination invariant since the filter output is not a function of $b[n]$. Because the filtering operation is linear, exponentiation of the results will provide homomorphic filtering. This filter is also DC removing as defined in Sect. 13.2. It may also be noted that $s[n]$ is not recovered with this filter.

Glass inspection and the general class of uniform field objects represent one type of application that can benefit significantly from illumination-invariant processing. Another application that requires similar processing is automatic furnace batch charge monitoring. Here, it is desired to measure the presence of batch material is measured with a vision system.

13.5.3 Batch Charge Monitoring

The major difficulty with automatic batch charge monitoring is the very large and temporally varying illumination gradients that are created by the flaming gas jets that are used to heat the furnace. Processing is needed to transform the original image so that a single threshold can detect unmelted batch material on the surface of the molten glass as illustrated in *Fig. 13.9a*. Recently injected unmelted batch material is visible at the far end of the furnace where it is not obscured by the large flame. This material gradually melts and mixes with the molten glass as it moves towards the bubblers. The bubblers are the very dark circular areas which mix the remaining unmelted batch material with the melted glass. No unmelted batch material should be present past the bubblers.

The objective here is to identify the areas of the furnace which are covered by batch material and determine whether or not it is covering the bubblers. Using an octagonal structuring element a closing operation was performed on the original image to estimate the background. The difference between the background estimate and the original image is shown in *Fig. 13.9c*. While it is clear that the low-frequency illumination variations have been removed, significant contrast variations remain since there is considerable variation in the brightness of the batch material as a function of illumination.

Using a logarithmic transformation, a density representation of *Fig. 13.9a* was obtained as shown in *Fig. 13.9b*. The density image was filtered by applying a 3 × 3 averaging filter 18 times. The difference between this image and the density image is given in *Fig. 13.9e*. A thresholded version of this image is given in *Fig. 13.9f*. While the effects of the illumination variations have been eliminated, the results are actually inferior to the traditional morphological filter because the background estimate is much poorer.

Morphological filtering of the density image provides the best results as shown in *Fig. 13.9g* and in its thresholded version given in *Fig. 13.9h*. The same morphological filtering operation used to generate *Fig. 13.9c* was performed here. A comparison of the three thresholded images given *Figs. 13.9d*, *13.9f* and *13.9h* shows that morphological filtering of the density image provides the best estimate for the presence of batch material.

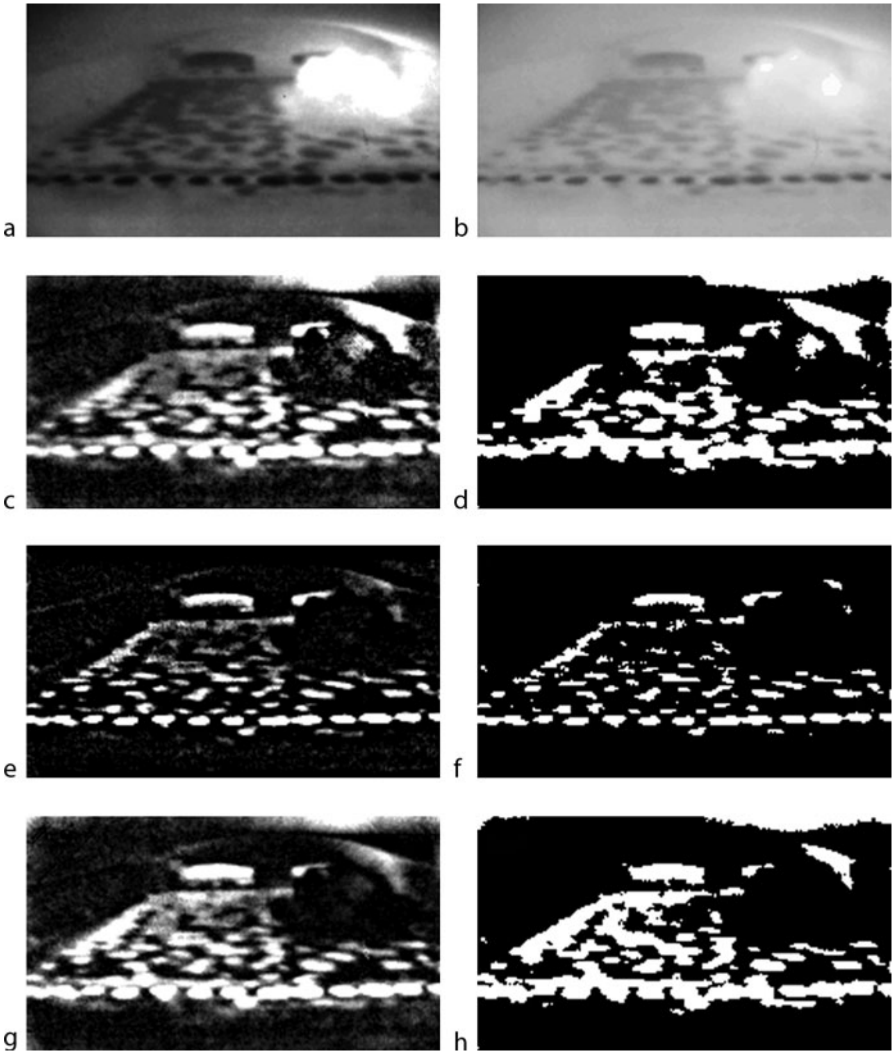


Fig. 13.9
Scene from a glass furnace is shown in (a). The same scene after logarithmic conversion is given in (b). Results obtained by closing (a) and subtracting are given in (c). Thresholding (c) provides the results shown in (d). Linearly filtering (b) results in images given in (e) and (f). Results obtained by closing (b) and subtracting are given in (g) and (h)

13.6 Conclusion

Illumination-invariant processing can be achieved in a variety of ways. The basic approach described here assumed that some method was available for measuring the background illumination. In some applications, background can be measured directly while in others low-pass filtering provides a good estimate. If direct but infrequent background measurements are performed, degradations associated with global short-term temporal variations can be removed by making adjustments based on spatially invariant regions of a scene.

The use of a logarithmic transformation to convert images from an intensity representation to a density representation is very important for illumination-invariant processing. The multiplicative effects of illumination becomes additive for density images so that the effects of nonuniform illumination can be removed by subtraction.

There are a variety of ways to obtain a density representation of an image. The most straightforward approach is to digitize the image and use a look-up table to perform the logarithmic transformation. Using an analog logarithmic amplifier prior to digitization results in a uniform distribution of quantization errors in density images and provides a much larger dynamic range for a given number of bits. In noncritical applications, digitization after gamma correction may provide suitably accurate results.

References

1. Benson KB (1986) Television engineering handbook. McGraw-Hill, New York
2. Cielo P (1988) Optical techniques for industrial inspection. Academic, San Diego, pp 234–235
3. Conners RW, McMillin CW, Lin K, Vasquez-Espinosa RE (1983) Identifying and locating surface defects in wood: part of an automated lumber processing system. IEEE Trans Pattern Anal Mach Intell PAMI5(6):573–583
4. Daut DG, Zhao D (1989) Mathematical morphology and its application in machine vision. Vis Commun Image Process IV SPIE 1199:181–191
5. Dinstein I, Merkle F, Lam TD, Wong KY (1984) Imaging system response linearization and shading correction. Opt Eng 23(6):788–793
6. Fairchild (1989) Fairchild Weston CCD imaging databook. Fairchild Weston, Milpitas. http://www.mirametrics.com/tech_note_tempvar.htm
7. (1990) Fordham Radio Catalog, vol 16, No. 5, Hauppauge. http://www.cs.columbia.edu/~hgs/rtp/cameras.html
8. Fries RW, Modestino JW (1979) Image enhancement by stochastic homomorphic filtering. Trans Acoust Speech Signal Process ASSP-27(6): 625–637
9. Gilbert B, Clarke B (1989) Monolithic DC-to-120 MHz log-amp is stable and accurate. Analog Dialog 23(3): 189–192
10. Hall EL (1979) Computer image processing and recognition. Academic, New York
11. Hsing TR (1984) Techniques of adaptive threshold setting for document scanning applications. Opt Eng 23(3):288–293
12. Kleinemeier B (1985) Real-time restoration of image sensor photoresponse nonuniformity. Opt Eng 24(1):160–170
13. Miller JWV (1984) Real-time system for the automatic inspection of uniform field objects. Proceedings of the third annual applied machine vision conference, Schaumburg, IL
14. Miller JWV, Miller PS (1990) Image analysis system employing filter look-up tables. US Patent No. 4,941,191
15. Murdoch JB (1985) Illumination engineering–from Edison's lamp to the laser. Macmillan, New York
16. National Semiconductor Corporation (1982) National Semiconductor Corporation linear databook. Santa Clara
17. Oppenheim AV, Schafer RW, Stockham TG (1968) Nonlinear filtering of multiplied and convolved signals. Proc IEEE 56(8):1264–1291
18. Pratt WK (1978) Digital image processing. Wiley, New York
19. Ray R, Wilder J (1984) Visual and tactile sensing system for robotic acquisition of jumbled parts. Opt Eng 23(5):523–530
20. Sanz JLC, Petkovic D (1988) Machine vision algorithms for automated inspection of thin-film disk heads. IEEE Trans Pattern Anal Mach Intell 10(6):830–848
21. Schreiber WF (1978) Image processing for quality improvement. Proc IEEE 66(12):1640–1651
22. Sternberg S (1983) Biomedical image processing. Computer 16(1):22–34
23. Stockham TG (1972) Image processing in the context of a visual model. Proc IEEE 60(7):828–842

14 Basic Machine Vision Techniques

Bruce G. Batchelor[1] · *Paul F. Whelan*[2]
[1]Cardiff University, Cardiff, Wales, UK
[2]Dublin City University, Glasnevin, Dublin, Ireland

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_14,

Abstract: A discussion of image processing must begin by considering how images are represented. The most commonly used format is that of a 2-dimensional array for monochrome images, while colour images are usually represented by three such arrays. The array format is generated by most cameras. While other representations present advantages in certain, rather restricted situations, they must usually be derived from an image array. Grey-scale image processing functions can be arranged into several major categories. The simplest, called *monadic* operators, process one image at a time, by performing a fixed intensity mapping from each of its pixels to the pixel at the corresponding position in the output image. *Dyadic* functions operate on two images; the intensities of pixels at corresponding addresses are combined to calculate a value for the pixel at the same address in the output image. *Local operators* process a single image. In order to calculate the intensity at a point (i,j) in the output image, several pixels surrounding point (i,j) in the input image are combined. For example, the average, maximum, or minimum intensity within the 3^*3 pixel neighbourhood surrounding point (i,j) might be computed. Such a process is repeated for all (i,j). Numerous filtering functions can be invented by extending the concept of "neighbourhood". The function that is applied to the pixel values within a given neighbourhood can be linear or non-linear. The latter gives rise to a useful family, given the name *morphology*. Another group is based on the application of a monadic function that is derived in some way from the intensity histogram, or other statistical features, of the input image. Another, varied family is based on Fourier, and other *integral transforms* of the 2-dimensional intensity function. *Image warping* and *geometric transforms* form another family of grey-scale image processing function. Some grey-scale operators have obvious counterparts for binary images. However, the latter also require functions such as object counting, feature detection, measurement of position, orientation, size and shape. Binary image processing operators do not fall neatly into well-defined categories but can often be described functionally by drawing on everyday concepts such as that of skeleton, rubber band, compass, ruler and our intuitive understanding of shape fitting. The processing of colour images is not covered here as it is discussed in detail elsewhere in this book (➲ Chap. 16). For the moment, let it suffice to say that grey-scale operators can be applied to the individual RGB, HSI or CMYK components of a colour image.

The purpose of this chapter is to outline some of the basic techniques used in the development of industrial machine vision systems. These are discussed in sufficient detail to understand the key ideas outlined elsewhere in this book. In the following discussion we will frequently indicate the equivalent QT operators for the vision techniques described. (QT is an interactive image processing system and is described in ➲ Chap. 21. ➲ Chap. 41 describes the function of each QT operator.) QT commands appear in square brackets. In certain cases, sequences of QT commands are needed to perform an operation and these are similarly listed.

14.1 Representations of Images

We will first consider the representation of *monochrome* (grey-scale) images. Let i and j denote two integers where $1 \leq i \leq m$ and $1 \leq j \leq n$. In addition, let f(i,j) denote an integer function such that $0 \leq f(i,j) \leq W$. (W denotes the white level in a grey-scale image.) The array **F** will be called a *digital image*:

$$F = \begin{vmatrix} f(1,1) & f(1,2) & \ldots & f(1,n) \\ f(2,1) & f(2,2) & \ldots & f(2,n) \\ f(3,1) & f(3,2) & \ldots & f(3,n) \\ \ldots & \ldots & \ldots & \ldots \\ f(m,1) & f(m,2) & \ldots & f(m,n) \end{vmatrix}$$

An address (i,j) defines a position in **F**, called a *pixel*, *pel* or *picture element*. The elements of **F** denote the intensities within a number of small rectangular regions within a real (i.e., optical) image (see ➋ *Fig. 14.1*). Strictly speaking, f(i,j) measures the intensity at a single point but if the corresponding rectangular region is small enough, the approximation will be accurate enough for most purposes. The array **F** contains a total of *mn* elements and this product is called the *spatial resolution* of **F** (➋ *Fig. 14.2*). We may *arbitrarily* assign intensities according to the following scheme:

$f(i,j) = 0$	Black
$0 \leq f(i,j) \leq 0.33W$	Dark grey
$0.33W < f(i,j) \leq 0.67W$	Mid-grey
$0.67W < f(i,j) < W$	Light grey
$f(i,j) = W$	White

Let us consider how much data is required to represent a grey-scale image in this form. Each pixel requires the storage of $\log_2(1+W)$ bits. This assumes that (1+W) is an integer power of two. If it is not, then $\log_2(1+W)$ must be rounded up to the next integer. This can be represented using the ceiling function $\lceil \cdots \rceil$. Thus, a grey-scale image requires the storage of $\lceil \log_2(1+W) \rceil$ bits. Since there are *mn* pixels, the total data storage for the entire digital image **F** is equal to $mn \lceil \log_2(1+W) \rceil$ bits. If $m = n \geq 128$, and $W \geq 64$, we can obtain a good image of a human face (➋ *Fig. 14.3*). Many of the industrial image processing systems in use nowadays manipulate images in which m and n are at least 512 and W is 255 (➋ *Fig. 14.4*). This leads to a storage requirement of 256 KB/image. A binary image is one in which only two

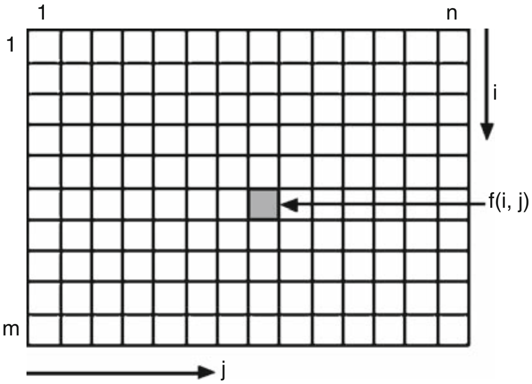


Fig. 14.1

A digital image consisting of an array of *mn* pixels. The pixel in the ith row and jth column (address *[i,j]*) has a intensity equal to *f(i,j)*

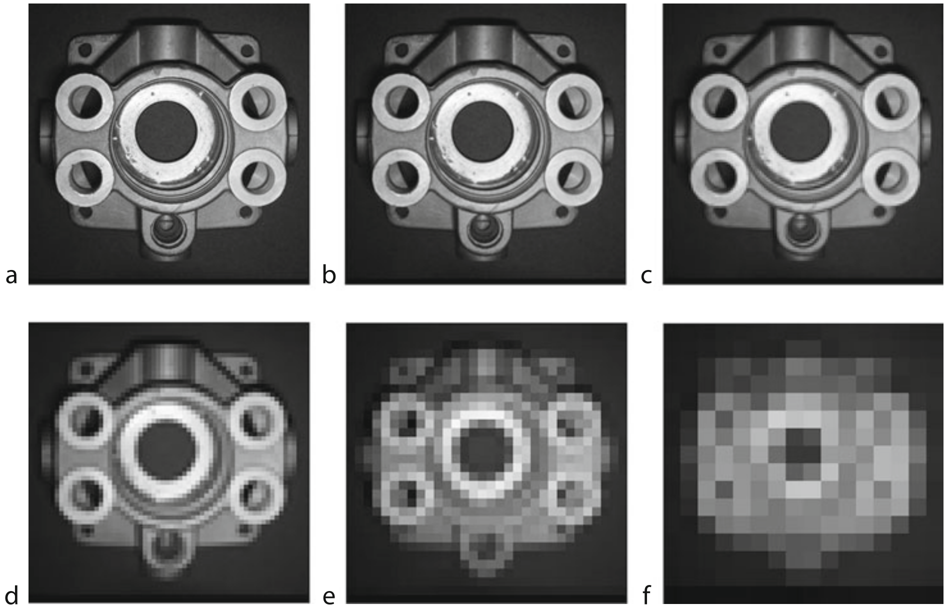


Fig. 14.2
Effects of varying resolution. In each case there are 256 possible grey levels. (a) 512 × 512 pixels, (b) 256 × 256 pixels, (c) 128 × 128 pixels, (d) 64 × 64 pixels, (e) 32 × 32 pixels, and (f) 16 × 16 pixels

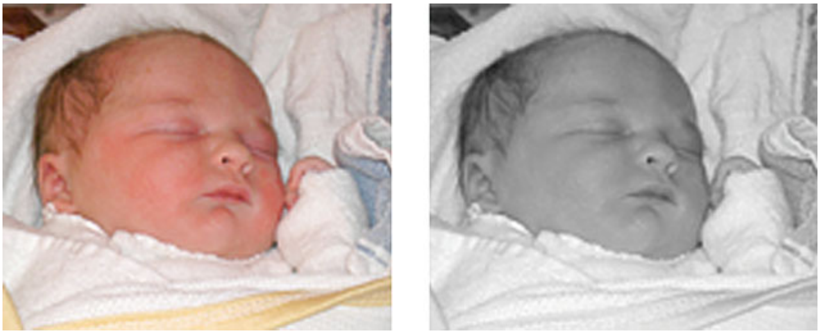


Fig. 14.3
A modest resolution, in this case 128 × 107 pixels, is sufficient to provide a reasonably good representation of human face. The colour image consists of 3 monochrome components, each 8 bits deep

intensity levels, black (0) and white (1), are permitted. This requires the storage of *mn* bits/image.

An impression of colour can be conveyed to the eye by superimposing *four* separate imprints. Cyan, magenta, yellow, and black inks are often used in printing (➔ *Fig. 14.5*). Ciné film operates in a similar way, except that when different colours of light, rather than ink, are added together, *three* components (red, green, and blue) suffice. Television operates

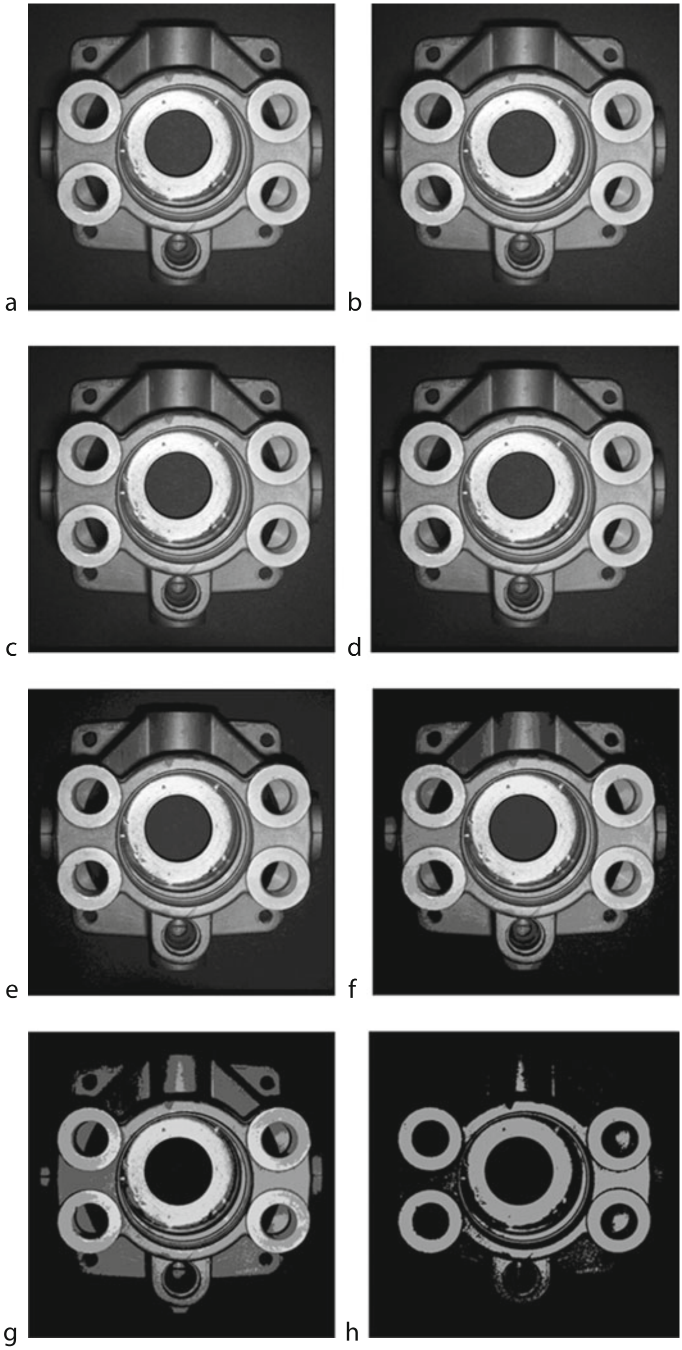


Fig. 14.4

Effects of varying the depth (number of bits needed to represent each intensity value). In each case there are 512 × 512 pixels. (a) 9 bits, (b) 7 bits, (c) 6 bits, (d) 5 bits, (e) 4 bits, (f) 3 bits, (g) 2 bits, and (h) 1 bit

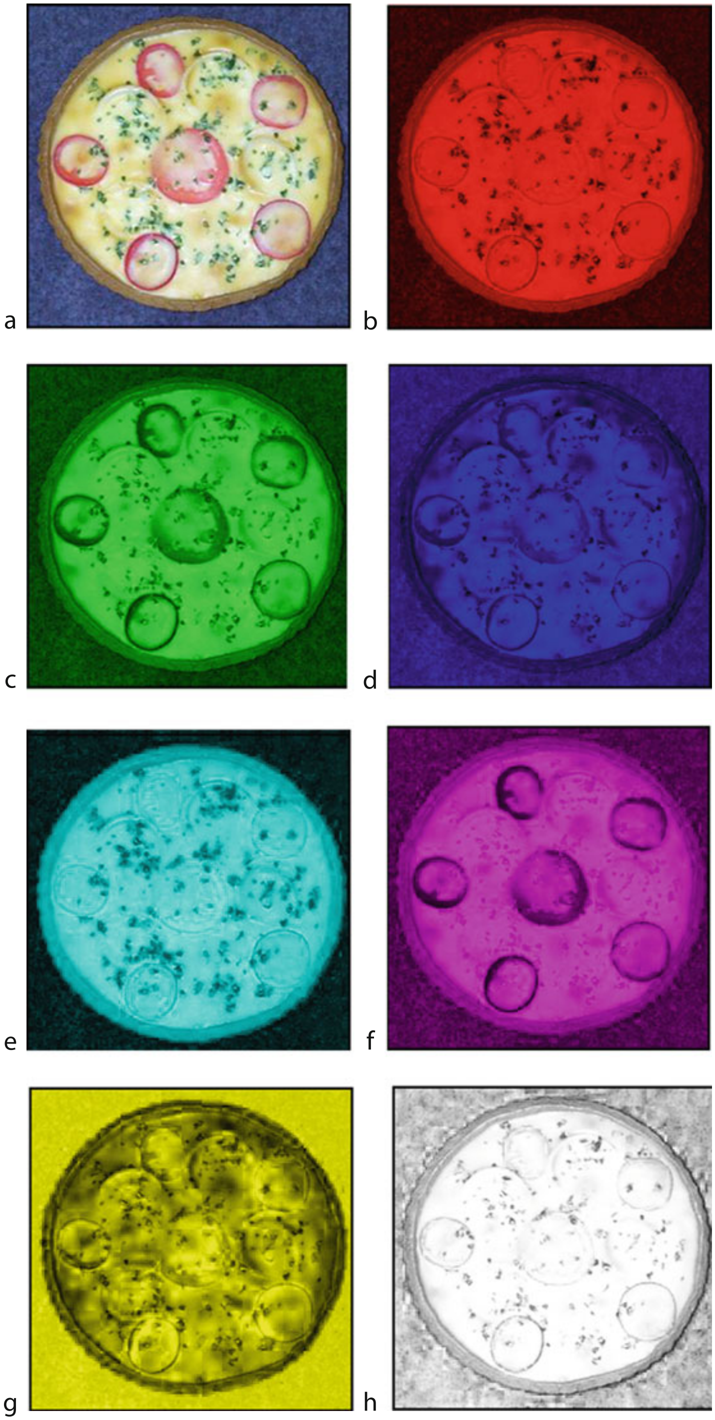


Fig. 14.5 (Continued)

in a similar way to film: the signal from a colour television camera may be represented using three components: **R** = *{r(i,j)}*; **G** = *{g(i,j)}*; **B** = *{b(i,j)}*, where **R, G** and **B** are defined in a similar way to **F**. The vector *{r(i,j), g(i,j), b(i,j)}* defines the intensity and colour at the point *(i,j)* in the colour image. (Colour image analysis is discussed in more detail in ➔ Chap. 16.) Multi-spectral images can also be represented using several monochrome images. The total amount of data required to code a colour image with *r* components is equal to $mnr\lceil \log_2(1+W) \rceil$ bits, where *W* is the maximum signal level on each of the channels.

In order to explain how moving scenes may be represented in digital form, ciné film and television will be referred to. A ciné film is, in effect, a time-sampled representation of the original moving scene. Each frame in the film is a standard monochrome or colour image, and can be coded as such. Thus, a monochrome ciné film may be represented digitally as a sequence of two-dimensional arrays [$\mathbf{F}_1$, $\mathbf{F}_2$, $\mathbf{F}_3$, $\mathbf{F}_4$,...]. Each $\mathbf{F}_i$ is an *mn* array of integers, as we defined above when discussing the coding of grey-scale images. If the film is in colour, then each of the $\mathbf{F}_i$ has three components. In the general case, when we have a spatial resolution of *mn* pixels, each spectral channel permits *(1+W)* intensity levels, there are *r* spectral channels, and *p* is the total number of "stills" in the image sequence, we require $mnpr\lceil \log_2(1+W) \rceil$ bits/image sequence.

We have considered only those image representations which are relevant to the understanding of simple image processing and analysis functions. Many alternative methods of coding images are possible but these are not relevant to this discussion.

14.2 Elementary Image Processing Functions

The following notation will be used throughout this section, in which we will concentrate upon grey-scale images, unless otherwise stated.

- *i* and *j* are row and column address variables and lie within the ranges $1 \leq i \leq m$ and $1 \leq j \leq n$ (➔ *Fig. 14.1*).
- **A** = {a(i,j)}, **B** = {b(i,j)} and **C** = {c(i,j)}.
- *W* denotes the white level.
- g(X) is a function of a single independent variable X.
- h(X,Y) is a function of two independent variables, X and Y.
- The assignment operator "←" will be used to define an operation that is performed upon one data element. The assignment operator "⇐" will be used to indicate that an operation is to be performed upon *all* pixels within an image.
- k, k1, k2, and k3 are constants.
- N(i,j) is that set of pixels arranged around the pixel (i,j) in the following way:

Fig. 14.5
Representing a colour image. (a) Original image, plastic replica of a quiche. (b) R component. (c) G component. (d) B component. Together the RGB components form the usual representation of colour used in video, film and Machine Vision. Notice that these are monochrome images. (e) Cyan (C) component. (f) Magenta (M) component. (g) Yellow (Y) component. (h) Black (K) component. Together the CMYK components form the representation that is often used in printing

(i–1, j–1)	(i–1, j)	(i–1, j+1)
(i, j–1)	(i, j)	(i, j+1)
(i+1, j–1)	(i+1, j)	(i+1, j+1)

Notice that N(i,j) forms a 3×3 set of pixels and is referred to as the *3×3 neighbourhood* of (i,j). In order to simplify some of the definitions, we will refer to the intensities of these pixels using the following notation:

A	B	C
D	E	F
G	H	I

Ambiguities over the dual use of A, B and C should not be troublesome, because the context will make it clear which meaning is intended. The points {A, B, C, D, F, G, H, I} are called the *8-neighbours* of E and are also said to be *8-connected* to E. The points {B, D, F, H)} are called the *4-neighbours* of E and are said to be *4-connected* to E.

14.2.1 Monadic, Pixel-by-Pixel Operators

These operators have a characteristic equation of the form

$$c(i,j) \Leftarrow g(a(i,j)) \text{ or } E \Leftarrow g(E)$$

Such an operation is performed for all (i,j) in the range [1,m],[1,n] (See ➜ *Fig. 14.6*). Several examples will now be described.

Intensity Shift (➜ *Fig. 14.7*) *[acn]*

$$c(i,j) \Leftarrow \left[\begin{array}{cc} 0 & a(i,j)+k<0 \\ a(i,j)+k & 0 \le a(i,j)+k \le W \\ W & W < a(i,\ j)+k \end{array} \right.$$

k is a constant, set by the system user. Notice that this definition was carefully designed to maintain c(i,j) within the same range as the input, viz [0,W]. This is an example of a process referred to as *intensity normalisation.* Normalisation is important because it permits iterative processing by this and other operators in a machine having a limited precision for arithmetic (e.g., 8-bits). Normalisation will be used frequently throughout this chapter.

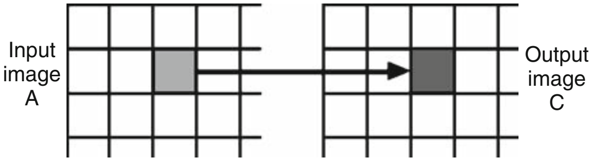


Fig. 14.6

Monadic pixel-by-pixel operator. The (i,j)th pixel in the input image has intensity a(i,j). This value is used to calculate c(i,j), the intensity of the corresponding pixel in the output image

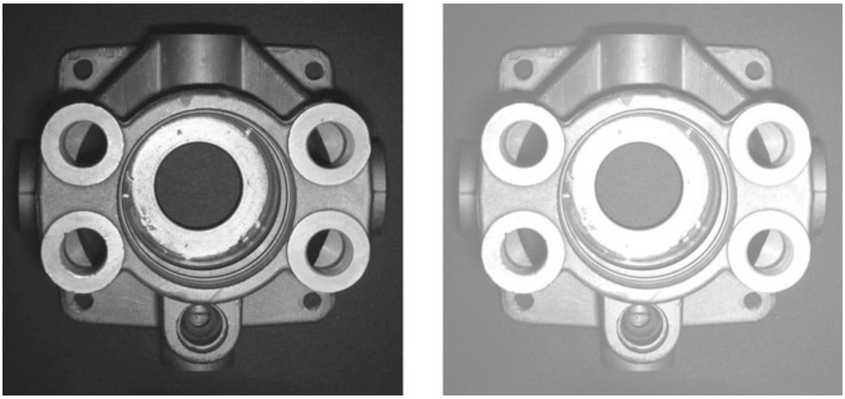


Fig. 14.7
Intensity shift. [*acn(100)*]. Notice that saturation occurs at level 255 (white)

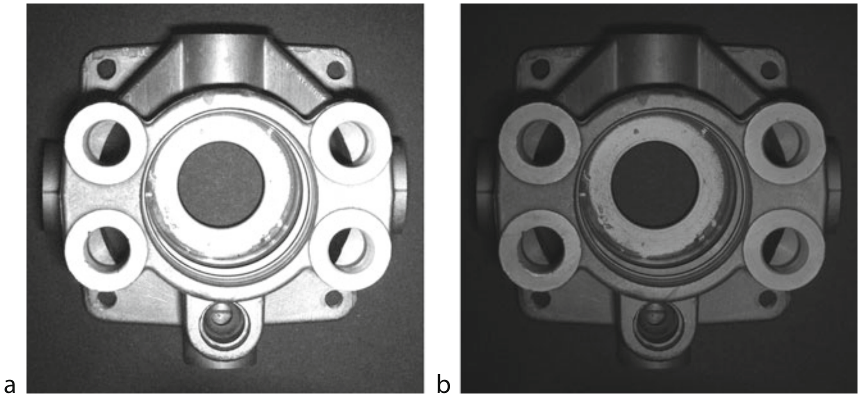


Fig. 14.8
Intensity multiply. (a) Intensities doubled [din]. (b) Intensities halved [*hin*. Also see *mcn*.]

Intensity Multiply (➋ *Fig. 14.8*) *[mcn]*

$$c(i,j) \Leftarrow \left[\begin{array}{cc} 0 & a(i,j)\cdot k < 0 \\ a(i,j)+k & 0 \le a(i,j)\cdot k \le W \\ W & W < a(i,j)\cdot k \end{array}\right.$$

Logarithm (➋ *Fig. 14.9*) *[log]*

$$c(i,j) \Leftarrow \left[\begin{array}{cc} 0 & a(i,j)=0 \\ W\dfrac{\log(a(i,j))}{\log(W)} & \text{otherwise} \end{array}\right.$$

This definition arbitrarily replaces the infinite value of log(0) by zero, and thereby avoids a difficult rescaling problem.

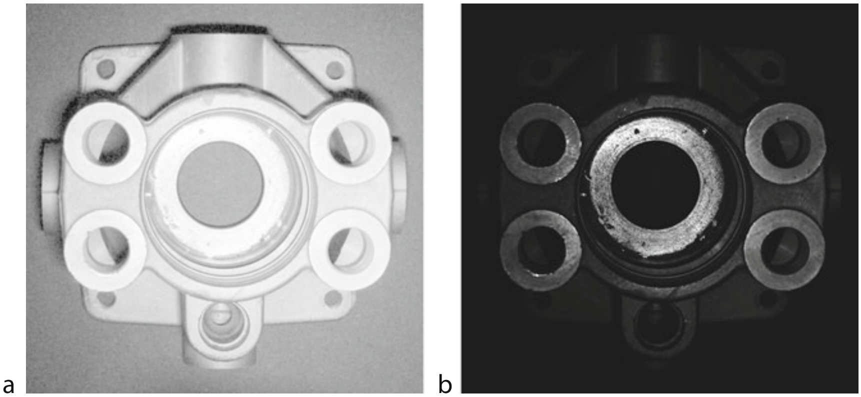


Fig. 14.9
(a) Logarithm of intensity [*nlg*]. (b) Antilogarithm (exponential) of intensity. [*alg*] Apart from zero intensity, these two operations are inverses of one another

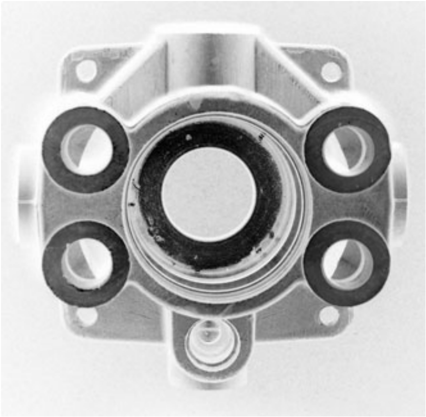


Fig. 14.10
Negation [*neg*]

Antilogarithm (exponential) (Fig. 14.9) *[exp]*

$$c(i, j) \Leftarrow W\exp(a(i, j))/\exp(W)$$

Negate (Fig. 14.10) *[neg]*

$$c(i, j) \Leftarrow W - a(i, j)$$

Threshold (Fig. 14.11) *[thr]*

$$c(i, j) \Leftarrow \begin{bmatrix} W & k1 \le a(i, j) < k2 \\ 0 & \text{otherwise} \end{bmatrix}$$

This is an important function, which converts a grey-scale image to a binary format. Unfortunately, it is often difficult or even impossible to find satisfactory values for the parameters *k1* and *k2*.

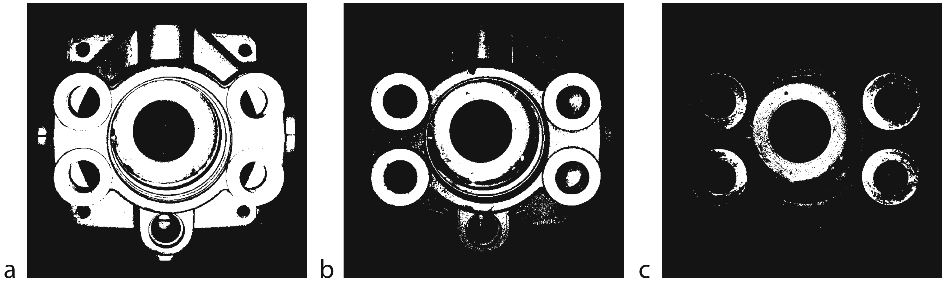


Fig. 14.11
Thresholding. (a) Thresholding at level 64. [thr(64)], (b) Thresholding at level 128. [*thr*, or *thr(128)*], and (c) Thresholding at level 192. [*thr(192)*]

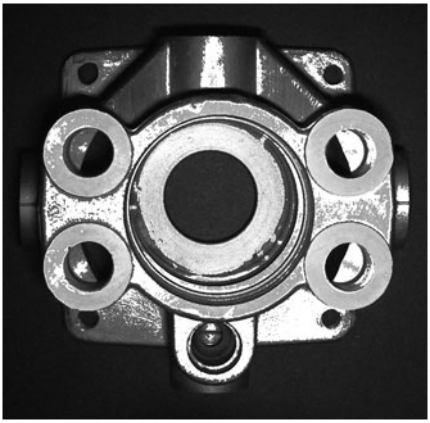


Fig. 14.12
Highlight. [*hil(100,150,255)*]

Highlight (Fig. 14.12) *[hil]*

$$c(i,j) \Leftarrow \begin{bmatrix} k3 & k1 \le a(i,j) < k2 \\ a(i,j) & \text{otherwise} \end{bmatrix}$$

Squaring (Fig. 14.13) *[sqr]*

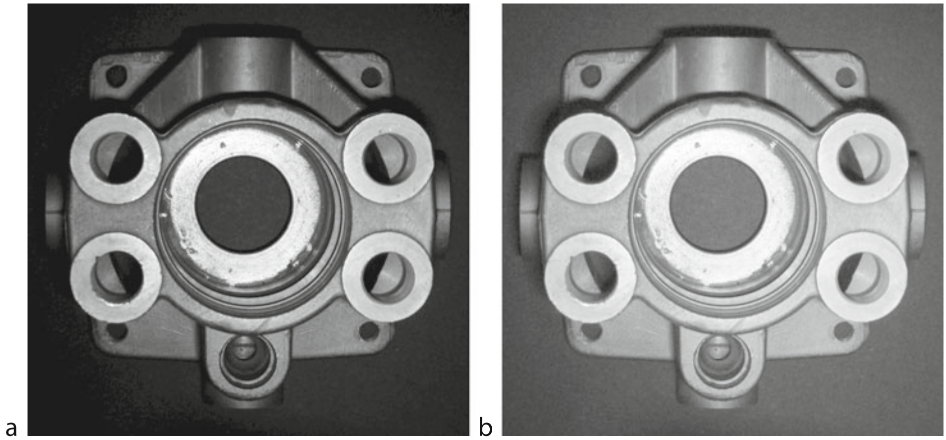
$$c(i,j) \Leftarrow [a(i,j)]^2/W$$

14.2.2 Dyadic, Pixel-by-Pixel Operators

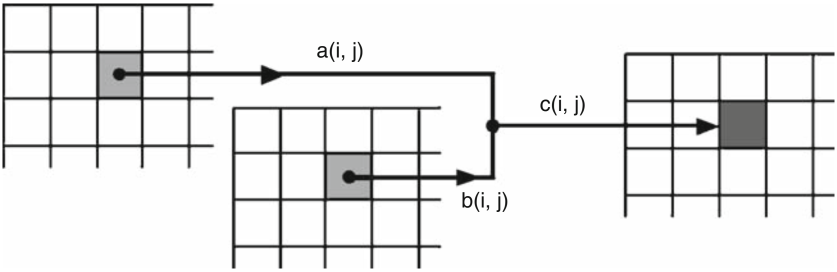
Dyadic operators have a characteristic equation of the form

$$c(i,j) \Leftarrow h(a(i,j),\ b(i,j))$$

There are two input images: **A** = {a(i,j)} and **B** = {b(i,j)} (Fig. 14.14), while the output image is ***C*** = {c(i,j)}. It is important to realise that c(i,j) depends upon only a(i,j) and b(i,j). Here are some examples of dyadic operators.



◘ Fig. 14.13
(a) Square intensities. [*sqr*] (b) Square root of intensities. [*sqt*]. These two operations are inverses of one another



◘ Fig. 14.14
Dyadic pixel-by-pixel operator. The intensities of the (i,j)th pixels in the two input images are combined to calculate the intensity c(i,j) at the corresponding address in the output image

Add (➔ *Fig. 14.15*) *[add]*

$$c(i,j) \Leftarrow [a(i,j) + b(i,j)]/2$$

Subtract (➔ *Fig. 14.16*) *[sub]*

$$c(i,j)[(a(i,j) - b(i,j)) + W]/2$$

Multiply (➔ *Fig. 14.17*) *[mul]*

$$c(i,j) \Leftarrow [a(i,j).b(i,j)]/W$$

Maximum (➔ *Fig. 14.18*) *[max]*

$$c(i,j) \Leftarrow \mathrm{MAX}[a(i,j), b(i,j)]$$

When the maximum operator is applied to a pair of binary images, the *union* (OR function) of their white areas is computed. This function may also be used to *superimpose* white writing onto a grey-scale image.

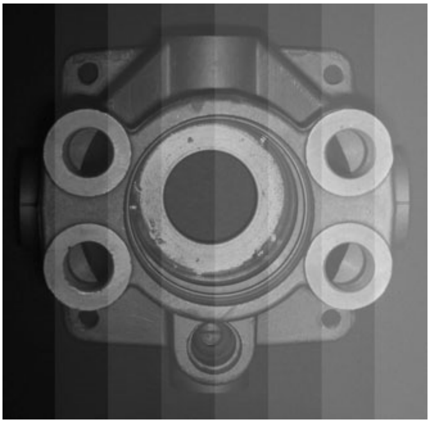


Fig. 14.15
Adding two images. [*adi*] The second image is a so-called "stair case", in which the intensity increases in doscrete steps along the X axis

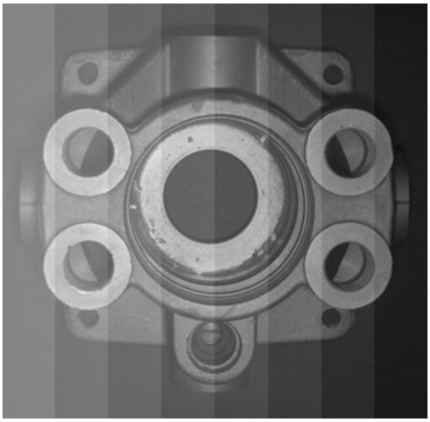


Fig. 14.16
Subtracting two images [*sub*]

Minimum (Fig. 14.19) *[min]*

$$c(i,j) \Leftarrow \mathrm{MIN}[a(i,j), b(i,j)]$$

When A and B are both binary, the *intersection* (AND function) of their white areas is calculated.

14.2.3 Local Operators

Figure 14.20 illustrates the principle of the operation of local operators. Notice that the intensities of several pixels are combined in order to calculate the intensity of just one pixel.

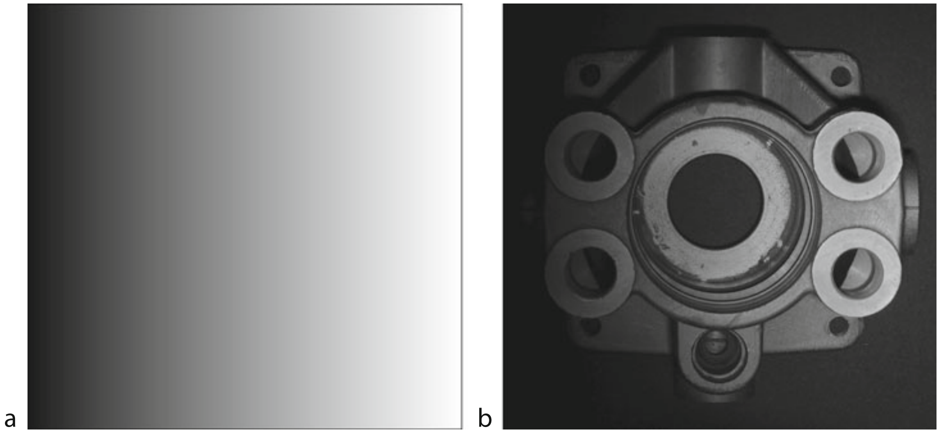


Fig. 14.17
Multiplying two images. (a) ''Intensity wedge''. The intensity varies linearly along the X axis. (b) Result of multiplying two images [*mul*]

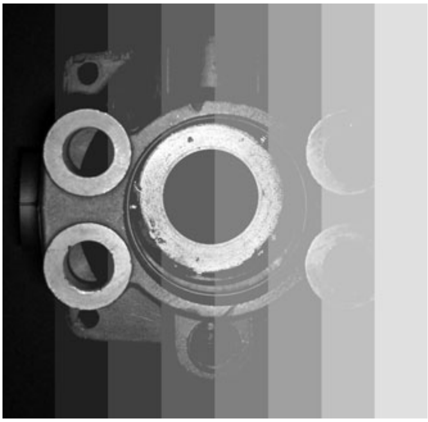


Fig. 14.18
Intensity maximum, using the ''stair case'' image [*mxi*]

Among the simplest of the local operators are those using a set of 9 pixels arranged in a 3×3 square. These 3×3 operators have a characteristic equation of the following form:

$$\begin{aligned} c(i,j) \Leftarrow g(&a(i-1,\ j-1), a(i-1,j), a(i-1,j+1), a(i,j-1), a(i,j),\\ &a(i,j+1), a(i+1,j-1), a(i+1,j), a(i+1,j+1)) \end{aligned}$$

where g(●) is a function of nine variables. This local operator is said to have a *3×3 processing window*. That is, it computes the value for one pixel on the basis of the intensities within a region containing 3×3 (= 9) pixels. Other local operators employ larger windows and we will discuss these briefly later. In the simplified notation which we introduced earlier, the above definition reduces to

$$E \Leftarrow g(A, B, C, D, E, F, G, H, I).$$

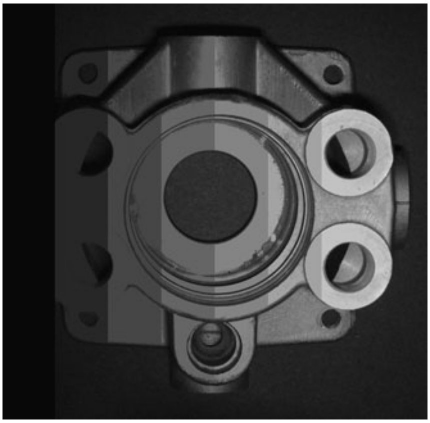


Fig. 14.19
Intensity minimum, using the "stair case" image [*mni*]

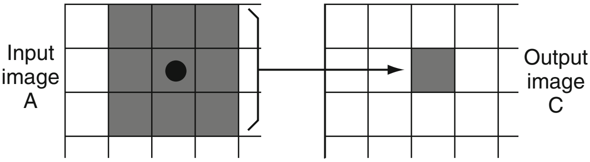


Fig. 14.20
Local operator. In this instance, the intensities of 9 pixels arranged in a 3×3 window are combined using linear or non-linear processes. Local operators may be defined which uses other, possibly larger, windows, which may or may not be square

14.2.4 Linear Local Operators

An important sub-set of the local operators is that group which performs a linear weighted sum, and which are therefore known as *linear local operators*. For this group, the characteristic equation is:

$$E \Leftarrow kl\,(A\,W1 + B\,W2 + C\ W3 + D\,W4 + E\,W5 \\ + F\,W6 + G\,W7 + H\,W8 + I\,W9) + k2$$

where W1, W2,...,W9 are weights, which may be positive, negative, or zero. Values for the normalisation constants k1 and k2 are given later. The matrix illustrated below is termed the *weight matrix*, and is important because it determines the properties of the linear local operator.

W1	W2	W3
W4	W5	W6
W7	W8	W9

The following rules summarise the behavior of this type of operator. They exclude the case where *all* the weights and normalisation constants are zero, since this would result in a null image:

1. If all weights are either positive or zero, the operator will *blur* the input image. Blurring is referred to as *low-pass filtering*. Subtracting a blurred image from the original results in

a highlighting of those points where the intensity is changing rapidly and is termed *high-pass filtering*.

2. If W1 = W2 = W3 = W7 = W8 = W9 = 0, and W4, W5, W6 > 0, then the operator blurs along the rows of the image. Horizontal features, such as horizontal edges and streaks, are not affected.
3. If W1 = W4 = W7 = W3 = W6 = W9 = 0, and W2, W5, W8 > 0, then the operator blurs along the columns of the image. Vertical features are not affected.
4. If W2 = W3 = W4 = W6 = W7 = W8 = 0, and W1, W5, W9 > 0, then the operator blurs along the diagonal (top-left to bottom-right). There is no smearing along the orthogonal diagonal.
5. If the weight matrix can be reduced to a matrix product of the form **PQ**, where

P =

0	0	0
V4	V5	V6
0	0	0

and

Q =

0	V1	0
0	V2	0
0	V3	0

the operator is said to be "separable." This is important because in such cases it is possible to apply two simpler operators in succession, with weight matrices **P** and **Q**, to obtain the same effect as that produced by the separable operator.

6. The successive application of linear local operators which use windows containing 3×3 pixels produces the same results as linear local operators with larger windows. For example, applying that operator which uses the weight shown below (left) twice in succession results in an image similar to that obtained from the 5×5 operator with the weight matrix shown below (centre). (For simplicity, normalisation has been ignored here.) Applying the same 3×3 operator three times in succession is equivalent to using the 7×7 operator shown below (right).

1	1	1
1	1	1
1	1	1

1	2	3	2	1
2	4	6	4	2
3	6	9	6	3
2	4	6	4	2
1	2	3	2	1

1	3	6	7	6	3	1
3	9	18	21	18	9	3
6	18	36	42	36	18	6
7	21	42	49	42	21	7
6	18	36	42	36	18	6
3	9	18	21	18	9	3
1	3	6	7	6	3	1

Notice that all of these operators are also separable. Hence it would be possible to replace the last-mentioned 7×7 operator with six simpler operators: 3×1, 3×1, 3×1, 1×3, 1×3, and 1×3, applied in any order. It is not always possible to replace a large-window operator with a succession of 3×3 operators. This becomes obvious when one considers, for example, that a 7×7 operator uses 49 weights and that three 3×3 operators provide only 27 degrees of freedom. Separation is often possible, however, when the larger operator has a weight matrix with some redundancy; for example, when it is symmetrical.

7. In order to perform normalisation, the following values are used for k1 and k2.

$$k1 \leftarrow 1/\left(\sum_{p,q} \left|W_{p,q}\right|\right)$$

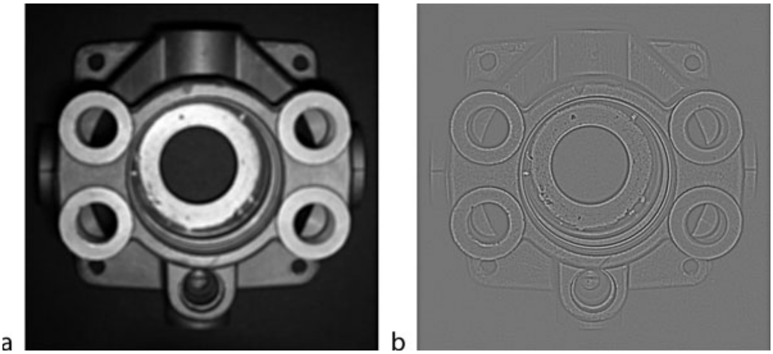


Fig. 14.21
Linear convolution filtering. (a) Low-pass (2D blurring) filter. [*raf(11,11)*] (b) High-pass filter. [*raf(11,11)sub*]

$$k2 \leftarrow \left| 1 - \left[\left(\sum_{p,q} W_{p,q} \right) \Big/ \left(\sum_{p,q} |W_{p,q}| \right) \right] \right|$$

8. A filter using the following weight matrix performs a *local averaging function* over an 11×11 window [*raf(11,11)*] (*Fig. 14.21*).

1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1

This produces quite a severe 2-directional blurring effect. Subtracting the effects of a blurring operation from the original image generates a picture in which spots, streaks and intensity steps are all emphasised. On the other hand, large areas of constant or slowly changing intensity become uniformly grey. This process is called *high-pass filtering*, and produces an effect similar to unsharp masking, which is familiar to photographers.

14.2.5 Nonlinear Local Operators

Largest Intensity Neighbourhood Function (*Fig. 14.22*) *[lnb]*

$$E \Leftarrow \mathbf{MAX}(A, B, C, D, E, F, G, H, I)$$

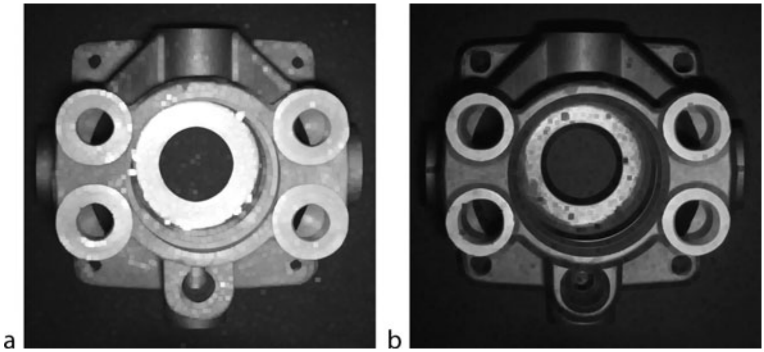


Fig. 14.22
Neighbourhood intensity operations (a) Largest neighbour. [*lnb(5)*]. (b) Smallest neighbour. [*snb(5)*]

This operator has the effect of spreading bright regions and contracting dark ones.

Edge Detector (Fig. 14.23) [lnb, sub]

$$E \Leftarrow \mathbf{MAX}(A, B, C, D, E, F, G, H, I) - E$$

This operator is able to highlight edges (i.e., points where the intensity is changing rapidly).

Median Filter (Fig. 14.24) *[mdf(5)]*

$$E \Leftarrow \mathbf{FIFTH\_LARGEST}\ (A, B, C, D, E, F, G, H, I)$$

This filter is particularly useful for reducing the level of noise in an image. Noise is generated from a range of sources, such as video cameras and x-ray detectors, and can be a nuisance if it is not eliminated by hardware or software filtering.

Crack Detector (Fig. 14.25) [*crk*. This is equivalent to applying the sequence *lnb, lnb, neg, lnb, lnb, neg* and then subtracting the result from the original image]. This detector is able to detect thin dark streaks and small dark spots in a grey-scale image; it ignores other features, such as bright spots and streaks, edges (intensity steps) and broad dark streaks. (This is an example of an operator that can be described far better using computer notation rather than mathematical notation.)

Roberts edge Detector (Fig. 14.23) *[red]* The Roberts gradient is calculated using a 2×2 pixel mask. This will determine the edge gradient in two diagonal directions. It is defined as.

$$E \Leftarrow \sqrt{(A - E)^2 + (B - D)^2}$$

The following approximation to the Roberts gradient magnitude is called the *Modified Roberts operator*. This is simpler and faster to implement and it more precisely defines the QT operator *red*. It is defined as

$$E \Leftarrow \{|A - E| + |B - D|\}/2$$

Sobel edge Detector (Fig. 14.23) *[sed]* This popular operator highlights the edges in an image; points where the intensity gradient is high are indicated by bright pixels in the output image. The Sobel edge detector uses a 3×3 mask to determine the edge gradient.

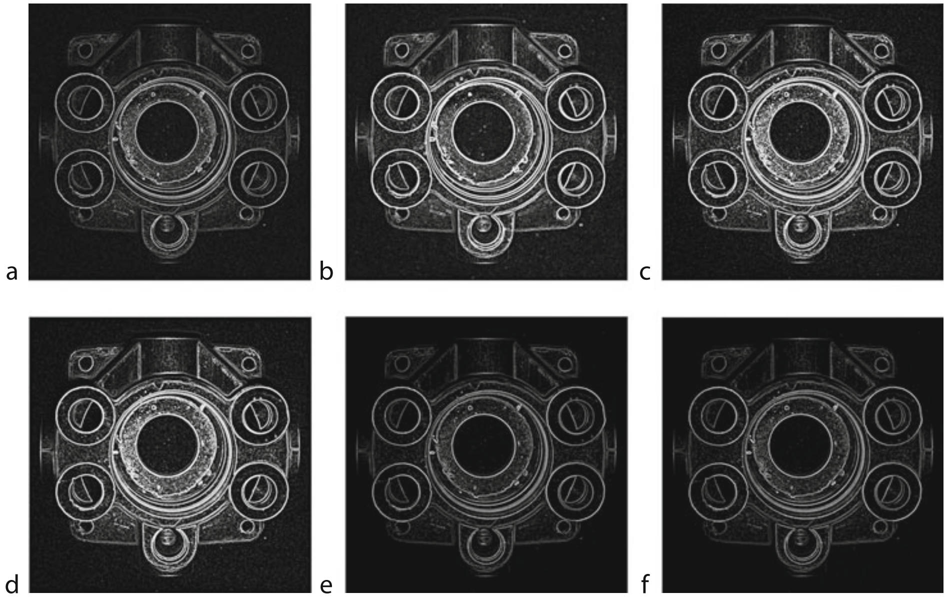


Fig. 14.23
Edge detectors. (a) Non-linear neighbourhood operator [*lnb,sub,neg,enc,din*]. (b) Another non-linear neighbourhood operator [*lnb(7), sub,neg, enc, din*]. (c) Roberts operator. [*red*] (d) Sobel operator [*sed*] (e) Prewitt operator [*wri, glc([−1,−1,−1;0,0,0;1,1,1]), abv, wri(1), rei, yxt, glc([−1,−1,−1;0,0,0;1,1,1]), yxt, abv,, rei(1), adi, enc*] (f) Frei-Chen operator [*wri, glc([−1,−1.4142,−1;0,0,0; 1,1.4142,1]), abv, wri(1), rei, yxt, glc([−1,−1.4142,−1;0,0,0; 1,1.4142,1]), yxt, abv,, rei(1), adi, enc*]

$$E \Leftarrow \sqrt{[(A+2B+C)-(G+2H+I)]^2+[(A+2D+G)-(C+2F+I)]^2}$$

The following approximation is simpler to implement in software and hardware and more precisely defines the QT operator *sed*:

$$E \Leftarrow \{|(A+2B+C)-(G+2H+I)|+|(A+2D+G)-(C+2F+I)|\}/6$$

See *Fig. 14.23* for a comparison of the Roberts and Sobel edge detector operators when applied to a sample monochrome image. Note that, while the Roberts operator produces thinner edges, these edges tend to break up in regions of high curvature. The primary disadvantage of the Roberts operator is its high sensitivity to noise, since fewer pixels are used in the calculation of the edge gradient. There is also a slight shift in the image when the Roberts edge detector is used. The Sobel edge detector does not produce such a shift.

Prewitt Edge Detector (*Fig. 14.23*) The Prewitt edge detector is similar to the Sobel operator, but is more sensitive to noise because it does not possess the same inherent smoothing. This operator uses the two 3×3 weighting matrices shown below to determine the edge gradient.

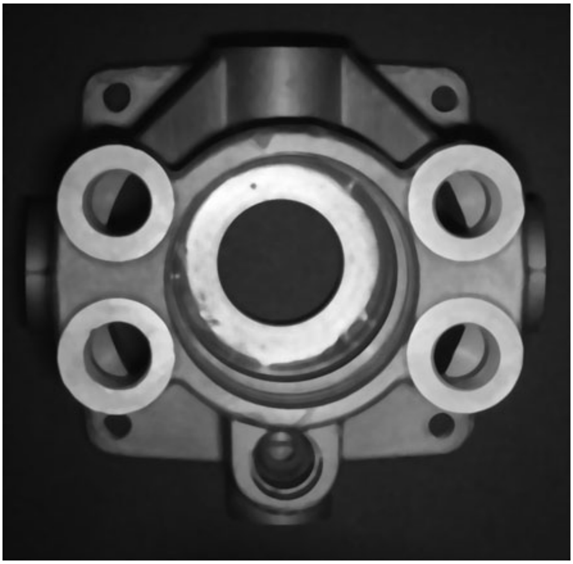


Fig. 14.24
Median filter, based on an 11×11 pixel window. [*mdf(11)*]

P_1:

–1	–1	–1
0	0	0
1	1	1

P_2:

–1	0	1
–1	0	1
–1	0	1

The Prewitt gradient magnitude is defined as $E \Leftarrow \surd(P_1{}^2 + P_2{}^2)$, where P_1 and P_2 are the values calculated from each mask, respectively.

Frei and Chen Edge Detector (➔ *Fig. 14.23*) This operator uses the two 3×3 masks shown below to determine the edge gradient.

F_1:

–1	–√2	–1
0	0	0
1	√2	1

F_2:

–1	0	1
–√2	0	√2
–1	0	1

The Frei and Chen gradient magnitude is defined as $E \Leftarrow \surd(F_1{}^2 + F_2{}^2)$, where F_1 and F_2 are the values calculated from each mask, respectively.

Rank Filters (➔ *Fig. 14.26*) *[mdf, rid]* The generalised 3×3 rank filter is

$$\begin{aligned} c(i,j) \Leftarrow\ & k1(A'W1 + B'W2 + C'W3 + D'W4 + E'W5 \\ & + F'W6 + G'W7 + H'W8 + I'W9) + k2 \end{aligned}$$

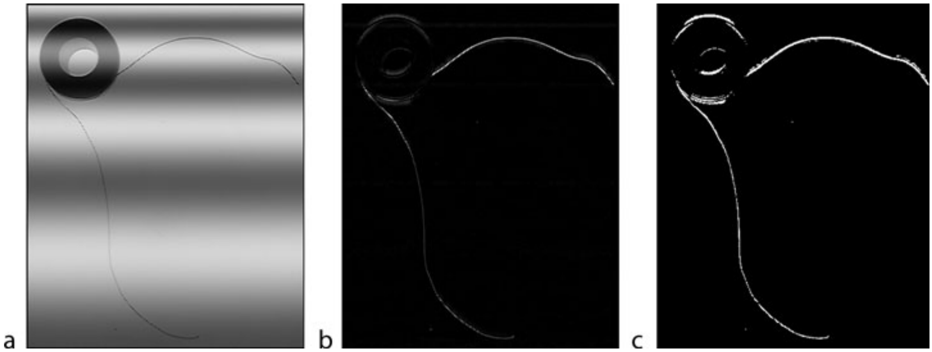


Fig. 14.25
Crack detector function. (a) Original image, specially contrived to show the power of this operator. (b) Crack detector operator applied. Contrast was increased for easier viewing. [*crk(5), enc*] (c) Thesholding applied to previous image shows that the crack detector creates an image with sufficient contrast to enable the thin dark streaks (wires) to be isolated

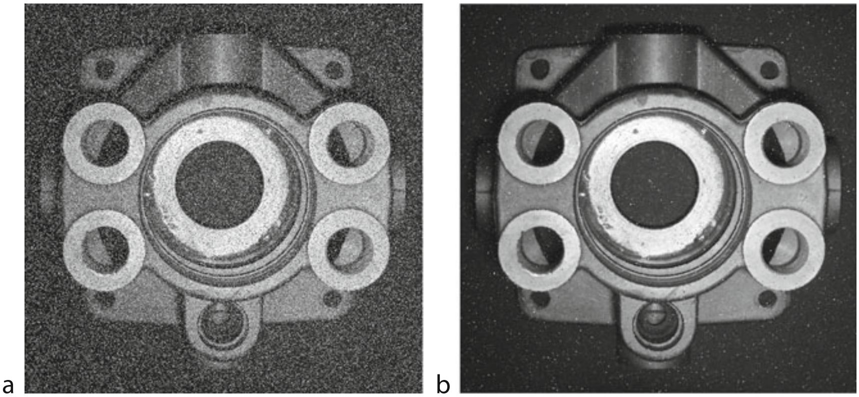


Fig. 14.26
Rank filter. (a) Original image. Noise was added specially to show the power of this operator. (b) Rank filter applied [*rcf([0,0,1,2,3,2,1,0,0]*)]

where

A′ = **LARGEST** (A, B, C, D, E, F, G, H, I)
B′ = **SECOND_LARGEST** (A, B, C, D, E, F, G, H, I)
C′ = **THIRD_LARGEST** (A, B, C, D, E, F, G, H, I)
…
I′ = **NINTH_LARGEST** (A, B, C, D, E, F, G, H, I)

and k1 and k2 are the normalisation constants defined previously. With the appropriate choice of weights (W1, W2,...,W9), the rank filter can be used for a range of operations including edge detection, noise reduction, edge sharping, and image enhancement.

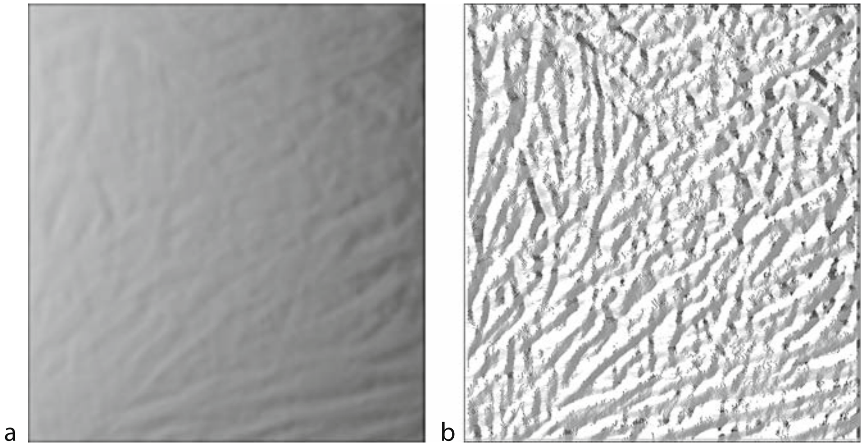


Fig. 14.27

Direction of brightest neighbour. (a) Original image (creased fabric). (b) Operator applied. [*mdf(5), caf(5), dbn,enc*] (The command sequence [*mdf(5), caf(5)*] reduces noise and thereby produces a "smoother" result.)

Direction Codes (Fig. 14.27) *[dbn]* This function can be used to detect the *direction* of the intensity gradient. A direction code function DIR_CODE is defined thus:

$$\mathrm{DIR\_CODE(A,B,C,D,F,G,H,I)} \Leftarrow \left[\begin{array}{ll} 1 & \text{if } \mathrm{A} \geq (\mathrm{MAX(B,C,D,F,G,H,I)} \\ 2 & \text{if } \mathrm{B} \geq (\mathrm{MAX(A,C,D,F,G,H,I)} \\ 3 & \text{if } \mathrm{C} \geq (\mathrm{MAX(A,B,D,F,G,H,I)} \\ 4 & \text{if } \mathrm{D} \geq (\mathrm{MAX(A,B,C,F,G,H,I)} \\ 5 & \text{if } \mathrm{F} \geq (\mathrm{MAX(A,B,C,D,G,H,I)} \\ 6 & \text{if } \mathrm{G} \geq (\mathrm{MAX(A,B,C,D,F,H,I)} \\ 7 & \text{if } \mathrm{H} \geq (\mathrm{MAX(A,B,C,D,F,G,I)} \\ 8 & \text{if } \mathrm{I} \geq (\mathrm{MAX(A,B,C,D,F,G,H)} \end{array}\right.$$

Using this definition, the operator *dbn* may be defined as

$$\mathrm{E} \Leftarrow \mathrm{DIR\_CODE(A,B,C,D,F,G,H,I)}$$

14.2.6 N-Tuple Operators

The N-tuple operators are closely related to the local operators and have a large number of linear and non-linear variations. N-tuple operators may be regarded as generalised versions of local operators. In order to understand the N-tuple operators, let us first consider a *linear* local operator which uses a large processing window, say r^*s pixels, with most of its weights equal to zero. Only N of the weights are non-zero, where N $\ll$ r^*s. This is an N-tuple filter (see Fig. 14.28). These filters are usually designed to detect specific patterns. In this role, they are able to locate a simple feature, such as a corner, an annulus, the numeral "2" in any position, etc. However, they are sensitive to changes of orientation and scale. The N-tuple can be regarded as a sloppy template which is convolved with the input image.

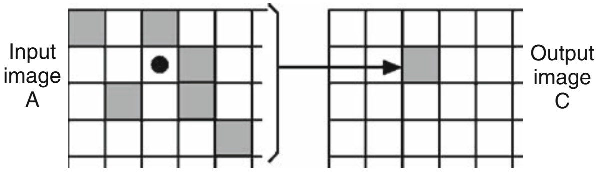


Fig. 14.28
An N-tuple filter operates much like a local operator. The only difference is that the pixels whose intensities are combined do not form a compact set. An N-tuple filter can be regarded as being equivalent to a local operator with a large window in which many of the weights are zero

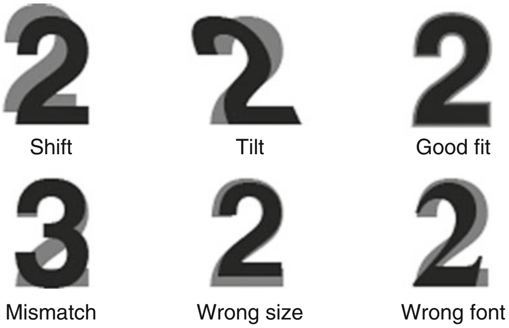


Fig. 14.29
Recognising a numeral "2" using an N-tuple

Nonlinear N-tuple operators may be defined in a fairly obvious way. For example, we may define operators which compute the average, maximum, minimum, or median values of the intensities of the N pixels covered by the N-tuple. An important class of such functions is the *morphological operators* (see Sects. 14.4 and 14.5). *Figure 14.29* illustrates the recognition of the numeral "**2**" using an N-tuple. Notice how the goodness of fit varies with the shift, tilt, size, and font. Another character ("**Z**" in this case) may give a score that is close to that obtained from a "**2**", thus making these two characters difficult to distinguish reliably.

14.2.7 Edge Effects

All local operators and N-tuple filters are susceptible to producing peculiar effects around the edges of an image. The reason is simply that, in order to calculate the intensity of a point near the edge of an image, we require information about pixels outside the image, which of course are simply not present. In order to make some attempt at calculating values for the edge pixels, it is necessary to make some assumptions, for example that all points outside the image are black, or have the same values as the border pixels. This strategy, or whatever one we adopt, is perfectly arbitrary and there will be occasions when the edge effects are so pronounced that there is nothing that we can do but to remove them by masking (*Fig. 14.30*) [*edg*]. Edge

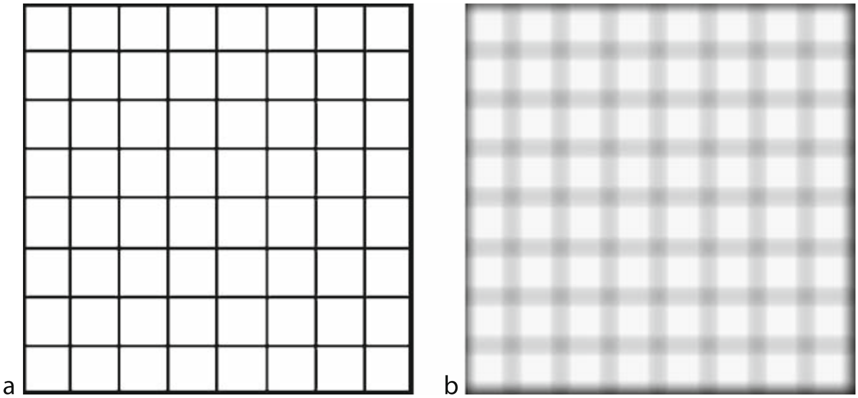


Fig. 14.30
Edge effects of a local operator. (a) Input image. (b) Blurring filter applied. Pixels near the edge do not have meaningful intensity values

effects are important because they require us to make special provisions for them when we try to patch several low-resolution images together.

14.2.8 Intensity Histogram (Fig. 14.31) *[hpi, hgi, hge, hgc]*

The intensity histogram is defined in the following way:

(a) Let

$$s(p,i,j) \leftarrow \begin{bmatrix} 1 & a(i,j) = p \\ 0 & \text{otherwise.} \end{bmatrix}$$

(b) Let h(p) be defined thus:

$$h(p) \leftarrow \sum_{i,j} s(p,i,j)$$

It is not, in fact, necessary to store each of the *s(p,i,j)*, since the calculation of the histogram can be performed as a serial process in which the estimate of *h(p)* is updated iteratively as we scan through the input image. This *cumulative histogram, H(p)*, can be calculated using the following recursive relation:

$$H(p) = H(p-1) + h(p), \text{ where } H(0) = h(0).$$

Both the cumulative and the standard histograms have a great many uses, as will become apparent later. It is possible to calculate various intensity levels which indicate the occupancy of the intensity range [*pct*]. For example, it is a simple matter to determine that intensity level *p(k)* which, when used as a threshold parameter, ensures that a proportion *k* of the output image is black. *p(k)* can be calculate using the fact that *H(p(k)) = m n k*. The *mean intensity* [*avg*] is equal to

$$avg = \sum_{p} (h(p) \times p)/(m \times n)$$

while the *maximum intensity* [*gli*] is equal to *MAX*(p | h(p) > 0) and the *minimum intensity* is equal to *MIN*(p | h(p) > 0).

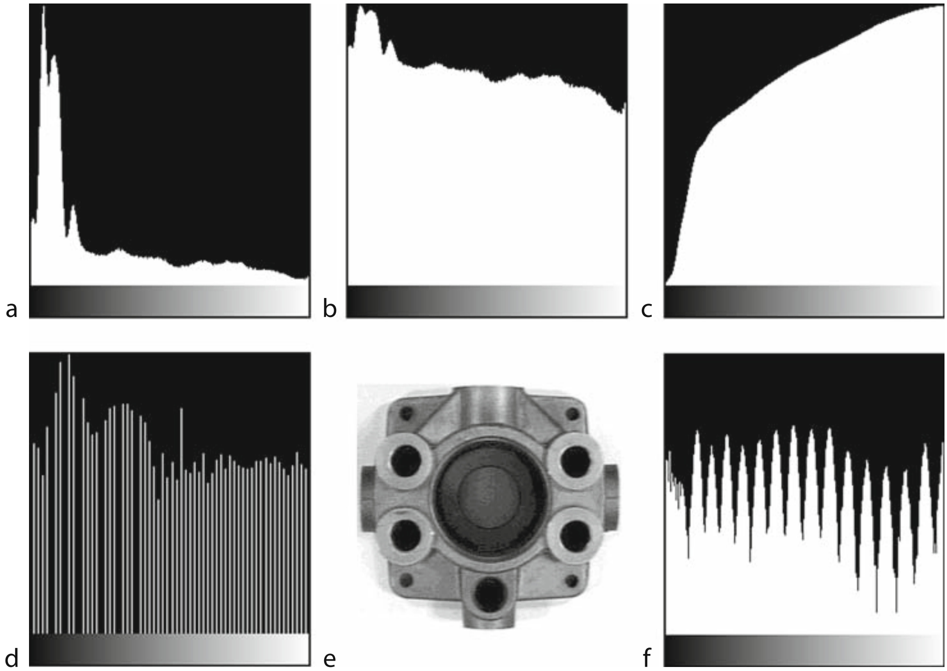


Fig. 14.31

Histograms. (a) Histogram of the image shown in *Fig. 14.2a*. [*hgm(1)*] (b) Histogram plotted with logarithmic ordinate scale. [*hgm(2)*] (c) Cumulative histogram. (This has the same shape as the mapping function required for histogram equalisation.) [*hgm(5)*] (d) Histogram equalisation. [*heq*] (e) Another image. (f) Histogram of (e) has a strange form, despite the apparently good image quality

One of the principal uses of the histogram is in the selection of threshold parameters. It is useful to plot *h(p)* as a function of *p*. It is often found from this graph that a suitable position for the threshold can be related directly to the position of the "foot of the hill" or to a "valley" in the histogram.

An important operator for image enhancement is given by the transformation

$$c(i,j) \Leftarrow [W\,H(a(i,j))]/(m \times n)$$

This has the interesting property that the histogram of the output image {c(i,j)} is flat, giving rise to the name *histogram equalisation* [*heq*]. Notice that histogram equalisation is a *data-dependent* monadic pixel-by-pixel operator.

An operation known as *local area histogram equalisation* relies upon the application of histogram equalisation within a small window (*Fig. 14.32*) [lhq]. The number of pixels in a small window that are darker than the central pixel is counted. This number defines the intensity at the equivalent point in the output image. This is a powerful filtering technique, which is particularly useful in texture analysis. (Sect. 14.7)

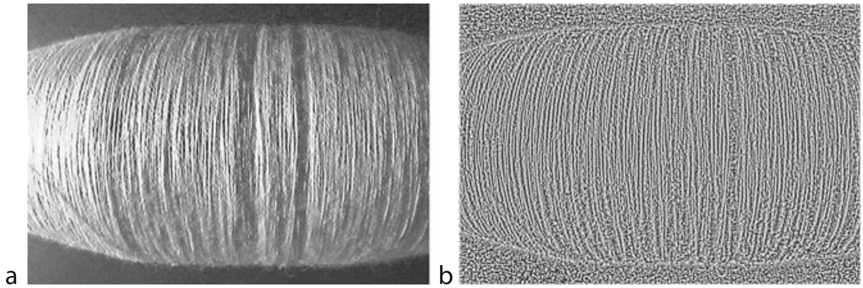


Fig. 14.32
Local area histogram equalisation. (a) Original image. (Cotton wound on a bobbin.) (b) After processing. Notice that the camera noise has been enhanced too. (The contrast was increased to improve visibility.) [*lhq,enc*]

14.3 Binary Images

For the purposes of this description of binary image processing, it will be convenient to assume that a(i,j) and b(i,j) can assume only two values: 0 (black) and 1 (white). The operator "+" denotes the Boolean *OR* operation, "×" represents the *AND* operation, "⊗" denotes the Boolean *Exclusive OR* operation, and #(i,j) denotes the number of white points addressed by N(i,j), including (i,j) itself.

Inverse (▶ *Fig. 14.33*) *[not]*

$$c(i,j) \Leftarrow ; \mathrm{NOT}(a(i,j))$$

AND (▶ *Fig. 14.34*) [mni]

$$c(i,j) \Leftarrow a(i,j) \times b(i,j)$$

OR (▶ *Fig. 14.35*) [mxi]

$$c(i,j) \Leftarrow a(i,j) + b(i,j)$$

Exclusive OR (▶ *Fig. 14.36*) [exr]

$$c(i,j) \Leftarrow a(i,j) \otimes b(i,j)$$

This finds differences between white regions.

Expand White Areas (▶ *Fig. 14.37*) *[dil]*

$$\begin{aligned} c(i,j) \Leftarrow\ & a(i-1,\ j-1) + a(i-1,j) + a(i-1,j+1) + a(i,j-1) + a(i,j) \\ & + a(i,j+1) + a(i+1,j-1) + a(i+1,j) + a(i+1,j+1) \end{aligned}$$

Notice that this is closely related to the local operator *lnb* defined earlier. This equation may be expressed in the simplified notation:

$$E \Leftarrow A + B + C + D + E + F + G + H + I$$

Shrink White Areas (▶ *Fig. 14.37*) *[ero]*

$$\begin{aligned} c(i,j) \Leftarrow\ & a(i-1,\ j-1) \times a(i-1,j) \times a(i-1,j+1) \times a(i,j-1) \times a(i,j) \\ & \times a(i,j+1) \times a(i+1,j-1) \times a(i+1,j) \times a(i+1,j+1) \end{aligned}$$

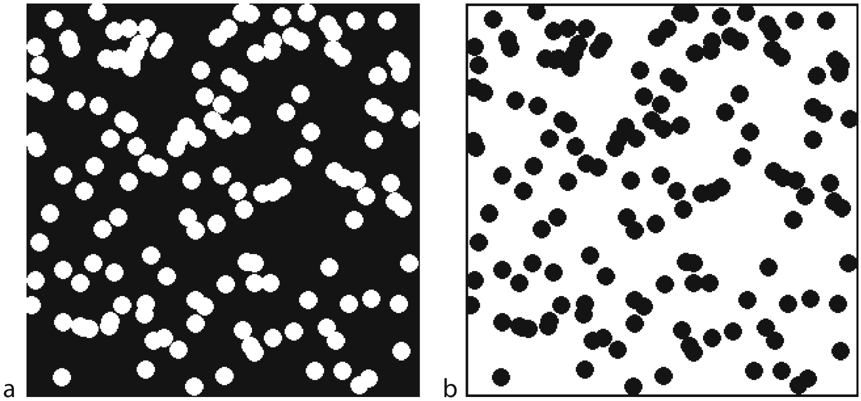


Fig. 14.33
Negating a binary image. (a) Original image. (b) Result of processing [*neg*]

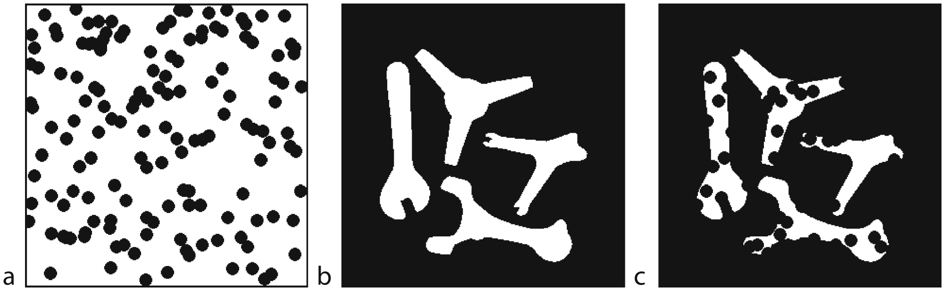


Fig. 14.34
ANDing two images. (a) Image 1. (b) Image 2. (c) Result of processing [*mni*. The grey-scale operators, neg *(NOT), mni (AND)* and *mxi (OR*) are also able to process binary images.]

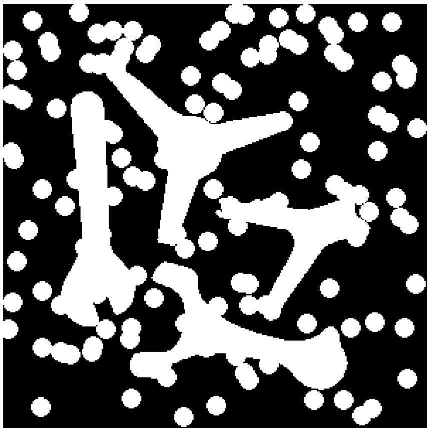


Fig. 14.35
ORing two images. The "spots" image in *Fig. 14.33a* was negated before the OR operation was performed [mxi]

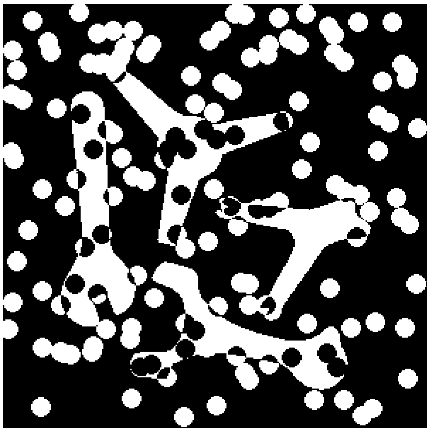


Fig. 14.36
Exclusive OR of two images: *Fig. 14.33a,b*

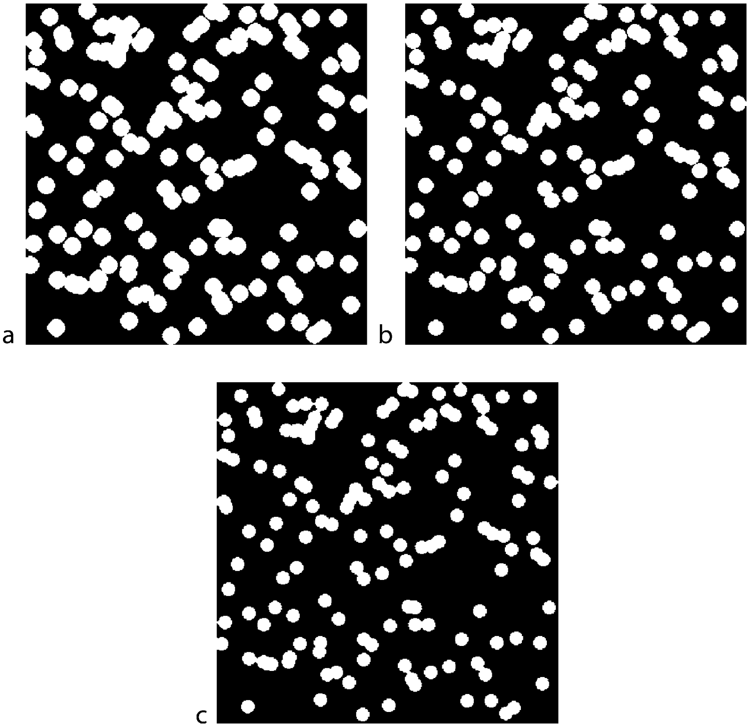


Fig. 14.37
Expanding (dilating) and shrinking (eroding) white regions in a 3×3 pixel neighbourhood. (a) Expansion. [*dil*]. (b) Original image. (c) Erosion [*ero*]

or more simply

$$c(i,j) \Leftarrow A \times B \times C \times D \times E \times F \times G \times H \times I$$

Expanding and shrinking white areas will be discussed again but in more general terms later in this chapter.

Edge detector (➲ *Fig. 14.38*) *[bed]*

$$c(i,j) \Leftarrow E \times \mathrm{NOT}(A \times B \times C \times D \times F \times G \times H \times I)$$

Remove Isolated White Points [cln]

$$c(i,j) \Leftarrow \left| \begin{array}{ll} 1 & a(i,j) \cdot (\#(i,j) > 1) \\ 0 & \text{otherwise} \end{array} \right.$$

Count White Neighbours (➲ *Fig. 14.39*) *[cnw]*

$$c(i,j) \Leftarrow N(i,j)$$

where #(Z) is the number of times Z occurs. This results in a grey-scale image.

Connectivity Detector (➲ *Fig. 14.39*) *[cny]* Consider the following pattern:

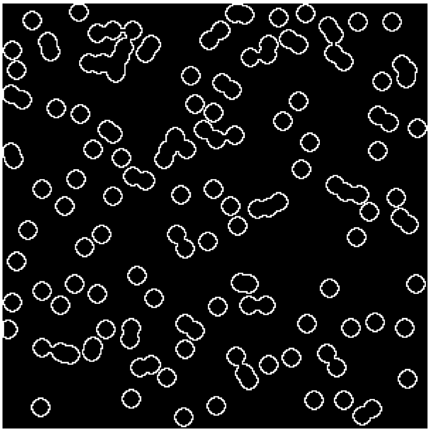
1	0	1
1	X	1
1	0	1

If X = 1, then all of the 1s are 8-connected to each other. Alternatively, if X = 0, then they are not connected. In this sense, the point marked X is *critical for connectivity.* This is also the case in the following examples:

1	0	0
0	X	1
0	0	0

1	1	0
0	X	0
0	0	1

0	0	1
1	X	0
1	0	1



▪ Fig. 14.38

Binary edge. This is the exclusive OR of ➲ *Fig. 14.37a, b* [*bed*]

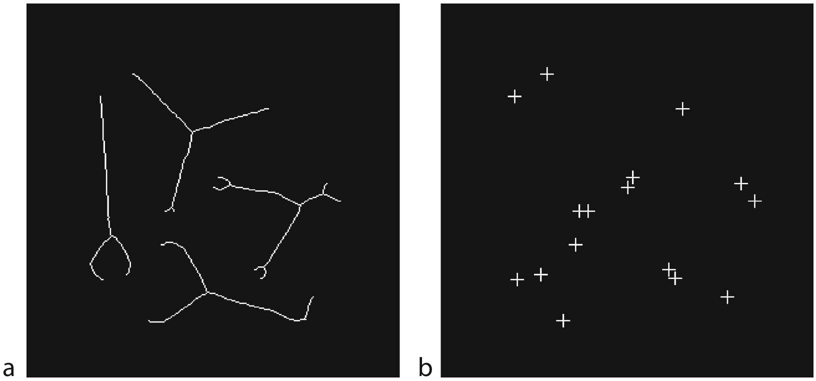


Fig. 14.39

Counting white neighbours. (a) Original image. This is the result of applying the thinning operator [thn] to *Fig. 14.34b*. (b) Limb ends [*lmb*] are white points that have exactly one white 8-neighbour. Joints are white points that have more than two white 8-neighbours. Since both limb ends and joints [*jnt*] are single pixels, they are too small to be visible and are represented here by crosses

However, those points marked X below are not critical for connectivity, since setting X = 0 rather than 1 has no effect on the connectivity of the remaining 1s.

1	1	1
1	X	1
0	0	1

0	1	1
1	X	0
1	1	1

0	1	1
1	X	0
0	1	1

A connectivity detector shades the output image with 1s to indicate the position of those points which are critical for connectivity and which were white in the input image. Black points and those which are not critical for connectivity are mapped to black in the output image.

Euler Number [eul] The Euler number is defined as the number of connected components (blobs) minus the number of holes in a binary image. The Euler number represents a simple method of counting blobs in a binary image, provided they have no holes in them. Alternatively, it can be used to count holes in a given object, providing they have no "islands" in them. The reason why this approach is used to count blobs, despite the fact that it may seem a little awkward to use, is that the Euler number is very easy and fast to calculate. It is also a useful means of classifying shapes in an image. The Euler number can be computed by using three local operators. Let us define three numbers N1, N2 and N3, where Nα, $\alpha = 1, 2$ or 3, indicates the number of times that one of the patterns in the pattern set α occurs in the input image.

0	0	0	0
0	1	1	0
0	1	1	0
0	0	0	0

Pattern Set 1 (N1)

0	1
1	0
1	0
0	1

Pattern Set 2 (N2)

1	1	1	1
1	0	0	1
1	0	0	1
1	1	1	1

Pattern Set 3 (N3)

The *8-connected* Euler number, where holes and blobs are defined in terms of 8-connected figures, is defined as *(N1-2 N2-N3)/4*. It is possible to calculate the *4-connected* Euler number using a slightly different formula, but this parameter can give results which seem to be anomalous when we compare them to the observed number of holes and blobs.

Filling Holes (➋ *Fig. 14.40*) [blf] Consider a white blob-like figure containing a hole (lake), against a black background. The application of the hole-filling operator will cause all of the holes to be *filled in* by setting all pixels in the holes to white. This operator will not alter the outer edge of the figure.

Region Labelling (➋ *Fig. 14.41*) *[ndo]* Consider an image containing a number of separate blob-like figures. A region-labelling operator will shade the output image so that each blob is given a separate intensity value. We could shade the blobs according to the order in which they

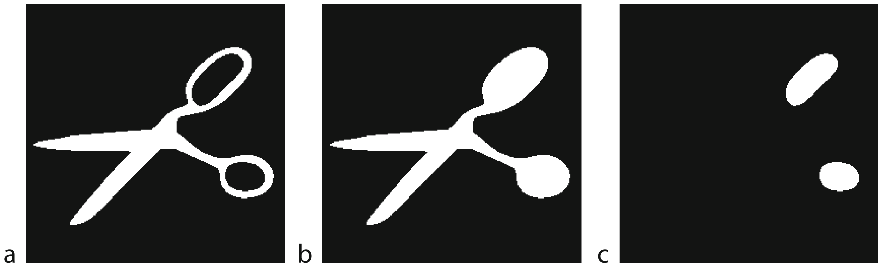


Fig. 14.40
Filling holes (lakes). (a) Original image. This has an Euler number of −1, since there is one object present and there are two lakes. (b) Result of filling. [*blf*] (c) Lakes identified using exclusive OR. [*blf,exr*]

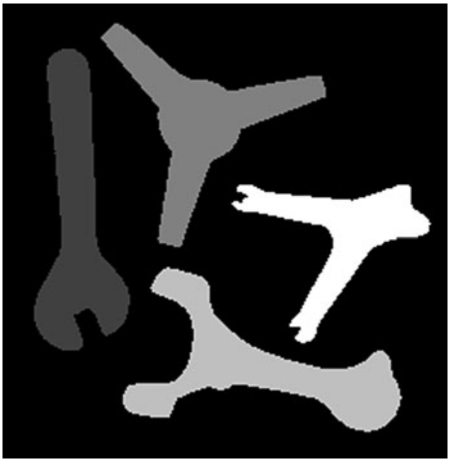


Fig. 14.41
Region labelling [*ndo*]

are found during a conventional raster scan of the input image. Alternatively, the blobs could be shaded according to their areas, the biggest blobs becoming the brightest. This is a very useful operator, since it allows objects to be separated and analysed individually. Small blobs can also be eliminated from an image using this operator [also *kgr*]. Region labelling can also be used to count the number of distinct binary blobs in an image. Unlike the Euler number, blob counting based on region labelling is not affected by the presence of holes.

Other Methods of Detecting/Removing Small Spots A binary image can be thought of as a grey-scale image in which only two grey levels, 0 and W, are allowed. The result of the application of a conventional low-pass (blurring) filter to such an image is a grey-scale image in which there is a larger number of possible intensity values. Pixels which were well inside large white areas in the input image are mapped to very bright pixels in the output image. Pixels which were well inside black areas are mapped to very dark pixels in the output image. However, pixels which were inside small white spots in the input image are mapped to mid-grey intensity levels (➲ *Fig. 14.42*). Pixels on the edge of large white areas are also mapped to mid-grey intensity levels. However, if there is a cluster of small spots, which are closely spaced together, some of them may also disappear.

This technique is generally easier and faster to implement than the blob shading technique described previously (➲ *Fig. 14.43*). Although it may not achieve precisely the desired result, it can be performed at high speed.

An N-tuple filter having the weight matrix illustrated below, can be combined with simple thresholding to distinguish between large and small spots. Assume that there are several small white spots within the input image and that they are spaced well apart. All pixels within a spot which can be contained within a circle of radius three pixels will be mapped to white by this particular filter. Pixels within a larger spot will become darker than this. The image is then thresholded at white to separate the large and small spots.

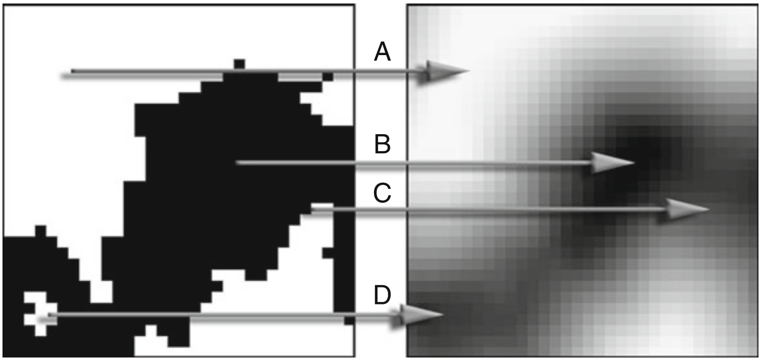


Fig. 14.42

Edge smoothing by low-pass filtering (blurring) and thresholding. *Left*: Original image. *Right*: Filtered image. A – points well within the white region are mapped to white. B – points well within the black region are mapped to black. C – points near the edge are mapped to grey; the intensity level depends on how many white points are in its immediate neighbourhood. D – small white spots are mapped to a low intensity and are eliminated by thresholding

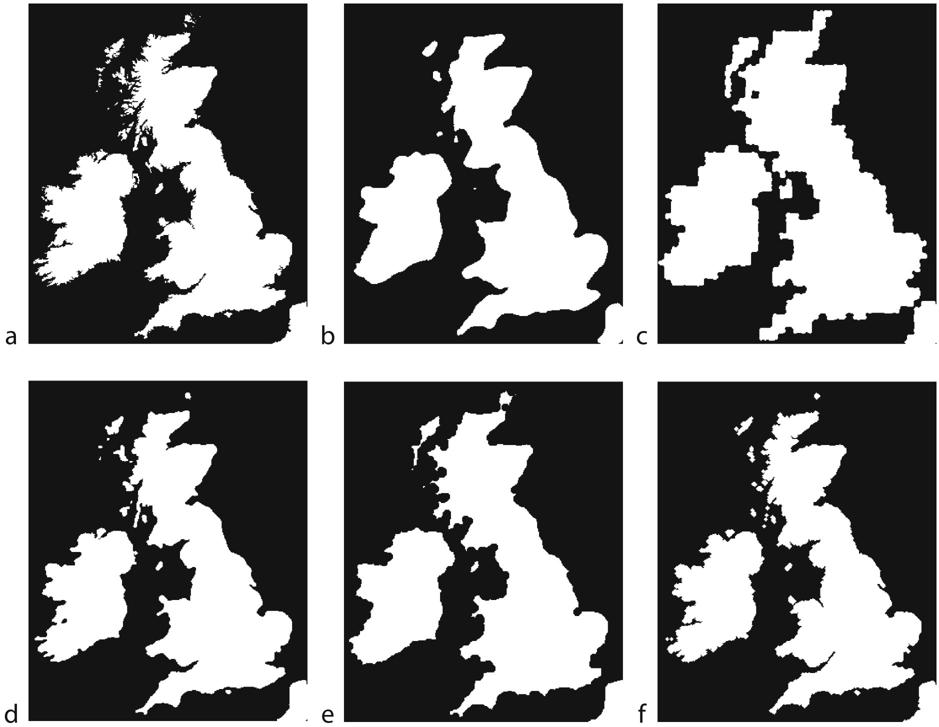


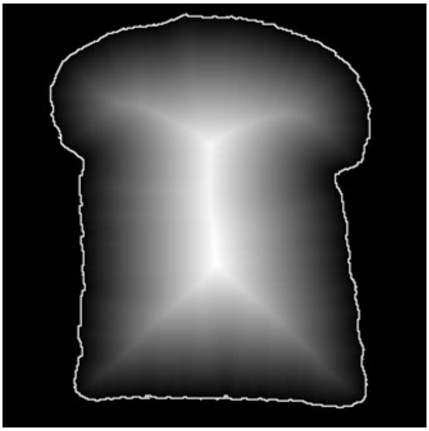
Fig. 14.43

Edge smoothing. (a) Binary image: map of Britain and Ireland. (b) After low-pass filtering (blurring the greyscale image) and thresholding. [*caf, thr*] (c) Majority voting in a 3×3 pixel neighbourhood. [*maj*] (This operator works well on smooth objects that have fine hair-like protuberances.) (d) Median filter, 5×5 pixel neighbourhood. [*mdf(5,5)*] (e) Closing. [*dil(5), ero(5)*] (f) Opening. [*ero(2), dil(2)*]

			–1	–1	–1			
		–1				–1		
	–1						–1	
–1								–1
–1				20				–1
–1								–1
	–1						–1	
		–1				–1		
			–1	–1	–1			

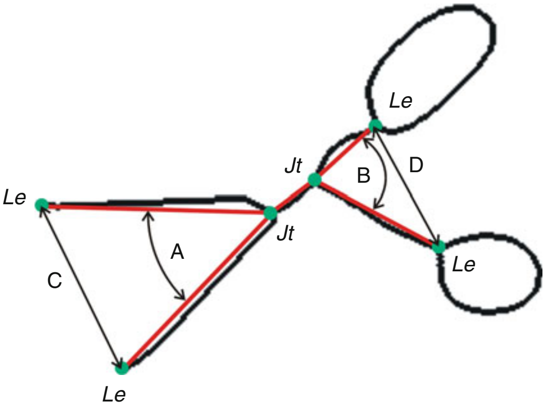
Grass-fire Transform and Skeleton [gft, ske & thn] Consider a binary image containing a single white blob, ➲ *Fig. 14.44*. Imagine that a fire is lit at all points around the blob's outer edge and the edges of any holes it may contain. The fire will burn inwards, until at some instant, advancing fire lines meet. When this occurs, the fire becomes extinguished locally. An output image is generated and is shaded in proportion to the time it takes for the fire to reach each point. Background pixels are mapped to black.

The importance of this transform, referred to as the *grass-fire* transform, lies in the fact that it indicates distances to the nearest edge point in the image [1]. It is therefore possible to distinguish thin and fat limbs of a white blob. Those points at which the fire lines meet are known as *quench* points. The set of quench points form a "match-stick" figure, usually referred to as a *skeleton* or *medial axis transform*. These figures can also be generated in a number of different ways (➋ *Fig. 14.45*).



◘ Fig. 14.44

Grassfire transform. The white contour shows the outer edge of the object. (slice of bread) [*gft*] The medial axis is the bright "spine"



◘ Fig. 14.45

Analysing the skeleton of a blob-like object. *Jt* repesents a joint on the skeleton and *Le* a limb end. Various measurements can be made on the skeleton, for example the distances between limb ends (C and D) and angles (A and B)

One such approach is described as *onion-peeling*. Consider a single white blob and a "bug" which walks around the blob's outer edge, removing one pixel at a time. No edge pixel is removed, if by doing so we would break the blob into two disconnected parts. In addition, no white pixel is removed, if there is only one white pixel among its 8-neighbours. This simple procedure leads to an undesirable effect in those instances when the input blob has holes in it; the skeleton which it produces has small loops in it which fit around the holes like a tightened noose. More sophisticated algorithms have been devised which avoid this problem.

Edge Smoothing and Corner Detection Consider three points B1, B2, and B3 which are placed close together on the edge of a single blob in a binary image (see ➋ *Fig. 14.46*). The perimeter distance between B1 and B2 is equal to that between B2 and B3. The point P is at the centre of the line joining B1 and B3. As the three points now move around the edge of the blob, keeping the spacing between them constant, the locus of P traces a smoother path than that followed by B2. This forms the basis of a simple edge smoothing procedure.

A related algorithm, used for corner detection, shades the edge according to the distance between P and B2. This results in an image in which the corners are highlighted, while the smoother parts of the image are much darker (➋ *Fig. 14.47*).

Many other methods of edge smoothing are possible. For example, we may map white pixels which have fewer than, say, three white 8-neighbours to black. This has the effect of

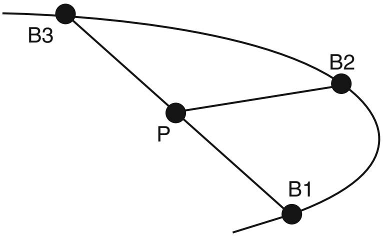


Fig. 14.46
Edge smoothing and corner detection

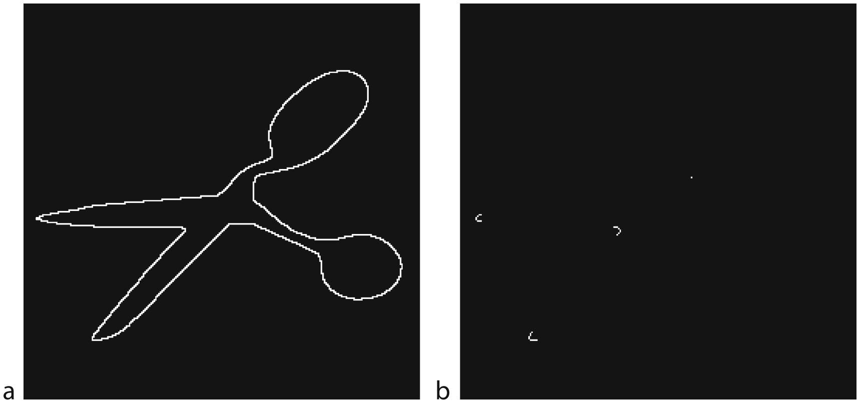


Fig. 14.47
Corner detection. (a) Original image: [bed] applied to ➋ *Fig. 14.40b*. (b) Corners detected [cnr] (c)

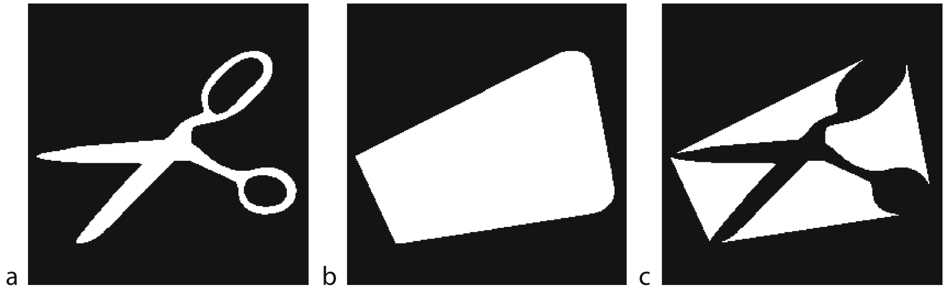


Fig. 14.48
Convex hull. (a) Original image. (b) Convex hull. Notice that this includes both bays and lakes. [*chu*] (c) Bays [*blf, chu, exr, kgr(2000)*] (Very small bays have been removed for clarity.)

eliminating "hair" around the edge of a blob-like figure. One of the techniques described previously for eliminating small spots offers another possibility. A third option is to use the processing sequence: [*exw, skw, skw, exw*], where *exw* represents expand white areas and *skw* denotes shrink white areas.

Convex Hull [chu] Consider a single blob in a binary image. The convex hull is that area enclosed within the smallest convex polygon which will enclose the shape (➋ *Fig. 14.48*). This can also be described as the region enclosed within an elastic string stretched around the blob. The area enclosed by the convex hull but not within the original blob is called the *convex deficiency*, which may consist of a number of disconnected parts and includes any holes and indentations. If we regard the blob as being like an *island*, we can understand the logic of referring to the former as *lakes* and the latter as *bays*.

14.3.1 Measurements on Binary Images

To simplify the following explanation, we will confine ourselves to the analysis of a binary image containing a single blob. The area of the blob can be measured by the total number of object (white) pixels in the image. However, we must first define two different types of edge points, in order to measure an object's perimeter.

The *4-adjacency* convention (➋ *Fig. 14.49*) allows only the four main compass points to be used as direction indicators, while *8-adjacency* uses all eight possible directions. If the 4-adjacency convention is applied to the image segment given in ➋ *Fig. 14.49c*, then none of the four shaded segments (two horizontal and two vertical) will appear as touching, i.e., they are not connected. Using the 8-adjacency convention, these four segments are now connected, but we have the ambiguity that the inside of the shape is connected to the outside. Neither convention is satisfactory, but because 8-adjacency allows diagonally-adjacent pixels to be regarded as connected, it leads to a more faithful perimeter measurement.

Assuming that the 8-adjacency convention is used, we can generate a coded description of the blob's edge. This is referred to as the *chain code* or *Freeman code* [*fcc*]. As we trace around the edge of the blob, we generate a number, 0–7, to indicate which of the eight possible directions we have taken, relative to the centre shaded pixel in ➋ *Fig. 14.49b*. Let N_o indicate how many *odd-numbered* code values are produced as we code the blob's edge, and N_e represent

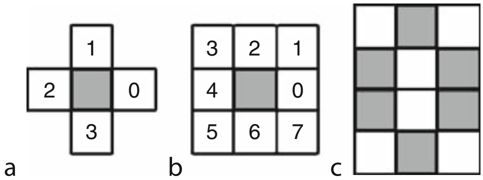


Fig. 14.49
Chain code. (a) 4-adjacency coding convention. (b) 8-adjacency coding convention. (c) Image segment

the number of *even-numbered* values found. The *perimeter* of the blob is given *approximately* by the formula $N_e + 2N_o$.

This formula will normally suffice for use in those situations where the perimeter of a smooth object is to be measured. The *centroid* of a blob [*cgr*] determines its position within the image and can be calculated using the formulae:

$$\bar{I} \leftarrow \sum_j \sum_i (i(a(i,j)))/N_{i,j}$$

$$\bar{J} \leftarrow \sum_j \sum_i (j(a(i,j)))/N_{i,j}$$

$$\text{where} \quad N_{i,j} \leftarrow \sum_j \sum_i a(i,j)$$

Although we are considering images in which the a(i,j) are equal to 0 (black) or 1 (white), it is convenient to use a(i,j) as an ordinary arithmetic variable as well.

14.3.2 Shape Descriptors (➋ *Fig. 14.50*)

The following are just a few of the shape descriptors that have been proposed:

(a) The distance of the furthest point on the edge of the blob from the centroid.
(b) The distance of the closest point on the edge of the blob from the centroid.
(c) The number of protuberances, as defined by that circle whose radius is equal to the average of the parameters measured in (a) and (b). This circle is located at the blob centroid.
(d) Circularity = Area/Perimeter. This will tend to zero for irregular shapes with ragged boundaries, and has a maximum value of $1/4\pi$ for a circle.
(e) The distances of points on the edge of the blob from the centroid, as a function of angular position. This describes the silhouette in terms of polar co-ordinates. (This is not a single-valued function.)
(f) The number of holes. (Use *eul* and [*blf, exr, ndo*] to count them.)
(g) The number of bays.
(h) The ratio of the area of the original blob to that of its convex hull.
(i) The ratio of the area of the original blob to that of its circumcircle.
(j) The ratio of the area of the blob to the square of the total limb-length of its skeleton.
(k) The distances between joints and limb ends of the skeleton.
(l) The ratio of the projections onto the major and minor axes.

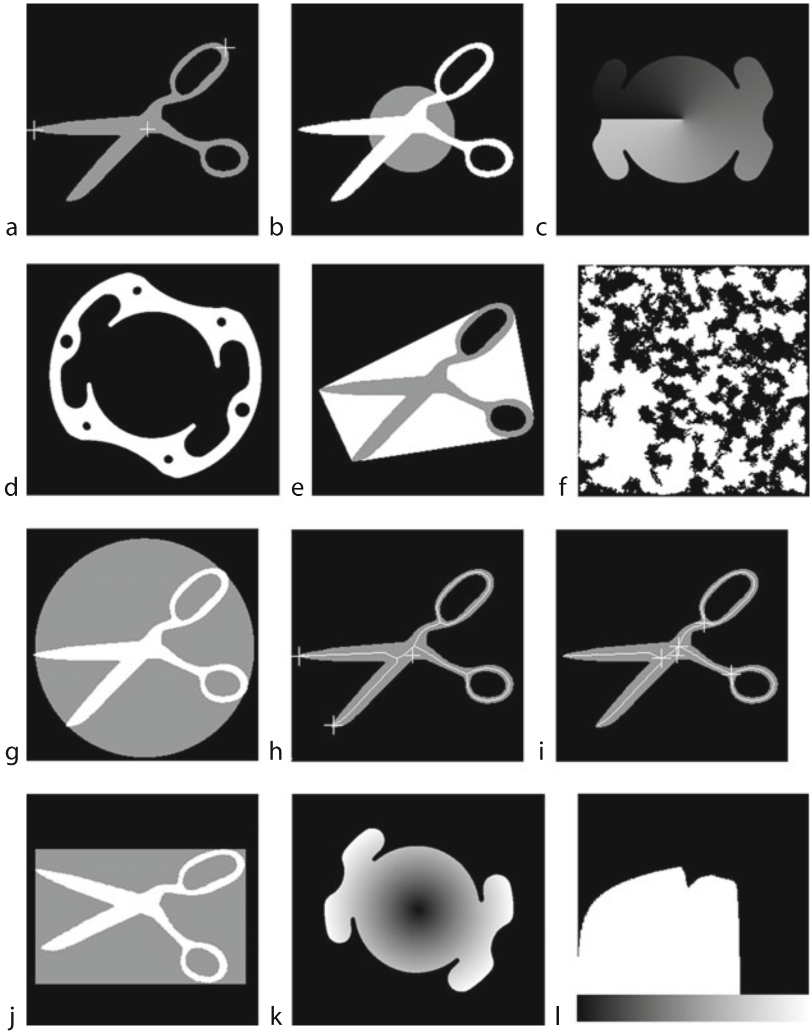


Fig. 14.50

Features and measurements on binary images (also see *Fig. 14.45*). (a) Distance from the centroid (central cross) to the furthest edge points. (b) Counting/ measuring protuberances (regions outside the circle of equivalent area). (c) Angular distribution of "mass" after normalising position and orientation. (This "object" is the lake in (d).) (d) Euler number (−6) provides a way to check the number of lakes. (This object is a stamping for the stator of a small electric motor.) (e) Relative areas of bays and lakes. (f) For very complex images, the Euler number is a rapid way to estimate the number of blobs. The result is approximate, since there may be some lakes formed by the nearly random process that created this image. (g) The ratio of the area of the circumcircle to that of the original blob provides a measure of shape. (h) Skeleton limb ends. (i) Skeleton joints. (j) Minimum enclosing rectangle after normalising orientation allows the object's aspect ratio to be measured. (k) Radial distribution of "mass", measured from the centroid. (l) Histogram of (k) provides a shape description vector that is independent of orientation. Notice that we treat the parts of (d) as objects in their own right

14.4 Binary Mathematical Morphology

The basic concept involved in mathematical morphology is simple: an image is probed with a template shape, called a structuring element, to find where it fits, or does not fit [2] (➲ *Fig. 14.51*). By marking the locations where the template shape fits, structural information can be gleaned about the image. The structuring elements used in practice are usually geometrically simpler than the image they act on, although this is not always the case. Common structuring elements include small groups of isolated points, point pairs, lines, squares, octagons, discs, rhombi, and rings. Since shape is a prime carrier of information in machine vision applications, mathematical morphology has an important role to play in industrial systems [5].

The language of binary morphology is derived from that of set theory [6]. General mathematical morphology is normally discussed in terms of Euclidean *N*-space, but in digital image analysis we are only interested in a discrete or digitised equivalent in two-space. The following analysis is therefore restricted to binary images in a digital two-dimensional integer space, Z2. The image set (or scene) under analysis will be denoted by *A*, with elements *a = (a1, a2)*. The shape parameter, or structuring element, that will be applied to scene *A* will be denoted by *B*, with elements *b = (b1, b2)*. The primary morphological operations that we will examine are dilation, erosion, opening, and closing.

Dilation (also referred to as *filling* and *growing*) is the expansion of an image set *A* by a structuring element *B*. It is a generalisation of the process of expanding white areas described in ➲ Sect. 14.3. Dilation is formally viewed as the combination of the two sets using vector addition of the set elements. The dilation of an image set *A* by a structuring element *B* will be denoted *A* ⊗ *B*, and can be represented as the union of translates of the structuring element *B* [6]:

$$A \oplus B = \bigcup_{a \in B} B_a$$

where ∪ represents the union of a set of points and the translation of *B* by point *a* is given by $B_a = \{c \in Z^2 | c = (b + a)\}$ for some $b \in B$.

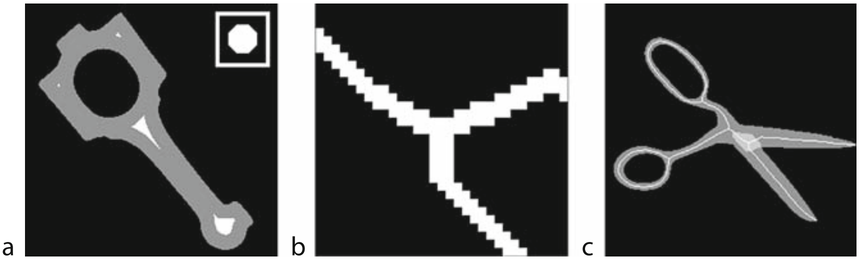


Fig. 14.51

Binary erosion with an arbitrary structuring element. (a) The structuring element (inset) fits within the object (automobile con-rod) at those places indicated by white pixels, which form four blob-like regions. (b) Structuring element, derived from the skeleton of the dark grey object (scissors) in (c). (c) The structuring element fits within the silhouette of the scissors at many points but these are all connected and form a single blob. (light grey) Making the structuring element bigger, for example by including both skeleton joints, will make it more selective

This is best explained by visualising a structuring element B moving over an image A in a raster fashion. Whenever the origin of the structuring element coincides with one of the image pixels in A, then the entire structuring element is placed in the output image at that location. For example, in *Fig. 14.52* the image is dilated by a cross-shaped structuring element contained within a 3×3 pixel grid.

Erosion is the dual morphological operation of dilation and is equivalent to the shrinking (or reduction) of the image set A by a structuring element B. It is a generalisation of shrinking as described in Sect. 14.3. Erosion is a morphological transformation which combines two sets using vector subtraction of set elements [6]. The erosion of an image set A by a structuring element B, denoted $A \ominus B$, can be represented as the intersection of the negative translates:

$$A \ominus B = \bigcap_{\mathrm{b}\in \mathrm{B}} \mathrm{A}_{-\mathrm{b}}$$

where ∩ represents the intersection of a set of points. Erosion of the image A by B is the set of all points for which B translated to a point x is contained in A. This consists of sliding the structuring element B across the image A, and where B is fully contained in A (by placing the origin of the structuring element at the point x) then x belongs to the eroded image $\mathrm{A} \ominus \mathrm{B}$. For example, in *Fig. 14.53* the image is eroded by a cross-shaped structuring element contained within a 3×3 pixel grid.

A duality relationship exists between certain morphological operators, such as erosion and dilation. This means that the equivalent of such an operation can be performed by its dual on the complement (negative) image and by taking the complement of the result [5].

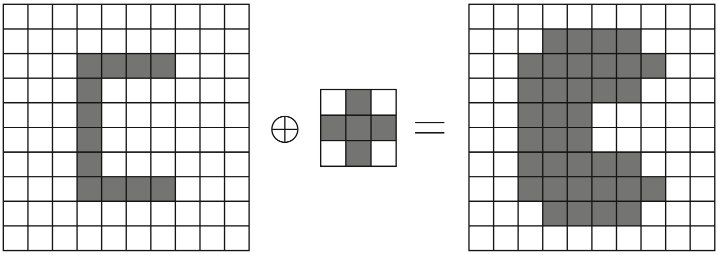


Fig. 14.52
Dilation of an image by a "cross" structuring element

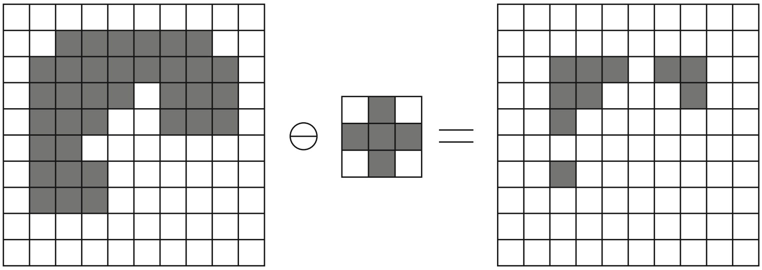


Fig. 14.53
Erosion of an image by a cross-shaped structuring element

Although duals, the erosion and dilation operations are not inverses of each other. Rather, they are related by the following duality relationships:

$$(A \ominus B)^C = A^C \oplus \bar{B} \qquad (A \oplus B)^C = A^C \ominus \bar{B}$$

where A^c refers to the complement of the image set *A* and

$$\bar{B} = \{x| \text{ for some } b \in B,\ x = -b\}$$

refers to the reflection of B about the origin. ([9] refers to this as the *transpose* of the structuring element.)

14.4.1 Opening and Closing Operations

Erosion and dilation tend to be used in pairs to extract or impose structure on an image. The most commonly found erosion-dilation pairings occur in the *opening* and *closing* transformations.

Opening is a combination of erosion and dilation operations that have the effect of removing isolated spots in the image set *A* that are smaller than the structuring element *B* and those sections of the image set *A* narrower than *B*. This is also viewed as a geometric *rounding* operation (➲ *Fig. 14.54*). The opening of the image set *A* by the structuring element B, is denoted $A \circ B$, and is defined as $(A \ominus B) \oplus B$.

Closing [cls] is the dual morphological operation of opening. This transformation has the effect of filling in holes and blocking narrow valleys in the image set *A*, when a structuring element *B* (of similar size to the holes and valleys) is applied. The closing of the image set *A* by the structuring element B is denoted $A \bullet B$, and is defined as $(A \oplus B) \ominus B$.

One important property that is shared by both the opening and closing operations is *idempotency*. This means that successful reapplication of the operations will not change the previously transformed image [5]. Therefore, $A \circ B = (A \circ B) \circ B$ and $A \bullet B = (A \bullet B) \bullet B$.

Unfortunately, the application of morphological techniques to industrial tasks, which involves complex operations on "real-world" images, can be difficult to implement. Practical imaging applications tend to have structuring elements that are unpredictable in shape and size. In practice, the ability to manipulate arbitrary structuring elements usually relies on their decomposition into component parts.

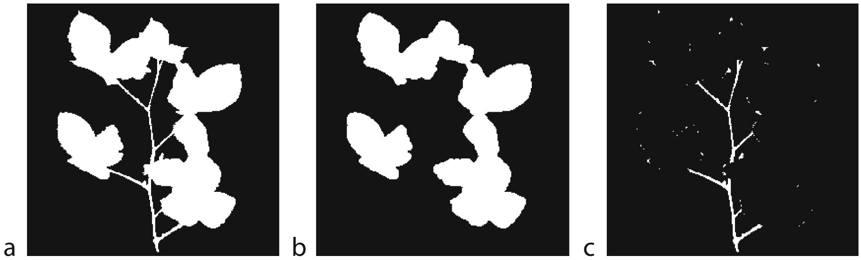


Fig. 14.54
Binary opening. (a) Original image (small plant). (b) Opening. [*ero(5), dil(5)*] (c) Exclusive OR of (a) and (b)

14.4.2 Structuring Element Decomposition

Some vision systems [3, 11, 12] can perform basic morphological operations very quickly in a parallel and/or pipelined manner. Implementations that involve such special purpose hardware tend to be expensive, although there are some notable exceptions [13]. Unfortunately, some of these systems impose restrictions on the shape and size of the structuring elements that can be handled. Therefore, one of the key problems involved in the application of morphological techniques to industrial image analysis is the generation and/or decomposition of large structuring elements. Two main strategies are used to tackle this problem.

The first technique is called *dilation* or *serial decomposition*. This decomposes certain large structuring elements into a sequence of successive erosion and dilation operations, each step operating on the preceding result. Unfortunately, the serial decomposition of large structuring elements into smaller ones is not always possible. Furthermore, those decompositions that are possible are not always easy to identify and implement.

If a large structuring element *B* can be decomposed into a chain of dilation operations, $B = B_1 \oplus B_2 \oplus \cdots \oplus B_N$ (➔ *Fig. 14.55*), then the dilation of the image set *A* by *B* is given by

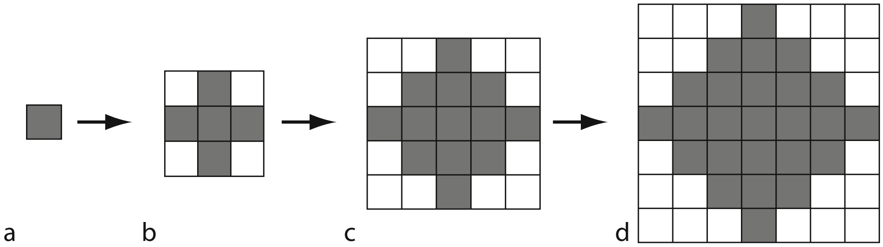
$$A \oplus B = A \oplus (B_1 \oplus B_2 \oplus \cdots \oplus B_N) = (((A \oplus B_1) \oplus B_2) \cdots) \oplus B_N.$$

Similarly, using the so-called chain rule [16], which states that $A \ominus (B \oplus C) = (A \ominus B) \ominus C$, the erosion of *A* by *B* is given by

$$A \ominus\ B = A \ominus\ (B_1 \oplus B_2 \oplus \cdots \oplus B_N) = (((A \ominus B_1) \ominus B_2) \cdots) \ominus B_N.$$

A second approach to the decomposition problem is based on "breaking up" the structuring element, B, into a union of smaller components, $B_1,\ldots, B_N$. We can think of this approach as "tiling" of the structuring element by sub-structuring elements (➔ *Fig. 14.56*). Since the "tiles" do not need to be contiguous or aligned, any shape can be specified without the need for serial decomposition of the structuring element, although the computational cost of this approach is proportional to the area of the structuring element [12]. This is referred to as *union* or *parallel decomposition*. Therefore, with B decomposed into a union of smaller structuring elements, $B = B_1 \cup B_2 \cup \cdots \cup B_N$, then the dilation of an image A by the structuring element B can be rewritten as

$$A \oplus B = A \oplus (B_1 \cup B_2 \cup \cdots \cup B_N) = (A \oplus B_1) \cup (A \oplus B_2) \cup \cdots \cup (A \oplus B_N).$$



■ **Fig. 14.55**

Construction of a 7×7 structuring element by successive dilation of a 3×3 structuring element. (a) Initial pixel. (b) 3×3 structuring element and the result of the first dilation. (c) Result of the second dilation. (d) Result of the third dilation [12]

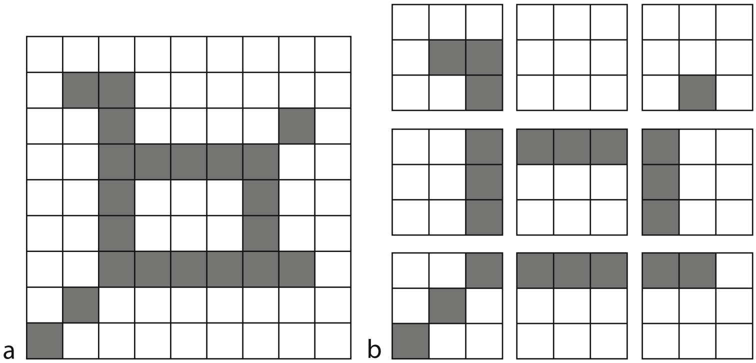


Fig. 14.56
Tiling of a 9×9 arbitrary structuring element. (a) The initial 9×9 structuring element. (b) Tiling with nine 3×3 sub-structuring elements [12]

Likewise, the erosion of *A* by the structuring element *B* can be rewritten as

$$A \ominus B = A \ominus (B_1 \cup B_2 \cup \cdots \cup B_N) = (A \ominus B_1) \cap (A \ominus B_2) \cap \cdots \cap (A \ominus B_N).$$

This makes use of the fact that $A \ominus (B \cup C) = (A \ominus B) \cap (A \ominus C)$ [5]. Due to the nature of this decomposition procedure, it is well suited to implementation on parallel computer architectures.

Waltz [12] compared these structural element decomposition techniques, and showed that the serial approach has a 9:4 speed advantage over its parallel equivalent. (This was based on an arbitrarily specified 9×9 pixel structuring element, when implemented on a commercially available vision system.) However, the parallel approach has a 9:4 advantage in the number of degrees of freedom. (Every possible 9×9 structuring element can be achieved with the *parallel decomposition*, but only a small subset can be realised with the serial approach.) Although slower than the serial approach, it has the advantage that there is no need for serial decomposition of the structuring element.

Classical parallel and serial methods mainly involve the numerous scanning of image pixels and are therefore inefficient when implemented on conventional computers. This is so because the number of scans depends on the total number of pixels (or edge pixels) in the shape to be processed by the morphological operator. Although the parallel approach is suited to some customised (parallel) architectures, the ability to implement such parallel approaches on *serial* machines is discussed by Vincent [14].

14.5 Grey Scale Morphology

Binary morphological operations can be extended naturally to process grey scale imagery by the use of neighbourhood minimum and maximum functions [5]. Heijmans [7] presents a detailed study of grey scale morphological operators, in which he outlines how binary morphological operators and thresholding techniques can be used to build a large class of useful grey scale morphological operators. Sternberg [11], discusses the application of such morphological techniques to industrial inspection tasks.

In ➋ *Fig. 14.57*, a one-dimensional morphological filter operates on an analogue signal (equivalent to a grey scale image). The input signal is represented by the thin curve and the output by the thick black curve. In this simple example, the structuring element has an approximately parabolic form. In order to calculate a value for the output signal, the structuring element is pushed upwards from below the input curve. The height of the top of the structuring element is noted. This process is then repeated by sliding the structuring element sideways. Notice how this particular operator attenuates the intensity peak but follows the input signal quite accurately everywhere else. Subtracting the output signal from the input would produce a result in which the intensity peak is emphasised and all other variations would be reduced.

The effect of the basic morphological operators on two-dimensional grey scale images can also be explained in these terms. Imagine the grey scale image as a landscape, in which each pixel can be viewed in 3-D. The extra height dimension represents the grey scale value of a pixel. We generate new images by passing the structuring element above/below this landscape (see ➋ *Fig. 14.57*).

Grey Scale Dilation This is computed as the maximum of translations of the grey surface. Grey level dilation of image *A* by the structuring element *B* produces an image *C* defined by

$$\mathrm{C(r,c) = MAX_{(i,j)}\{A(r-i,c-j)+B(i,j)\} = (A \oplus B)(r,c)}$$

where *A*, *B* and *C* are grey level images. Commonly used grey level structuring elements include rods, disks, cones and hemispheres. This operation is often used to smooth small negative-contrast grey level regions in an image. *[lnb]* is an example of a dilation operator (➋ *Fig. 14.22*).

Grey Scale Erosion The grey value of the erosion at any point is the *maximum* value for which the structuring element centred at that point, still fits entirely within the foreground under the surface. This is computed by taking the *minimum* of the grey surface translated by all the points of the structuring element. Grey level erosion of image *A* by the structuring element *B* produces an image *C* defined by

$$\mathrm{C(r,c) = MIN_{(i,j)}\{A(r+i,c+j)-B(i,j)\} = (A\emptyset}B\mathrm{)(r,c)}$$

This operation is commonly used to smooth small positive-contrast grey level regions in an image. [*snb*] is an example of an erosion operator.

Grey Scale Opening This operation is defined as the grey level erosion of the image followed by the grey level dilation of the eroded image. That is, it will cut down the peaks in the grey level topography to the highest level for which the elements fit under the surface.

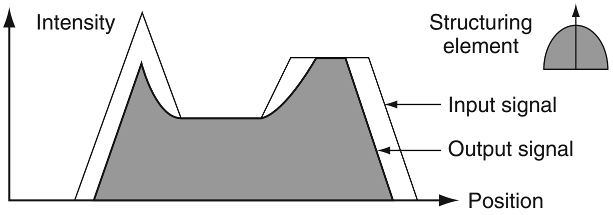


Fig. 14.57
A one-dimensional morphological filter eroding an analogue signal

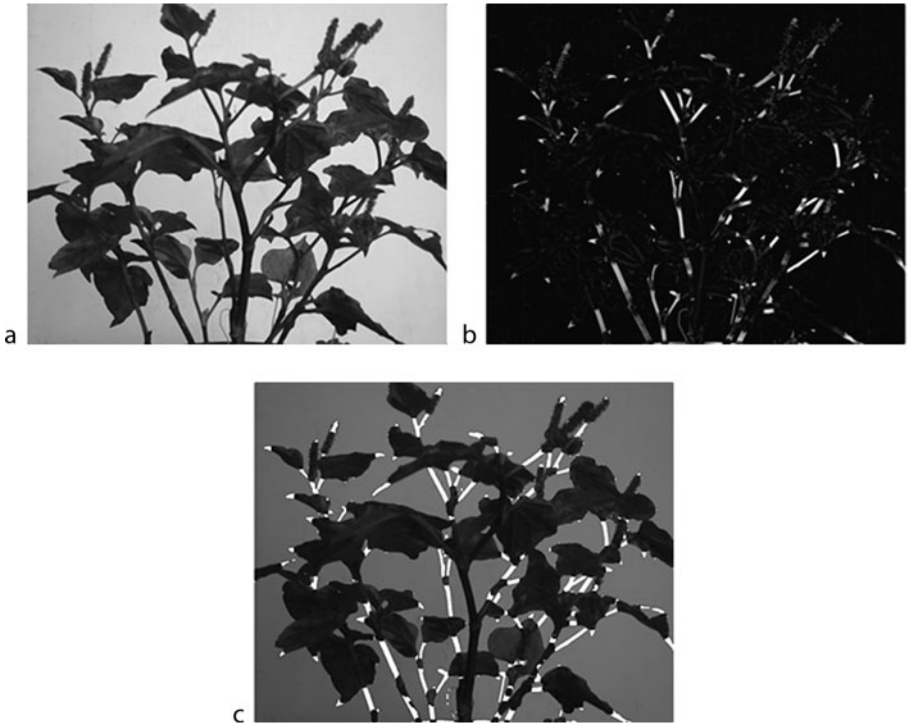


Fig. 14.58

Grey-scale closing. (a) Original image. (b) Result of closing. The structuring element was a disc of uniform height and diameter 13 pixels. [*crk(7)*] (c) Image in (b) thresholded and superimposed on the original. Notice that the operator is highly selective for size and detects thin "linear" features (stems) and the triangular tips of large wedge-like objects (leaves)

Grey Scale Closing (*Fig. 14.58*) This operation is defined as the grey level dilation of the image followed by the grey level erosion of the dilated image. Closing fills in the valleys to the maximum level for which the element fails to fit above the surface. For a more detailed discussion on binary and grey scale mathematical morphology, see Haralick and Shapiro [6] and Dougherty [2]. The crack detector [*crk/1*] is an example of a closing operator (*Fig. 14.25*).

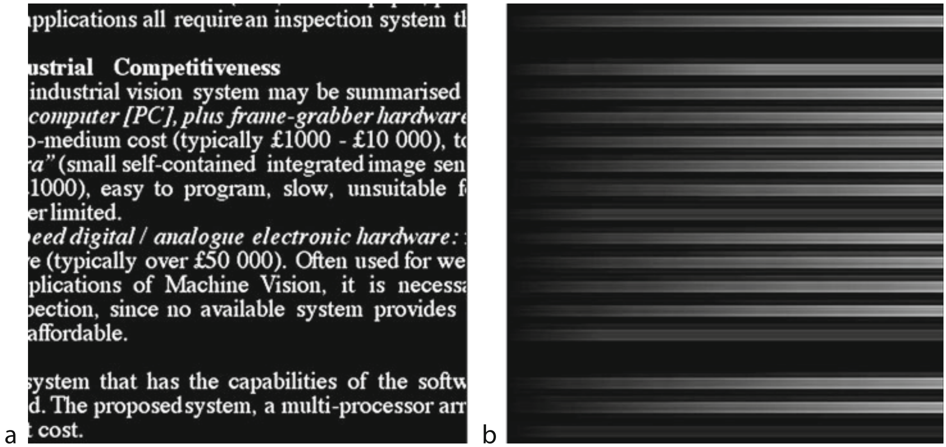
14.6 Global Image Transforms

An important class of image processing operators is characterised by an equation of the form **B** ⇐ f(**A**), where **A** = {a(i,j)} and **B** = {b(p,q)}. Each element in the output picture **B** is calculated using all or at least a large proportion of the pixels in **A**. The output image **B** may well look quite different from the input image **A**. Examples of this class of operators are *lateral shift, rotation, warping, Cartesian-to-Polar co-ordinate conversion, Fourier transform*, and *Hough transform.*

Integrate Intensities Along Image Rows (➲ *Fig. 14.59*) *[rin]* This operator is rarely of great value when used on its own, but can be used with other operators to good effect, for example detecting horizontal streaks and edges. The operator is defined recursively:

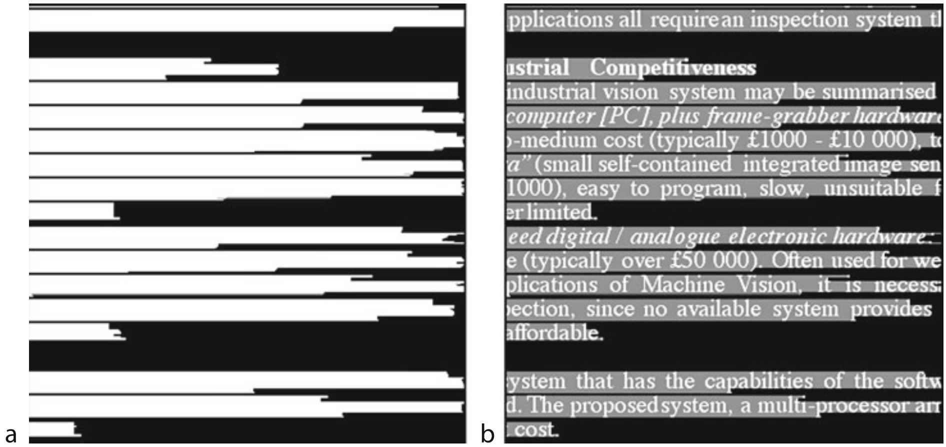
$$b(i,j) \Leftarrow b(i,j-1) + (a(i,j)/n) \text{ where } b(0,0) = 0.$$

Row Maximum (➲ *Fig. 14.60*) *[rox]* This function is often used to detect local intensity minima.



pplications all require an inspection system tl
ustrial Competitiveness
industrial vision system may be summarised
computer [PC], plus frame-grabber hardware
)-medium cost (typically £1000 - £10 000), to
a" (small self-contained integrated image sen
1000), easy to program, slow, unsuitable f
er limited.
eed digital / analogue electronic hardware:
e (typically over £50 000). Often used for we
plications of Machine Vision, it is necess
pection, since no available system provides
affordable.
system that has the capabilities of the softw
d. The proposed system, a multi-processor arr
t cost.

■ Fig. 14.59
Row integration. (a) Original image. (b) Result of processing [*rin*]



pplications all require an inspection system tl
ustrial Competitiveness
industrial vision system may be summarised
computer [PC], plus frame-grabber hardware
)-medium cost (typically £1000 - £10 000), to
a" (small self-contained integrated image sen
1000), easy to program, slow, unsuitable f
er limited.
eed digital / analogue electronic hardware:
e (typically over £50 000). Often used for we
plications of Machine Vision, it is necess
pection, since no available system provides
affordable.
ystem that has the capabilities of the softw
d. The proposed system, a multi-processor arr
t cost.

■ Fig. 14.60
Row maximum. (a) Row maximum, running from right to left. [*lrt, rox, lrt*] (b) Original superimposed on (a)

$$c(i,j) \Leftarrow \mathrm{MAX}(a(i,j), c(i,j-1))$$

Geometric Transforms Algorithms exist by which images can be shifted [*psh* and *psw*], rotated [*tur*], magnified [*pex* and *psq*] and warped (Figs. 14.61 and 14.62) [*ctp* and *ptc*]. The reader should note that certain operations, such as rotating a digital image, can cause some difficulties because pixels in the input image are not mapped exactly to pixels in the output image. This can cause smooth edges to appear stepped. To avoid this effect, interpolation may be used, but this has the unfortunate effect of blurring edges. (See [BAT91a] for more details.)

The utility of axis transformations is evident when we are confronted with the examination of circular objects, or those displaying a series of concentric arcs, or streaks radiating from a fixed point. Inspecting such objects is often made very much easier, if we first convert from

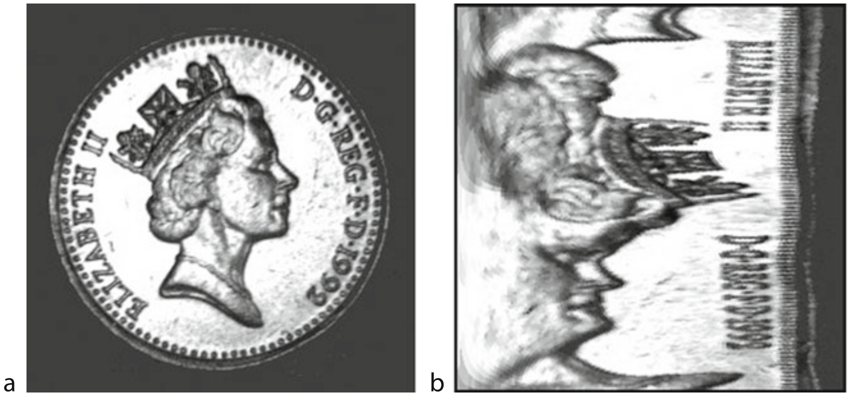


Fig. 14.61
Cartesian-to-polar coordinate transformation. (a) Original image. (b) Result of processing [*ctp*]

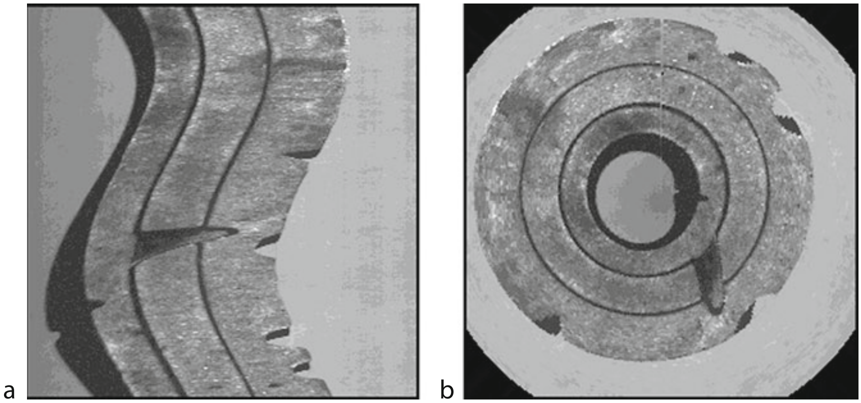


Fig. 14.62
Polar-to-Cartesian coordinate transformation. (This is a toroidal object and was scanned with a line-scan sensor, on a turn-table.) (a) Original image. (b) Result of processing [*ptc*] (The widget looks just like this!)

Cartesian to *Polar* co-ordinates (➲ *Fig. 14.61*). Warping is also useful in a variety of situations. For example, it is possible to compensate for *barrel* or *pin-cushion* distortion in a camera. Geometric distortions introduced by a wide-angle lens, or trapezoidal distortion due to viewing the scene from an oblique angle, can also be corrected. Another possibility is to convert simple curves of known shape into straight lines, in order to make subsequent analysis easier.

14.6.1 Hough Transform

The Hough transform provides a powerful and robust technique for detecting lines, circles, ellipses, parabolas, and other curves of pre-defined shape, in a binary image. Let us begin our discussion of this fascinating topic by describing the simplest version, the basic Hough Transform, which is intended to detect straight lines. Actually, our objective is to locate *nearly linear* arrangements of disconnected white spots and "*broken*" *lines.* Consider that a straight line in the input image is defined by the equation $r = x\ \mathrm{Cos}\ \phi + y\ \mathrm{Sin}\ \phi$, where r and ϕ are two unknown parameters whose values are to be found. Clearly, if this line intersects the point (x_i, y_i), then $r = x_i\ \mathrm{Cos}\ \phi + y\ \mathrm{Sin}\ \phi$ can be solved for many different values of (r, ϕ). So, each white point (x_i, y_i) in the input image may be associated with a *set* of (r, ϕ) values. Actually, this set of points forms a *sinusoidal curve* in (r, ϕ) space. The latter is called the *Hough Transform (HT)* image. Since each point in the input image generates such a sinusoidal curve, the whole of that image creates a multitude of overlapping sinusoids in the resulting HT image. In many instances, a large number of sinusoidal curves are found to converge on the same spot in the HT image. The (r, ϕ) address of such a point indicates the slope ϕ and position r of a straight line that can be drawn through a large number of white spots in the input image.

The implementation of the Hough transform for line detection begins by using a two-dimensional accumulator array, $A(r, \phi)$, to represent quantised (r, ϕ) space. (Clearly, an important choice to be made is the step size for quantising r and ϕ. However, we will not dwell on such details here.) Assuming that all the elements of $A(r, \phi)$ are initialised to zero, the Hough Transform is found by computing a set $S(x_i, y_i)$ of (r, ϕ) pairs satisfying the equation $r = x_i\ \mathrm{Cos}\phi + y_i\ \mathrm{Sin}\ \phi$. Then, for all (r, ϕ) in $S(x_i, y_i)$, we increment $A(r, \phi)$ by one. This process is then repeated for all values of i such that the point (x_i, y_i) in the input image is white. We repeat that bright spots in the HT image indicate "linear" sets of spots in the input image. Thus, line detection is transformed to the rather simpler task of finding local maxima in the accumulator array $A(r, \phi)$. The co-ordinates (r, ϕ) of such a local maximum give the parameters of the equation of the corresponding line in the input image. The HT image can be displayed, processed, and analysed just like any other image, using the operators that are now familiar to us.

The robustness of the HT techniques arises from the fact that, if part of the line is missing, the corresponding peak in the HT image is simply not quite so bright. This occurs because fewer sinusoidal curves converge on that spot and the corresponding accumulator cell is incremented less often. However, unless the line is almost completely obliterated, this new darker spot can also be detected. In practice, we find that "nearly straight lines" are transformed into a *cluster* of points, with a spreading of the intensity peaks in the HT image, due to noise and quantisation effects. In this event, to calculate the parameters of the straight line in the input image we threshold the HT image and then find the centroid of the resulting spot. Pitas [8] gives a more detailed description of this algorithm. ➲ *Figure 14.63* illustrates how this approach can be used to find a line in a noisy binary image.

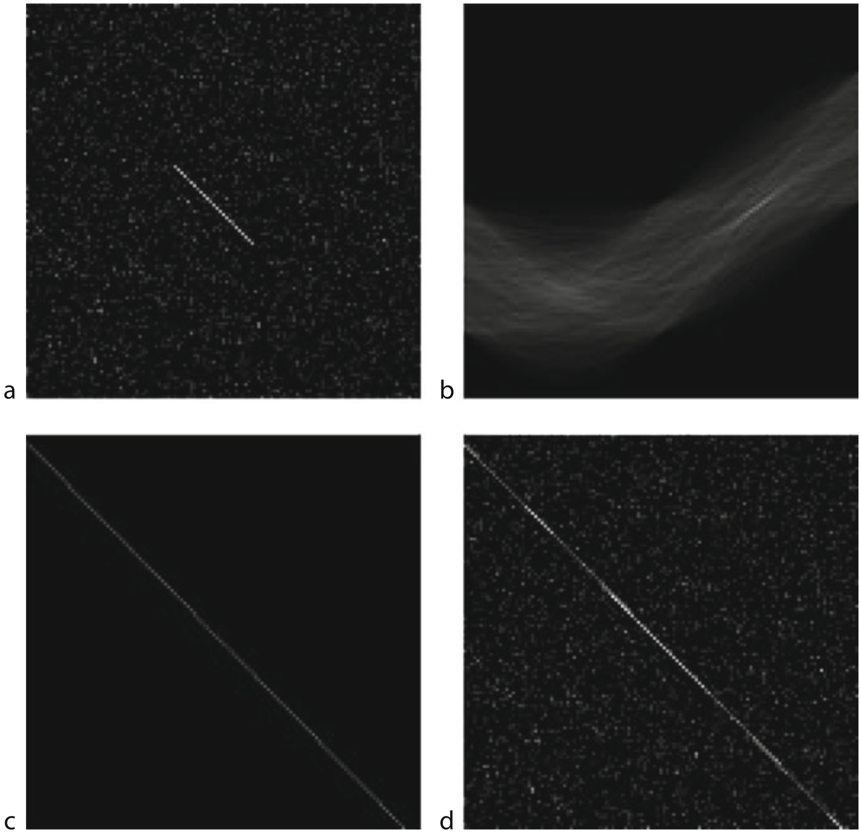


Fig. 14.63

Hough transform. (a) Original image. (b) Hough transform. (c) Inverse Hough applied to a single white pixel located at the point of maximum intensity in (b). (d) Inverse transform superimposed on original image. Notice how accurately this process locates the line in the input image, despite the presence of a high level of noise

The Hough transform can also be generalised to detect groups of points lying on a curve. In practice, this may not be a trivial task, since the complexity increases very rapidly with the number of parameters needed to define the curve. For circle detection, we define a circle parametrically as: $r^2 = (x - a)^2 + (y - b)^2$ where (a, b) determines the co-ordinates of the centre of the circle and r is its radius. This requires a *three-dimensional* parameter space, which, of course, cannot be represented and processed as a single image. For an arbitrary curve, with no simple equation to describe its boundary, a look-up table is used to define the relationship between the boundary coordinates, an orientation, and the Hough transform parameters. (See [10] for more details.)

14.6.2 Two-Dimensional Discrete Fourier Transform

We have just seen how the transformation of an image into a different domain can sometimes make the analysis task easier. Another important operation to which this remark applies is the

Fourier Transform. Since we are discussing the processing of images, we will discuss the *two-dimensional Discrete Fourier Transform (DFT)*. This operation allows spatial periodicities in the intensity within an image to be investigated, in order to find, among other features, the dominant frequencies. The two-dimensional DFT of an N×N image f(x,y) is defined as follows:

$$F(u,v) = \frac{1}{N} \cdot \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \exp[-i2\pi(ux+vy)/N] \quad \text{where } 0 \leq u,v \leq N-1.$$

The inverse transform of F(u,v) is defined as

$$f(x,y) = \frac{1}{N} \cdot \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u,v)) \exp[i2\pi(ux+vy)/N] \quad \text{where } 0 \leq x,y \leq N-1.$$

Several algorithms have been developed to calculate the two-dimensional DFT. The simplest makes use of the fact that this is a *separable* transform, which therefore can be computed as a sequence of two one-dimensional transforms. Thus, we can generate the two-dimensional transform by calculating the one-dimensional Discrete Fourier Transform along the image rows and then repeating this on the resulting image but, this time, operating on the columns. This reduces the computational overhead when compared to direct two-dimensional implementations. The sequence of operations is as follows:

$$f(x,y) \rightarrow \text{Row Transform} \rightarrow F_1(x,v) \rightarrow \text{Column Transform} \rightarrow F_2(u,v)$$

Although this is still computationally slow compared to many other shape measurements, the Fourier transform is quite powerful. It allows the input to be represented in the frequency domain, which can be displayed as a *pair* of images. (It is not possible to represent both amplitude and phase using a single monochrome image.) Once the frequency domain processing is complete, the inverse transform can be used to generate a new image in the original, so-called, *spatial* domain.

The Fourier power spectrum, or amplitude spectrum, plays an important role in image processing and analysis. This can be displayed, processed, and analysed as an intensity image. Since the Fourier transform of a real function produces a complex function:

$$F(u,v) = R(u,v) + iI(u,v)$$

the frequency spectrum of the image is the magnitude function

$$|F(u,v)| = \sqrt{[R^2(u,v) + I^2(u,v)]}$$

and the power spectrum (spectral density) is defined as

$$P(u,v) = |F(u,v)|^2.$$

➋ *Figure 14.64* illustrates how certain textured features can be highlighted using the two-dimensional Discrete Fourier Transform. The image is transformed into the frequency domain and an ideal band-pass filter (with a circular symmetry) is applied. This has the effect of limiting the frequency information in the image. When the inverse transform is calculated, the resultant textured image has a different frequency content, which can then be analysed. For more details on the Fourier transform and its implementations, see [8].

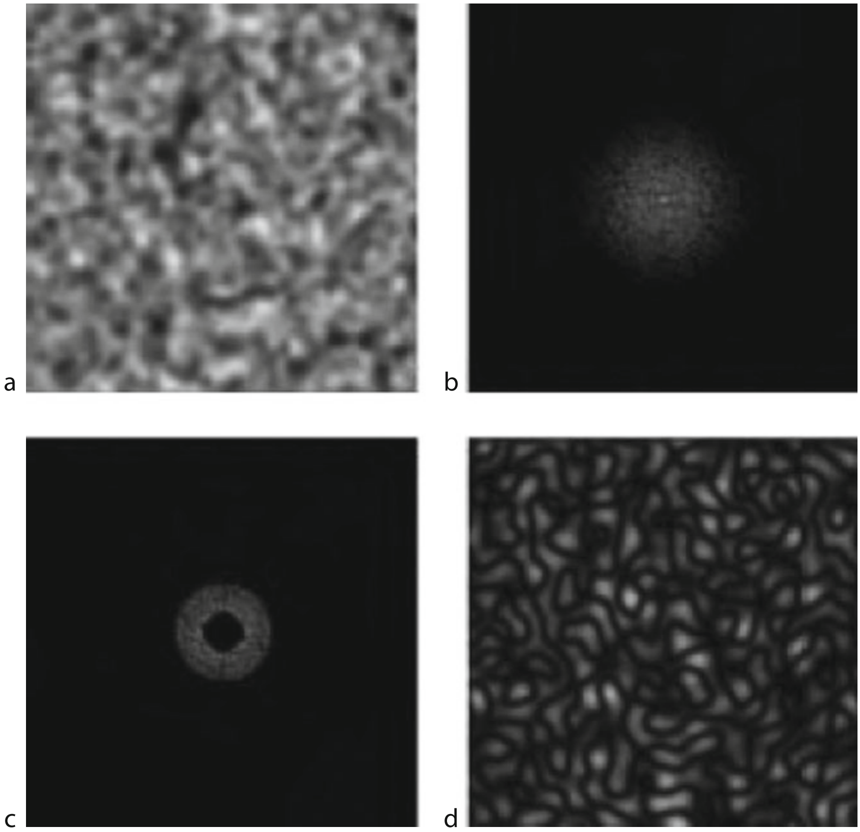


Fig. 14.64
Filtering a textured image in the frequency domain. (a) Original textured image. (b) Resultant transformed image after using the two-dimensional DFT. (The image is the frequency spectrum shown as an intensity function.) (c) Resultant frequency domain image after an ideal band-pass filter is applied to image. (d) The resultant spatial domain image after the inverse two-dimensional discrete Fourier transform is applied to the band-pass filtered image in (c)

14.7 Texture Analysis

Texture is observed in the patterns of a wide variety of synthetic and natural surfaces (e.g., wood, metal, paint and textiles). If an area of a textured image has a large intensity variation, then the dominant feature of that area would be *texture*. If this area has little variation in intensity, then the dominant feature within the area is *tone*. This is known as the *tone-texture concept*. Although a precise formal definition of texture does not exist, it may be described subjectively using terms such as *coarse, fine, smooth, granulated, rippled, regular, irregular*, and *linear*. These features are used extensively in manual region segmentation. There are two main classification techniques for texture: *statistical* and *structural*.

14.7.1 Statistical Approaches

The statistical approach is well suited to the analysis and classification of random or natural textures. A number of different techniques have been developed to describe and analyse such textures [4], a few of which are outlined below.

14.7.1.1 Auto-Correlation Function (ACF)

Auto-correlation derives information about the basic 2-dimensional tonal pattern that is repeated to yield a given periodic texture. The auto-correlation function A(δx,δy) of an image matrix [I(i,j)] is defined as follows:

$$A(\delta x, \delta y) = \left(\sum_{i,j} [(I(i,j))(I(i+\delta x, j+\delta y))]\right) \Big/ \left(\sum_{i,j} [I(i,j)]^2\right)$$

Although useful at times, the ACF has severe limitations. It cannot always distinguish between textures, since many subjectively-different textures have the same ACF. The variables (i, j) are restricted to lie within a specified window outside which the intensity is zero. Incremental shifts of the image are given by (δx,δy). It is worth noting that the ACF and the power spectral density are Fourier transforms of each other.

14.7.1.2 Fourier Spectral Analysis

The Fourier spectrum is well suited to describing the directionality and period of repeated texture patterns, since they give rise to high energy narrow peaks in the power spectrum (see ➲ Sect. 14.6.2 and ➲ *Fig. 14.64*). Typical Fourier descriptors of the power spectrum include *the location of the highest peak, mean, variance*, and *the difference in frequency between the mean and the highest value of the spectrum*. This approach to texture analysis is often used in aerial/ satellite and medical image analysis. The main disadvantage of this approach is that the procedures are not invariant, even under monotonic transforms of its intensity.

14.7.1.3 Edge Density

This is a simple technique in which an edge detector or high pass filter is applied to the textured image. The result is then thresholded and the edge density is measured by the average number of edge pixels per unit area. Two-dimensional or directional filters/edge detectors may be used, as appropriate.

14.7.1.4 Histogram Features

This useful approach to texture analysis is based on the intensity histogram of all or part of an image. Common histogram features include *moments, entropy dispersion, mean* (an estimate of the average intensity level), *variance* (this second moment is a measure of the dispersion of the region intensity), *mean square value* or *average energy, skewness* (the third moment which gives an indication of the histograms symmetry), and *kurtosis* (cluster prominence or "peakness"). For example, a narrow histogram indicates a low contrast region, while two peaks with a well-defined valley between them indicates a region that can readily be separated by simple thresholding.

Texture analysis based solely on the grey scale histogram suffers from the limitation that it provides no information about the *relative position* of pixels to each other. Consider two binary images, where each image has 50% black and 50% white pixels. One of the images might be a checkerboard pattern, while the second one may consist of a salt and pepper noise pattern. These images generate exactly the same grey level histogram. Therefore, we cannot distinguish them using first order (histogram) statistics alone. This leads us naturally to the examination of the co-occurrence approach to texture measurement.

14.7.2 Co-occurrence Matrix Approach

The co-occurrence matrix technique is based on the study of *second-order grey level spatial dependency statistics*. This involves the study of the grey level spatial interdependence of pixels and their spatial distribution in a local area. Second order statistics describe the way grey levels tend to occur together, in pairs, and therefore provide a description of the type of texture present. A *two-dimensional* histogram of the spatial dependency of the various grey level picture elements within a textured image is created. While this technique is quite powerful, it does not describe the shape of the primitive patterns making up the given texture.

The co-occurrence matrix is based on the estimation of the second order joint conditional probability density function, *f(p,q,d,a)*. This is the probability of going from one pixel with grey level *p* to another with grey level *q*, given that the distance between them is *d* and the direction of travel between them is given by the angle *a*. (For an image with N_g grey levels, the size of the co-occurrence matrix will be $N_g \times N_g$.) For example, assuming the intensity distribution shown in the sub-image given below, we can generate the co-occurrence matrix for *d = 1* and *a* is taken as 0 degrees.

2	3	3	3
1	1	0	0
1	1	0	0
0	0	2	2
2	2	3	3

Sub-image with 4 grey-levels.

Grey scale	0	1	2	3
0	6	2	1	0
1	2	4	0	0
2	1	0	4	2
3	0	0	2	6

Co-occurrence matrix {f(p,q,1,0)} for the sub-image.

A co-occurrence distribution that changes rapidly with distance *d* indicates a fine texture. Since the co-occurrence matrix also depends on the image intensity range, it is common practice to normalise the grey scale of the textured image prior to generating the co-occurrence matrix. This ensures that first-order statistics have standard values and avoids confusing the effects of first- and second-order statistics of the image.

A number of texture measures (also referred to as texture attributes) have been developed to describe the co-occurrence matrix numerically and allow meaningful comparisons between various textures [4] (see Fig. 14.65). Although these attributes are computationally intensive, they are simple to implement. Some sample texture attributes for the co-occurrence matrix are given below.

Energy, or angular second moment, is a measure of the homogeneity of a texture. It is defined thus:

$$\text{Energy} = \sum_p \sum_q [f(p, q, d, a)]^2$$

In a uniform image, the co-occurrence matrix will have few entries of large magnitude. In this case the *Energy* attribute will be large.

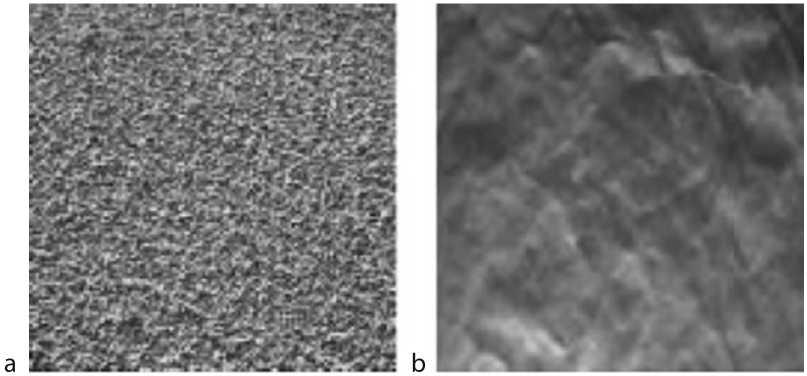
Entropy is a measure of the complexity of a texture and is defined thus:

$$\text{Entropy} = -\sum_p \sum_q [f(p, q, d, a) \log(f(p, q, d, a))]$$

It is commonly found that what a person judges to be a complex image tends to have a higher *Entropy* value than a simple one.

Inertia is the measurement of the moment of inertia of the co-occurrence matrix about its main diagonal. This is also referred as the *contrast* of the textured image. This attribute gives an indication of the amount of local variation of intensity present in an image.

$$\text{Inertia} = \sum_p \sum_q \left[(p - q)^2 f(p, q, d, a)\right]$$



c

	Sand	Sand	Paper	Paper
	f(p,q,1,0)	f(p,q,1,90)	f(p,q,1,0)	f(p,q,1,90)
Energy ($\times 10^6$)	1.63	1.7	3.49	3.42
Inertia ($\times 10^8$)	5.4	6.5	0.181	0.304

Fig. 14.65
Co-occurrence based texture analysis. (a) Sand texture. (b) Paper texture. (c) Texture attributes

14.7.3 Structural Approaches

Certain textures are deterministic in that they consist of identical *texels* (basic texture element), which are placed in a repeating pattern according to some well-defined but unknown placement rules. To begin the analysis, a texel is isolated by identifying a group of pixels having certain invariant properties, which repeat in the given image. A texel may be defined by its *grey level, shape*, or *homogeneity of some local property*, such as size or orientation. Texel spatial relationships may be expressed in terms of *adjacency, closest distance*, and *periodicities.*

This approach has a similarity to language; with both image elements and grammar, we can generate a syntactic model. A texture is labelled *strong* if it is defined by deterministic placement rules, while a *weak* texture is one in which the texels are placed at random. Measures for placement rules include *edge density, run lengths of maximally connected pixels*, and the *number of pixels per unit area* showing grey levels that are locally maxima or minima relative to their neighbours.

14.7.4 Morphological Texture Analysis

Textural properties can be obtained from the erosion process (➋ Sects. 14.4 and ➋ 14.5) by appropriately parameterising the structuring element and determining the number of elements of the erosion as a function of the parameters value [2]. The number of white pixels of the morphological opening operation as a function of the size parameter of the structuring element **H** can determine the size distribution of the *grains* in an image. Granularity of the image **F** is defined as

$$G(d) = 1 - (\#[F \bigcirc H_d]/\#F)$$

where $\mathbf{H}_d$ is a disk structuring element of diameter *d* or a line structuring element of length *d*, and #F is the number of elements in **F**. This measures the proportion of pixels participating in grains smaller than *d*.

14.8 Further Remarks

The image processing operators described in this chapter have all found widespread use in industrial vision systems. It should be understood that they are always used in combination with one another. Other areas of application for image processing may well use additional algorithms to good effect. Very often, these can be formulated in terms of the procedures described above, and implemented as a sequence of QT commands. Two key features of industrial image processing systems are the cost and speed of the target system (i.e., the one installed in a factory). It is common practice to use a more versatile and slower system for problem analysis and prototyping, whereas the target system must continue to operate in an extremely hostile environment. (It may be hot, greasy, wet, and/or dusty.) It must also be tolerant of abuse and neglect. As far as possible, the target system should be self-calibrating and able to verify that it is “seeing” appropriate images. It should provide enough information to ensure that the factory personnel are able to trust it; no machine system should be built that is a viewed by the workers as a mysterious black box. Consideration of these factors is as much a part of the design process as writing the image processing software.

References

1. Borgefors G (1986) Distance transformations in digital images. Comput Vis Graph Image Process 34:344–371
2. Dougherty ER (1992) An introduction to morphological image processing. Tutorial text, vol TT9. SPIE Press, Bellingham
3. Duff MJB, Watson DM, Fountain TM, Shaw GK (1973) A cellular logic array for image processing. Pattern Recogn 5:229–247
4. Haralick RM (1979) Statistical and structural approaches to texture. Proc IEEE 67(5):768–804
5. Haralick RM (1987) Image analysis using mathematical morphology. IEEE Trans Pattern Anal Mach Intell 9(4):532–550
6. Haralick RM, Shapiro LG (1992) Computer and robot vision: volumes I and II. Addison Wesley, Reading MA
7. Heijmans HJAM (1991) Theoretical aspects of grey-scale morphology. IEEE Trans Pattern Anal Mach Intell 13(6):568–582
8. Pitas I (1993) Digital image processing algorithms. Prentice-Hall, Englewood Cliffs, NJ
9. Serra J (1988) Image analysis and mathematical morphology. Theoretical advances, vol 2. Academic Press, New York
10. Sonka M, Hlavac V, Boyle R (1993) Image processing, analysis and machine vision. Chapman & Hall, London
11. Sternberg SR (1978) Parallel architectures for image processing. In: Proceedings of IEEE conference international computer software and applications, Chicago, pp 712–717
12. Waltz FM (1998) Fast implementation of standard and "fuzzy" binary morphological operations with large structuring elements. In: Proceedings of SPIE conference intelligent robots and computer vision VII, vol 1002, pp 434–441
13. Waltz FM (2001) Algorithms and architectures for fast execution. In: Batchelor BG, Waltz FM (eds) Intelligent machine vision. Springer-Verlag, New York
14. Vincent L (1991) Morphological transformations of binary images with arbitrary structuring elements. Signal Process 22:3–23
15. Vogt RC (1989) Automatic generation of morphological set recognition algorithms. Springer-Verlag, New York
16. Zhuang X, Haralick RM (1986) Morphological structuring element decomposition. Comput Vis Graph Image Process 35:370–382

Further Reading

Andrews HC (1970) Computer techniques in image processing. Academic, NY

Andrews HC, Hunt BR (1977) Digital image restoration. Prentice Hall, Upper Saddle River, NJ

Ballard DH, Brown CM (1982) Computer vision. Prentice Hall, Upper Saddle River, NJ

Batchelor BG, Waltz FM (1993) Interactive image processing. Springer, New York

Batchelor BG, Whelan PF (eds) (1997a) Industrial vision systems, vol MS 97, SPIE Milestone Series. SPIE – The International Society for Optical Engineering, Bellingham, WA

Batchelor BG, Whelan PF (1997b) Intelligent vision systems for industry. Springer, London & Berlin

Baxes GA (1994) Digital image processing: principles & applications. Wiley, NY

Bezdek JC et al (2005) Fuzzy models & algorithms for pattern recognition & image processing. Springer, NY

Bovik AC (ed) (2005) Handbook of image & video processing, 2nd edn. Academic, NY

Bracewell RN (1995) Two-dimensional imaging. Prentice Hall, Upper Saddle River, NJ

Castleman KR (1996) Digital image processing, 2nd edn. Prentice Hall, Upper Saddle River, NJ

Davies ER (2005) Machine vision: theory, algorithms, practicalities. Morgan Kaufmann, San Francisco

Dougherty ER (ed) (2000) Random processes for image & signal processing. IEEE Press, NY

Dougherty ER, Lotufo RA (2003) Hands-on morphological image processing. SPIE – The International Society for Optical Engineering, Bellingham, WA

Duda RO, Hart PE, Stork DG (2001) Pattern classification, 2nd edn. Wiley, NY

Edelman S (1999) Representation & recognition in vision. The MIT Press, Cambridge, MA

Etienne EK, Nachtegael M (eds) (2000) Fuzzy techniques in image processing. Physica-Verlag, NY

Forsyth DF, Ponce J (2002) Computer vision – a modern approach. Prentice Hall, Upper Saddle River, NJ

Fu KS (1982) Syntactic pattern recognition & applications. Prentice Hall, Upper Saddle River, NJ

Geladi P, Grahn H (1996) Multivariate image analysis. Wiley, NY

Giardina CR, Dougherty ER (1988) Morphological methods in image & signal processing. Prentice Hall, Upper Saddle River, NJ

Gonzalez RC, Thomason MG (1978) Syntactic pattern recognition: an introduction. Addison-Wesley, Reading, MA

Gonzalez RC, Wintz P (1977) Digital image processing. Addison-Wesley, Reading, MA

Gonzalez RC, Wintz P (1987) Digital image processing, 2nd edn. Addison-Wesley, Reading, MA

Gonzalez RC, Woods RE (1992) Digital image processing. Addison-Wesley, Reading, MA

Gonzalez RC, Woods RE (2002) Digital image processing, 2nd edn. Prentice Hall, Upper Saddle River, NJ

Gonzalez RC, Woods RE (2008) Digital image processing, 3rd edn. Prentice Hall, Upper Saddle River, NJ

Gonzalez RC, Woods RE, Eddins SL (2004) Digital image processing using MATLAB. Prentice Hall, Upper Saddle River, NJ

Gonzalez RC, Woods RE, Eddins SL (2009) Digital image processing using MATLAB, 2nd edn. Gatesmark Publishing, Knoxville, TN, http://www.gatesmark.com

Goutsias J, Vincent L, Bloomberg DS (eds) (2000) Mathematical morphology & its applications to image & signal processing. Kluwer, Boston, MA

Hall EL (1979) Computer image processing & recognition. Academic, NY

Haskell BG, Netravali AN (1997) Digital pictures: representation, compression, & standards. Perseus Publishing, NY

Jahne B (1997) Digital image processing: concepts, algorithms, & scientific applications. Springer, NY

Jain AK (1989) Fundamentals of digital image processing. Prentice Hall, Upper Saddle River, NJ

Jain R, Rangachar K, Schunk B (1995) Computer vision. McGraw-Hill, NY

Klette R, Rosenfeld A (2004) Digital geometry – geometric methods for digital picture analysis. Morgan Kaufmann, San Francisco

Levine MD (1985) Vision in man & machine. McGraw-Hill, NY

Lillesand TM, Kiefer RW (1999) Remote sensing & image interpretation. Wiley, NY

Lim JS (1990) Two-dimensional signal & image processing. Prentice-Hall, Upper Saddle River, NJ

Mallot AH (2000) Computational vision. The MIT Press, Cambridge, MA

Marchand-Maillet S, Sharaiha YM (2000) Binary digital image processing: a discrete approach. Academic, NY

Mather PM (1999) Computer processing of remotely sensed images: an introduction. Wiley, NY

Mitiche A (1994) Computational analysis of visual motion. Perseus Publishing, NY

Mitra SK, Sicuranza GL (eds) (2000) Nonlinear image processing. Academic, NY

Nevatia R (1982) Machine perception. Prentice-Hall, Upper Saddle River, NJ

Niblack W (1986) An introduction to digital image processing. Prentice Hall, Upper Saddle River, NJ

Parker JR (1997) Algorithms for image processing & computer vision. Wiley, NY

Pavlidis T (1977) Structural pattern recognition. Springer, NY

Pavlidis T (1982) Algorithms for graphics & image processing. Computer Science Press, Rockville, MD

Petrou M, Bosdogianni P (1999) Image processing: the fundamentals. Wiley, UK

Pratt WK (1991) Digital image processing, 2nd edn. Wiley Interscience, NY

Pratt WK (2001) Digital image processing, 3rd edn. Wiley Interscience, NY

Prince JL, Links JM (2006) Medical imaging, signals, & systems. Prentice Hall, Upper Saddle River, NJ

Rangayyan RM (2005) Biomedical image analysis. CRC Press, Boca Raton, FL

Ritter GX, Wilson JN (2001) Handbook of computer in vision algorithms image algebra. CRC Press, Boca Raton, FL

Rosenfeld R, Kak AC (1982) Digital picture processing, vol 1 & 2, 2nd edn. Academic, NY

Russ JC (1999) The image processing handbook, 3rd edn. CRC Press, Boca Raton, FL

Schalkoff RJ (1989) Digital image processing & computer vision. Wiley, NY

Serra J (1982) Image analysis & mathematical morphology. Academic, NY

Seul M, O'Gorman L, Sammon MJ (2000) Practical algorithms for image analysis. Cambridge University Press, Cambridge, UK

Shapiro LG, Stockman GC (2001) Computer vision. Prentice-Hall, Upper Saddle River, NJ

Sid-Ahmed MA (1995) Image processing: theory, algorithms, & architectures. McGraw-Hill, NY

Smirnov A (1999) Processing of multidimensional signals. Springer, NY

Snyder WE, Hairong Qi (2004) Machine vision. Cambridge University Press, NY

Soille P (2003) Morphological image analysis: principles & applications, 2nd edn. Springer, NY

Sonka M, Hlavac V, Boyle R (1999) Image processing, analysis, & computer vision. PWS Publishing, NY

Tou JT, Gonzalez RC (1974) Pattern recognition principles. Addison-Wesley, Reading, MA

Ulichney R (1987) Digital halftoning. The MIT Press, Cambridge, MA

Umbaugh SE (1998) Computer vision & image processing: a practical approach using CVIPtools. Prentice-Hall, Upper Saddle River, NJ

Umbaugh SE (2005) Computer imaging: digital image analysis & processing. CRC Press, Boca Raton, FL

Whelan PF, Molloy D (2000) Image processing in Java. Springer, London

Won CS, Gray RM (2004) Stochastic image processing. Kluwer/Plenum, NY

15 Imaging and Range Image Processing

Dongming Zhao
The University of Michigan-Dearborn, Dearborn, MI, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_15,

In many industry applications, the image processing tools with input images in 2-D format may find themselves lacking the power of dimensional measurement in a space that has to deal with a third dimension. For an example of typical pick-and-place process on a manufacturing line, a robot tries to grab an unpolished engine block from a bin and the robot controller should give the guidance to the robot arm so that the orientation of the engine block is input to the robot arm prior to its movement. In such a scenario, range information of the engine block, or the depth in the *Z*-direction, should be part of the input images. The conventional imaging methods for capturing and analyzing grayscale images are inadequate for this type of application.

Range imaging deals with *Z*-directional data. When many call it 3-D, it is in fact 2.5-D, since we cannot get image data on the negative *Z*-direction; therefore, it is more pertinent to call it range imaging. There are several techniques that are found to be valid approaches to get range images. In this chapter, we introduce two imaging devices that produce 3-D images and applications using these imaging devices for industry automation.

15.1 Imaging Devices

In range image data collection, the pixel information within an image consists at least of the relative distances from a sensor to imaged points on objects. A conventional means to do so is to derive the range data by processing images acquired with structured light, triangulation or stereo pairs. These image acquisition methods employ industry-standard CCD cameras and require strict lighting environment and are limited to geometric properties of imaged objects.

The laser radar technology provides another valid tool for acquiring range image data [1–3]. A typical approach uses a modulated laser beam and collects the phase differentials between light source (laser) and collected infrared signal to postulate relative distance of surface geometry. With this approach, a classical design methodology employs a point-of-light, collecting point by point the imaged object to form pixel by pixel in a range image [4–10]. The modulating frequency determines the ambiguity distance, which is typically a fraction of actual distance from sensor to object. When the distance is required, additional measuring has to be performed with other means.

The laser radar imaging has been developed over the past few decades, and the scannerless laser radar was developed in the early 1990s. Just a few years after the invention of the laser, workers at Hughes Aircraft demonstrated the first application of pulse lasers for distance measurement. By photographically recording oscilloscope traces of the output pulse from a ruby laser and its subsequent echo from the remote target, they were able to measure the distance to the target. Now over 30 years later, laser rangefinder technology has matured, and a variety of instruments are commercially available. For many applications, however, the state of the art in laser rangefinders is still relatively primitive: data acquisition rates are too slow for the three-dimensional (3-D) imaging required for industrial applications, and the equipment is often too delicate and bulky for many applications.

The principle of CW radar, as illustrated in ➋ *Fig. 15.1*, relies on the measurement of the phase difference between an intensity-modulated probe beam and its return from the target. The range to the target, *R*, is calculated from

$$R = \frac{c\phi}{4\pi f_m} \tag{15.1}$$

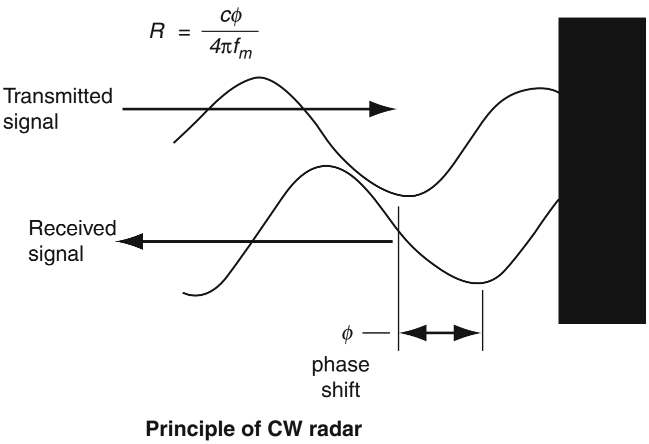


Fig. 15.1
Continuous wave (CW) laser radar works by measuring the relative phase shift between a transmitted modulated signal and the signal received after reflection from the object of interest. The range to the target, *R*, is calculated from $R = c\phi/4\pi f_m$, where *c* is speed of light, ϕ is the round-trip phase shift, and f_m is the modulation frequency

where *c* is the speed of light, ϕ is the round-trip phase shift and f_m is the modulation frequency. In a CW radar (or laser radar), the range resolution is limited by the resolution to which the relative phase shift of the return signal can be measured.

15.1.1 Scanning Based Laser Radar Imaging

A classical approach to the scanning based laser radar imaging is typical of radar application. The lighting source, or active lighting, is from a laser source. This laser source is typically designed using infrared lasers. Infrared lasers work well within the near infrared spectrum, and the key issue of eye safety has been addressed for many years, and the technology has improved, so that the issue is no longer a major concern.

In continuous-beam laser sensors, a continuous-beam laser is used rather than a pulsed one. The phase shift in the return signal is used to measure the distance. A range image is generated by scanning the laser beam horizontally and vertically across the scene. A representative laser radar system is illustrated in *Fig. 15.2*. The principle of image acquisition works as follows. A laser beam is sent out onto faceted polygon and is then reflected onto a nodding mirror. The mirror reflects the beam to the targeted object, as shown in direction *Z* in *Fig. 15.2*. The imaging object is within the viewing scope of area on the mirror. When the laser beam returned from the object is received at the mirror, it is transmitted to the sensor and then to the demodulation processor. The laser beam is modulated on a continued amplitude of a sinusoidal wave with a consistent frequency. The phase shift is compared and interpreted as the geometrical differentiation pixel by pixel on the received image grids, and such phase shift parameters are then translated into the range information.

The mechanism of range imaging is illustrated in ➲ *Fig. 15.2*. The laser source produces a laser beam that is projected onto faceted rotating mirror. Within a fraction of a millisecond, it reflects the incoming beam, r_1, onto a nodding mirror, as in r_2. The nodding mirror projects the beam r_3 toward the object. The light beam r_2 acts similar to a scanning line of TV frame. The nodding mirror moves through an angle, β, which determines the viewing range vertically. The scanning angle, α, determines the viewing range horizontally. The nodding mirror will be in initial position after the beam r_3 travels up and down, limited by the viewing angle range, thus completing a frame of imaging procedure.

The range image is therefore referenced as a 3-D image with X-Y as the imaging plane, as shown in ➲ *Fig. 15.3*, when the range of a single point at (i, j) is inferred through intensity modulation on a continued waveform, as indicated by (➲ 15.2)–(➲ 15.12).

Each pixel in the range image, denoted by $r(i, j)$, $0 \leq i < R(\text{rows})$ and $0 \leq j < C$ (columns), is converted to Cartesian coordinates as follows:

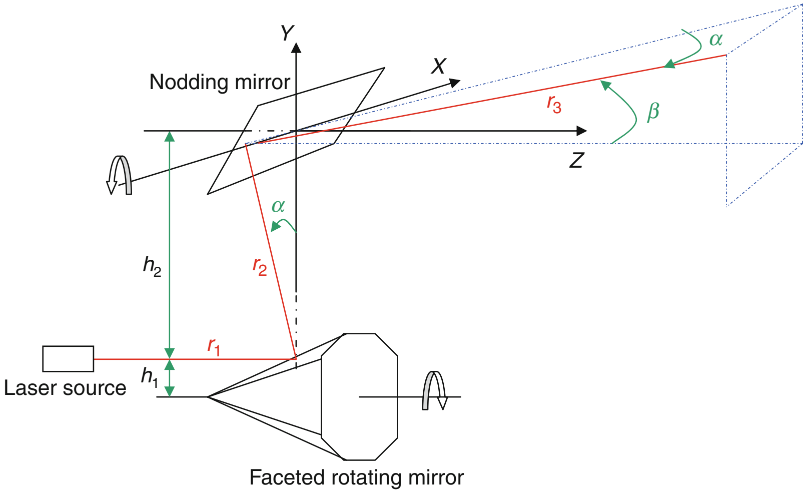


Fig. 15.2
Illustration of range image data acquisition using continuous scanning laser technology

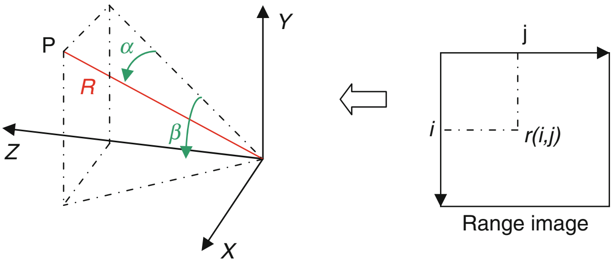


Fig. 15.3
Illustration of range image in 3-D imaging space and on imaging plane in X-Y

$$x(i,j) = dx + r_3 .\, \sin\alpha \tag{15.2}$$

$$y(i,j) = dy + r_3 .\, \cos\alpha .\, \sin\beta \tag{15.3}$$

$$z(i,j) = dz + r_3 .\, \cos\alpha .\, \cos\beta \tag{15.4}$$

$$r_1 = (dx - h_2)/\,\delta \tag{15.5}$$

$$r_2 = \left(\sqrt{(dx)^2 + (h_2 + dy)^2}\right) \Big/ \delta \tag{15.6}$$

$$r_3 = (r\,(i,\,j) + r_0 - (r_1 + r_2)).\,\delta \tag{15.7}$$

$$\beta = \frac{\left(\frac{R-1}{2} - i\right)}{R}.V \tag{15.8}$$

$$\alpha = \frac{\left(\frac{C-1}{2} - j\right)}{C}.\,H \tag{15.9}$$

$$dx = (h_2 + dy).\ \tan\alpha \tag{15.10}$$

$$dy = dz.\tan\left(\theta + \frac{\beta}{2}\right) \tag{15.11}$$

$$dz = -h_1\,.\,(1 - cos\,\alpha)\,/\,tan\,\gamma \tag{15.12}$$

In (➲ 15.8) and (➲ 15.9), H and V, respectively, represent the horizontal and vertical fields of view; r_0 the standoff distance (length of the laser beam at the point where $r = 0$); γ the slope of the facets of the rotating mirror with respect to the z-axis; and θ the angle of the nodding mirror with respect to the z-axis when $\beta = 0$.

➲ *Figure 15.4* shows a pair of images. The left is similar to the conventional intensity image, where on the right is its corresponding range image. In the range image, the brightness represents the closeness of the object surface points to the sensor; therefore, the range is defined, which is further inferred into Z-direction measure.

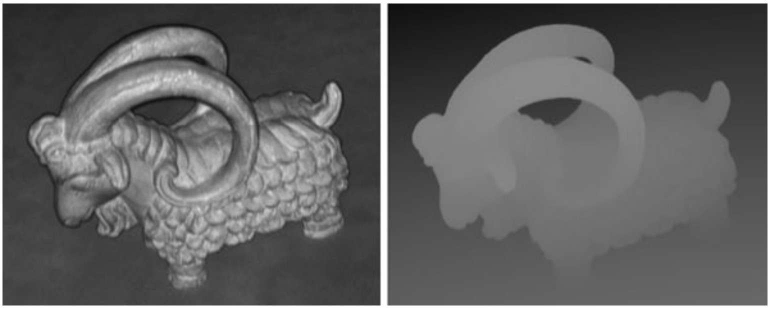


Fig. 15.4
On the *left* is an intensity image, and on the *right* is its corresponding range image

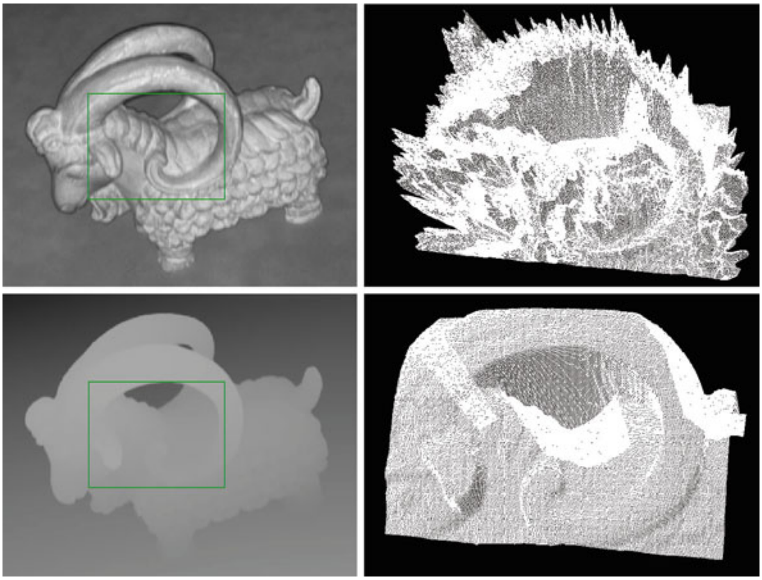


Fig. 15.5

On the *top* row, the region of interest in the status is within the squared, and its corresponding intensity is plotted with a wireframe as shown on its *right*. On the *bottom* row, the region of interest in the range image (*left*) is plotted in the *right*

To further illustrate the difference between two images in *Fig. 15.5*, we use the plots to enhance the views of these two images. On the top row, the region of interest in the status is within the squared, and its corresponding intensity is plotted with a wireframe as shown its right. On the bottom row, the region of interest in the range image (left) is plotted in the right. It is clear that the range image has the representation of the *Z*-direction range where the intensity image is meaningful to human perception but its data representation is rather much less meaningful. *Figure 15.6* shows another illustrative pair of images that are used for testing pick-and-place operations on manufacturing assembly lines (*Fig. 15.6*).

15.1.2 Scannerless Laser Radar Imaging

Shown schematically in *Fig. 15.7*, the scannerless laser radar illuminates the entire field of view at once with sinusoidally intensity-modulated laser light. Light scattered from the scene is collected by a lens and imaged onto a microchannel plate image intensifier tube. The gain of the image intensifier is modulated synchronously with, and at the same frequency as, the laser illuminator, and the optical output of the modulated image intensifier tube is then integrated and recorded by a two-dimensional (2-D) charge-coupled device (CCD) detector. Each pixel in the CCD contains information about the phase shift (and thus the range) to its image point in the scene. The net result is a phase-sensitive detector that operates on an entire image at once instead of scanning and sequentially measuring the phase shift to each pixel in the image.

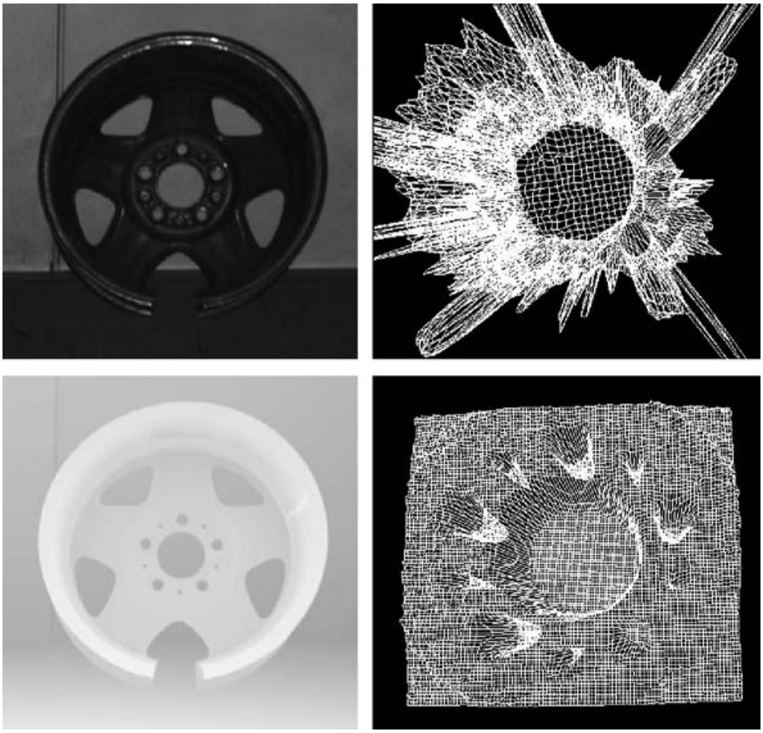


Fig. 15.6
On the *top* row, the region of interest in the *left* intensity image is plotted in wireframe at the *right*. On the *bottom* row, the region of interest in the range image (*left*) is plotted in the *right*

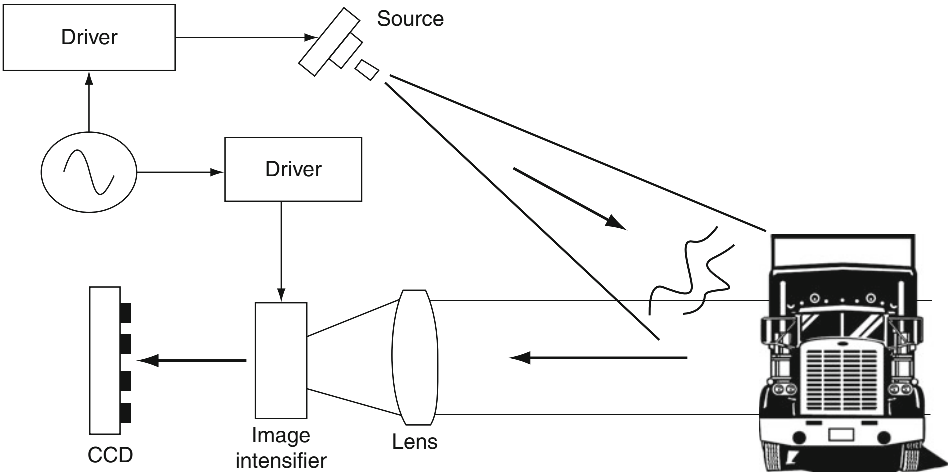


Fig. 15.7
Schematic diagram of the scannerless laser radar. A laser diode illuminates the entire field of view at once with a sinusoidally intensity-modulated laser light

15.1.2.1 Theory of Operation

Continuous wave (CW) rangefinders work by measuring the phase difference between an intensity-modulated transmitted laser beam and its return from the target. As shown in *Fig. 15.1*, the spatial wavelength, λ, of the sinusoidally modulated beam is given by

$$\lambda = \frac{c}{f_m} \tag{15.13}$$

where c is the speed of light and f_m is the modulation frequency. The light travels from the transmitter to the target, and some of the light is backscattered by the target and collected by the receiver optics. In its round trip to the target and back, the light acquires a phase shift relative to the transmitted beam of

$$\phi = \frac{4\pi f_m R}{c} \tag{15.14}$$

where R is the distance (range) to the target. In order to measure the range to the target, the CW rangefinder measures the relative phase shift, ϕ, and computes the range using the equation

$$R = \frac{c\phi}{4\pi f_m} \tag{15.15}$$

Since relative phase is a periodic function (with period 2π), ranges that produce round-trip phase shifts of $\phi \geq 2\pi$ are not distinguishable from ranges with phase shifts $\phi - 2\pi$. Thus, with a CW rangefinder, one must ensure that the modulation frequency is selected to produce unambiguous range measurements for the expected maximum range interval. In CW radar, it is common to define an ambiguity interval, R_{amb}. This is the range interval over which the round-trip phase delay will be $\phi \leq 2\pi$, and distance measurements will be unambiguous. R_{amb} is given by

$$R_{\text{amb}} = \frac{c}{2f_m} = \frac{\lambda}{2} \tag{15.16}$$

The theorem within the scannerless laser radar imaging can be expressed in the following equations and illustrated with *Fig. 15.8*. The signal flows are as follows:

$$P_t = P_o[1 + m_l \sin(\omega t)] \tag{15.17}$$

$$P_r = P_a[1 + m_l \sin(\omega t + \phi)] \tag{15.18}$$

$$G = G_0[1 + m_g \sin(\omega t)] \tag{15.19}$$

$$I = \int_0^{nT} SG_0(1 + m_g \sin(\omega t))P_a(1 + m_l \sin(\omega t + \phi))dt \tag{15.20}$$

and the output signal at the detector is

$$I = nTSG_0P_a(1 + \frac{1}{2}m_g m_l \cos(\phi)) \tag{15.21}$$

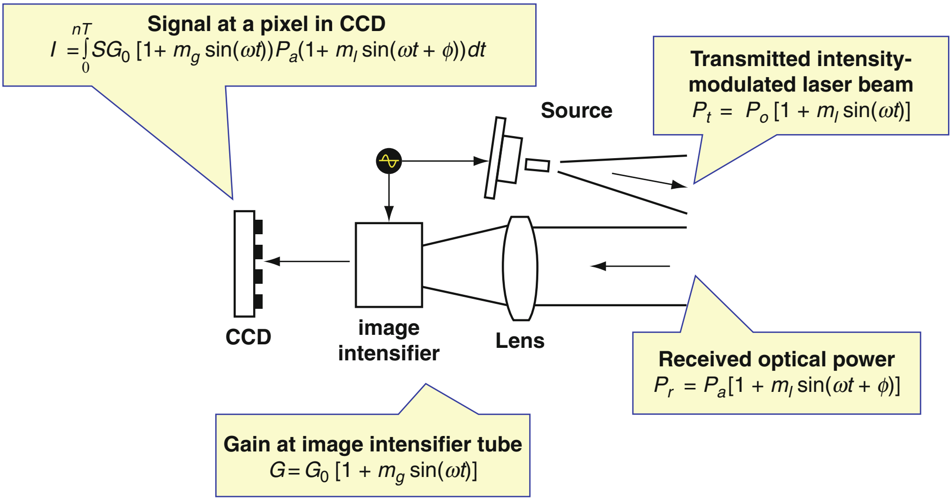

Signal at a pixel in CCD
$I = \int_0^{nT} SG_0\,[1+ m_g \sin(\omega t))P_a(1+ m_l \sin(\omega t + \phi))dt$
Transmitted intensity-modulated laser beam
$P_t = P_o\,[1 + m_l \sin(\omega t)]$
Source
CCD
image intensifier
Lens
Received optical power
$P_r = P_a[1 + m_l \sin(\omega t + \phi)]$
Gain at image intensifier tube
$G = G_0\,[1 + m_g \sin(\omega t)]$

Fig. 15.8
Operation block diagram of a scannerless laser radar imaging system

15.2 Machine Vision Application Using 3-D Imaging

Machine vision is an important subject within the machine intelligence systems area. The goal of a computer vision system is to understand the given image data and use the extracted information for a high-level task. Typical application tasks include (1) the assembly or inspection of manufacturing parts, (2) the navigation of autonomous vehicles and (3) the analysis of microscopic images [1]. Identifying and locating a specified object in the scene is one of the most important goals within these tasks. The general problem of identifying the desired object is referred to as *object recognition*.

For an ideal computer vision system, the objects in the following cases should be located: (1) objects may have arbitrary and complicated shapes, (2) objects may be viewed from various directions and (3) objects may be partially occluded by other objects in a scene. Specially, in a manufacturing environment, such a machine vision system can be used in determining the location of grasp sites for a robot to manipulate for an object in the assembly or inspection of parts.

To realize such vision systems, system designers must resolve the following issues: (1) the type of sensor for data collection, (2) the methods of describing the collected data and the object and (3) the methods of matching the object descriptions obtained from the input data to the desired object model. The sensor determines the resolution and precision. More importantly, it determines whether the data provides 2-D or 3-D information of the scene. Object representations are used to describe the collected data and the desired object model. Also, the descriptions are used to calculate the various properties of objects in the scene during the matching stage. Matching strategies are performed for real-time online processes and must resolve many ambiguities between the input data and the model descriptions. Once the match has been established, the orientation and location information of the target object can be computed for further robot tasks.

In this section, the basic issues and corresponding methods in 3-D object recognition are discussed in detail. For an automated assembly application, a 3-D object pose estimation method is developed using range images as the input data. The local surface geometrical feature is described by a 3-D surface descriptor, Angle Distance Map (ADM). A triangular mesh model of the 3-D object surface is used for decreasing the cost of computing and storage of applying the ADM descriptions. The principal component analysis (PCA) method is applied on the original ADM description to extract the principal features for matching strategies. The pose information is computed according to the final 3-D feature points. The proposed approach is tested in an application for flexible robot assembly. The experimental results show that accurate pose estimation can be obtained for the 3-D computer vision applications.

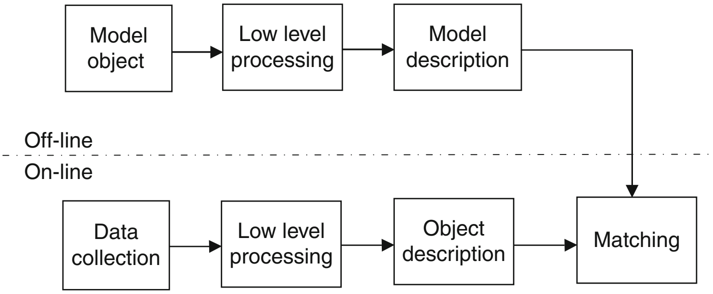
15.2.1 Computer Vision System

The basic issues involved in the design of a computer vision system are outlined in the introduction section. ➋ *Figure 15.9* shows a general paradigm of a computer vision system. The system includes online and offline processes. In an offline process, a description of a standard object is established as the model object. In an online process, the input data are represented by the same description method that is used in the offline process. The input object description and the model description are then compared in the matching stage. The located object and its pose information are used for further applications.

15.2.2 Low-Level Processes

For the specified high-level processes, the collected range images must be represented using symbolic descriptions. The first step is the partitioning of the input images based on the nature of the input data and the high-level processes. Also, since the ideal vision system must be able to recognize arbitrarily shaped objects from any view direction, the contour information for 3-D object recognition is very important. Segmentation and edge detection are the primary low-level processes in 3-D object recognition.

Range scanners sample points on the object surface. Therefore, the surface properties, such as surface normal and surface curvature, are used to obtain the segmentation and edge detection. Since the distance measurement made by the laser sensors is often inaccurate



■ **Fig. 15.9**
A general paradigm in computer vision

due to detection noise, quantization and calibration errors, the original range data should be smoothed. Most noise processes have been modelled as additive Gaussian stochastic processes independent of the scene. In range image processing, most researchers have adapted noise reduction methods developed from intensity images [5]. The methods of segmentation and edge detection for range images are discussed in this section.

Segmentation is an assignment of pixels into one of many disjoint sets such that the pixels in each set share a common property. Segmentation has been most often used in range image analysis and recognition as it helps partition an image array into low-level entities. In general, segmentation methods for range images are classified into two categories. One is based on region features, and the other is based on edge information.

In range image processing, unlike traditional 2-D intensity image processing, there are three primary types of edges in 3-D range images. These types of edges are step edge, which is discontinuous in depth value, roof edge, which is discontinuous in surface normals, and curvature edge that is discontinuous in a variety of surface normals. Since the roof and curvature edges do not correspond to depth discontinuity, the edge detection in range images is more complicated than in traditional intensity images.

There have been a number of research reports on edge detection in range images [11]. To extract all of the three types of edges in range images, the edge detection methods are divided into two classes. In the first class of methods, different operators or algorithms are applied to extract different types of edges. A merging process is then used to obtain final edge results.

15.2.3 3-D Object Description

Design of an appropriate surface representation scheme is a crucial factor for 3-D object recognition. Armen and Aggarwal [11] and Besl and Jain [12] made comprehensive surveys of 3-D object representation schemes and recognition strategies. Both locally and globally based representations, such as points or groups of points, straight line segmentations and polylines, planar and quadric surface patches, constructive surface geometry and superquadrics, have been proposed to describe a 3-D object surface.

Bolle and Vemuri utilized the Extended Gaussian Image (EGI) to describe 3-D objects in terms of their surface normal distributions on the unit sphere [13]. This type of global surface representation handles convex polyhedra efficiently, but for arbitrarily curved objects, they are either approximated by planar patches or divided into regions based on their Gaussian curvature. Besl and Jain proposed their representation method for quadric surfaces [14]. The Gaussian and Mean curvature information is used to classify the quadric surface types. These quadric surface–based representations may not adequately illustrate some objects with complicated surface forms. Superquadrics are an extension of the basic quadric surfaces and solids [15]. Several parameters of the superquadric function are used to determine the object's squareness, size, bending, tapering and twisting. Minimization techniques are used to recover the parameters from a set of data [16, 17]. 3-D objects with superquadric surfaces can be described accurately using the superquadric surface model. The occlusion problem can be solved using the model.

A free-form surface is defined to be a smooth surface, such that the surface normal is well defined and continuous almost everywhere, except at vertices, edges and cusps. It is not constrained to be polyhedral, piecewise quadric or superquadric. Most vision researchers have proposed both global and local structural descriptions based on local unconstrained geometrical features to represent free-form surfaces. Dorai and Jain proposed a representation scheme for free-form objects based on COSMOS (Curvedness-Orientation-Shape Map On Sphere) [18]. In their approach, an object is represented in terms of maximal surface patches of constant shape index. The maximal patches are mapped onto the unit sphere according to their orientations and shape spectral functions. This representation yields a meaningful and rich description for object recognition. A further analysis is needed for the object surfaces with rapidly changed shape indices. For the objects with some complex surface shapes, some comprehensive methods are also needed to analyse the angle features extracted by tripod operators. A 3-D surface descriptor, or a spin image, has been proposed by Johnson and Hebert [19]. A spin image is a data level descriptor used to match the surfaces which are represented by surface meshes. For a 3-D point on an object surface, the 3-D surface descriptor is generated by accumulating two projection distances between this point and its neighbourhood points. A robust performance is demonstrated in the presence of clutters and occlusions in a 3-D object recognition system.

15.2.4 Matching and Pose Estimation

Once the appropriate descriptions are derived from the range images, the 3-D computer vision system is able to match the two descriptions for completing the task of object recognition. This is performed in two steps. In the first step, a correspondence is established between the two sets of descriptions. For some partial occlusion cases, only partial descriptions of objects can be obtained. The matching strategy should be able to establish the correspondence between the partial description of the object and its full model description. The calculation of correlative coefficient is used to match the simple descriptions, such as feature vectors. For more complicated descriptions, such as symbolic geometrical features, some matching strategies, such as tree search, attributed graphs and triplet methods, are applied to match the two descriptions. In the second step, using the established correspondence, a geometrical transformation is derived. The 3-D orientation information of the object in the scene is then estimated according to the geometrical transformation.

Object pose estimation is an important application in 3-D object recognition. There are two primary approaches to obtain pose estimation. They are a feature-based approach and a model-parameter-based approach. In feature-based approaches, pose information is directly computed from the derived features corresponding to model features. The method by Sim and Dudek utilizes image-domain landmarks as the features for pose estimation [20]. Each landmark is detected as a local extremum of a measure of uniqueness and represented by an appearance-based encoding. Pose estimation is obtained through matching the observed landmarks to the learned prototypes. Burtnyk and Greenspan proposed a signature search method for 3-D pose refinement [21]. This approach generates a synthetic signature of what the camera would see if the object were in its currently assumed pose, and then proceeds to search for this signature in the actual data. It extracts a footprint signature from the model and performs a simple search through the data for the best match. The main difficulties in these feature-based approaches are feature extraction and correspondence specification for object pose estimation.

In model-parameter-based approaches, first a parametric object model is obtained from raw data. Pose information is computed from the model parameters. Stark and Fuchs defined a 3-D object model with six degrees of freedom in their pose estimation approach [22]. An active contour model utilizing Kalman filter is used to efficiently track the object silhouette.

The 3-D pose tracker uses the results of the contour tracker. The tracking results for polyhedral objects were presented. Although not based on geometry, a recent technique called parametric eigenspace has been proposed to project a large set of data of an object into the eigenspace using principal component analysis (PCA). Crowley and Schiele proposed a robot position estimation approach based on PCA of range data [23]. The structure of an environment is represented as a family of surfaces in the constructed eigenspace. Subsequent range data sets from the environment project as a point in this space. This representation converts the problem of associating range profiles to possible poses into a table lookup. Pose information is selected from the candidate poses using a Kalman filter.

In 3-D image processing, low-level processing, object description and matching methods are the most important tasks for 3-D object recognition. There have been a lot of approaches developed for these tasks. The selection of various methods is decided by the requirements of 3-D computer vision applications. The field knowledge of the applications is also applied in the three primary tasks to build the 3-D computer vision system.

15.3 Industry Application Example Using 3-D Imaging

In this section, a 3-D object pose estimation method is developed for guiding flexible robots on an automated assembly line. Accurate pose information, including the position and orientation of torque converters, is needed to guide a robot to grab a target torque converter. A laser camera is mounted on top of the working cell. ➲ *Figure 15.10* shows the work field and a grasping example of the automated assembly application.

➲ *Figure 15.11a, b* show, respectively, the intensity image and its corresponding range image. There are five torque converters laid in a bin, as shown in ➲ *Fig. 15.11*. In this assembly application, the objects, torque converters, are stacked in a bin of four layers. Using the range images, the developed 3-D object pose estimation method provides the orientation and position information of each torque converter. The framework includes two stages: (1) Segmenting each torque converter and locating the centre positions of torque converters, (2) Determining the orientation of the located torque converters. A diagram of the integrated 3-D pose estimation method is shown in ➲ *Fig. 15.12*.

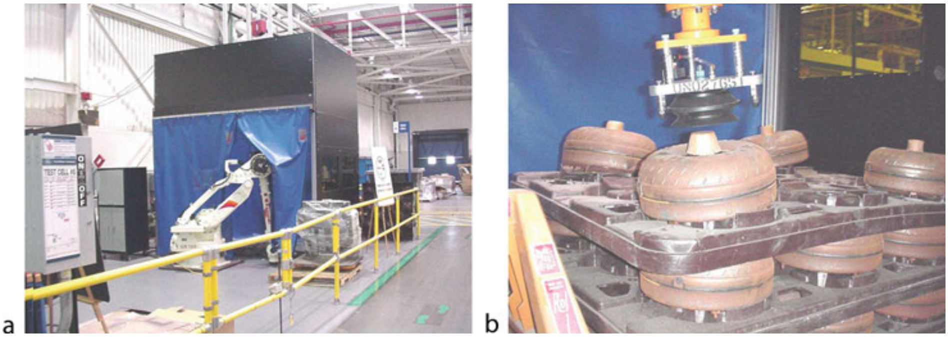


Fig. 15.10

An automated assembly application. (a) Work field of the robotic assembly application; (b) a grasping example of the robotic assembly application

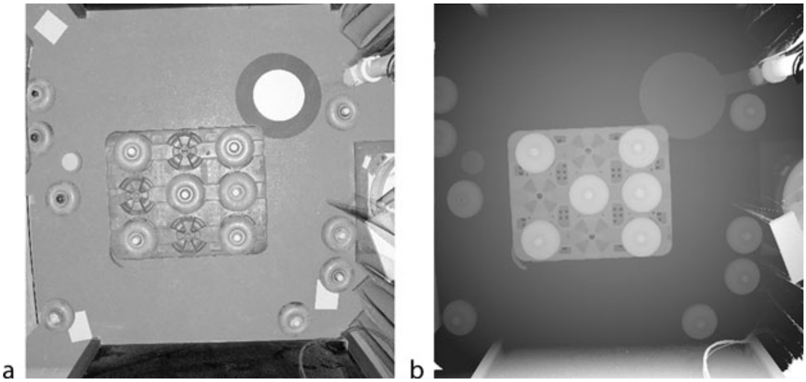


Fig. 15.11
An example image of the assembly application. (a) Intensity image; (b) corresponding range image

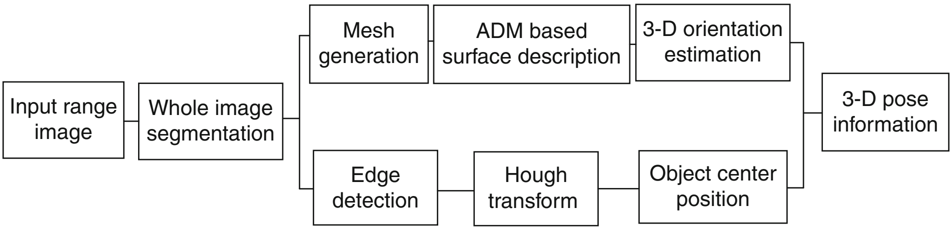


Fig. 15.12
A diagram of the integrated 3-D pose estimation method for robotic assembly application

15.3.1 Low-Level Processing

The goal of this 3-D computer vision system is to obtain the pose information for each torque converter in the stack. Therefore, the tasks in low-level processing are the localization of the stack area and each torque converter inside the stack area from the input workcell range image. As discussed earlier, edge detection and segmentation approaches are applied on the original range images.

As shown in *Fig. 15.11b*, there are only step edges between the stack area and the background. A normal intensity-based edge detection method is used to extract the edge. *Figure 15.13a* shows the edge detection result of *Fig. 15.11b*. The stack area is a large rectangle area in the range images. Using this application knowledge, a Hough transform is applied to the detected edge results. The Hough transform is a traditional image processing algorithm to extract some geometrical shapes, such as line and circle. After the stack area is extracted, nine subareas are equally segmented for each torque converter. These subareas are applied as input for the orientation and position estimation methods. *Figure 15.13* shows the edge detection and segmentation results.

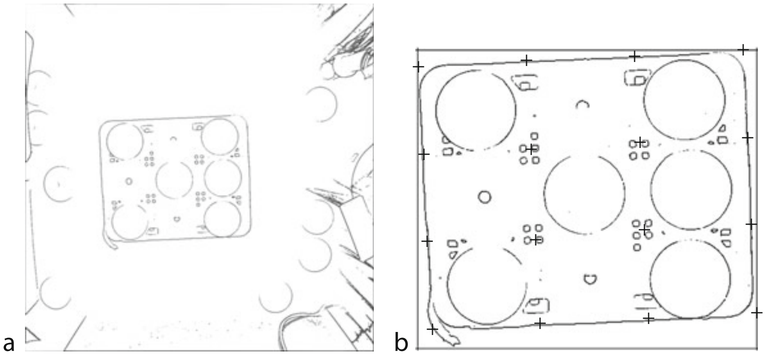


Fig. 15.13

Low-level processing of the 3-D computer vision system. (a) Edge detection result of the range image in Fig. 4b. (b) Extracted stack area and nine equally segmented subareas

15.3.2 3-D Surface Representation Using Angle Distance Map

In range images, surface shape is described by a dense collection of 3-D points. The 3-D points with their associated 3-D positions and surface normals are used as the fundamental components of our surface representation. An Angle Distance Map is a 3-D surface descriptor proposed to describe the local geometric features using global surface information. Using point *P* and neighbourhood point *Q*, as shown in *Fig. 15.14*, angle α between surface normals $\vec{n}_1$ and $\vec{n}_2$ of these two points and their 3-D distance *d* are the two parameters in the ADM descriptions (*Fig. 15.14*).

The ADM M_p of point *P* is defined as the function that projects its neighbourhood 3-D points to the 2-D coordinates corresponding to point *P*

$$M_P : R^3 \rightarrow R^2,\ M_P(Q) \rightarrow (\alpha,\ d) \tag{15.22}$$

If the coordinates of points *P* and *Q* are (x_1, y_1, z_1) and (x_2, y_2, z_2) and their surface normals are $(f_{x1}, f_{y1}, -1)$ and $(f_{x2}, f_{y2}, -1)$, angle parameter $\alpha$and distance parameter *d* are given by

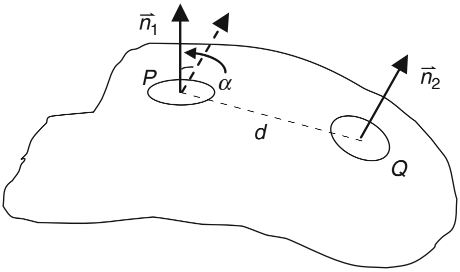


Fig. 15.14

A paradigm of angle and distance attributions used in ADM

$$\alpha = \arccos\left(\frac{f_{x1}f_{x2} + f_{y1}f_{y2} + 1}{\sqrt{f_{x1}^2 + f_{y1}^2 + 1}\sqrt{f_{x2}^2 + f_{y2}^2 + 1}}\right), \tag{15.23}$$

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2}. \tag{15.24}$$

In Angle Distance Map descriptions, the parameter α is an angle in degrees between 0 and 180. Parameter *d* is a 3-D distance in Cartesian coordinates. Normally the proposed ADM descriptions are square maps. The size of an ADM description is determined by the degree scale of parameter α. The distance area changes with different distance scales of parameter *d*.

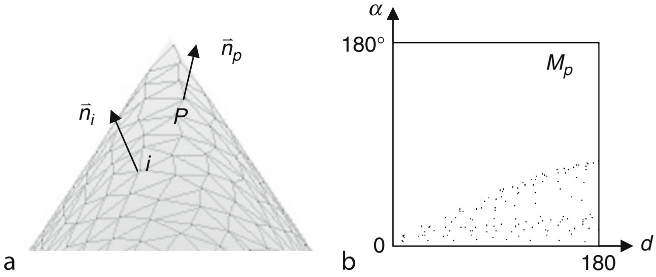
The ADM-based surface representation describes the geometric features of a 3-D point. For an object surface with dense collections of points, a triangular mesh model of the 3-D object is used to contain the computing complexity and the size of data representation of the ADM description. A triangular mesh is a piecewise linear surface, which consists of vertices, edges and faces. In this study, the vertices of the triangular meshes on an object surface are used to generate the ADM descriptions for the object surface representation.

➋ *Figure 15.15* shows an example of the ADM description of a point on a cone surface. As shown in ➋ *Fig. 15.15a*, for vertex *P* on the cone surface, ADM description M_p is obtained. The size of M_p is selected to be 180 × 180, and the scale of distance is selected to be one. The ADM description M_p is shown in ➋ *Fig. 15.15b*.

In the ADM-based surface description, each vertex has an ADM description. The surface geometric features are represented according to the distributions of ADM descriptions.

The ADM descriptions are compared using their correlation coefficients. Given two ADM descriptions M_p and M_q with a size of $N * N$, the linear correlation coefficient, $R\,(M_p, M_q)$, is given by

$$R(M_p,\ M_q) = \frac{N \times N \times \sum_i \sum_j p_{ij} q_{ij} - \sum_i \sum_j p_{ij} \sum_i \sum_j q_{ij}}{\sqrt{\left(N \times N \times \sum_i \sum_j p_{ij}^2 - \left(\sum_i \sum_j p_{ij}\right)^2\right)\left(N \times N \times \sum_i \sum_j q_{ij}^2 - \left(\sum_i \sum_j q_{ij}\right)^2\right)}} \tag{15.25}$$



■ **Fig. 15.15**
An ADM description example of a cone. (a) The range image and surface mesh of part of a cone; (b) the ADM description result of vertex *P*

Coefficient R is between -1 (anti-correlated) and 1 (completely correlated), and it provides a measure for comparing two ADM descriptions. The surface representation of ADM description is estimated through the correlation coefficient.

15.3.3 3-D Object Pose Estimation for Automated Assembly Application

In this automated assembly application, the 3-D orientation and position information of each torque converter is required for the robot system. The assembly application is illustrated in *Fig. 15.16* that shows the robot for pick and place operations and its test bed that is enclosed for eye safety issues. The red box on the top of the test bed is the range imaging device. *Figure 15.16b* shows the robot in action to move and angled-point to the object, a converter, to be picked and placed. From the top of the test bed, the range imaging device provides the vision function for the robot. The images, both intensity and range, are shown in *Fig. 15.16c*. The image processing in the vision system provides guidance data for the robot to move to a precise position and angled-aim to the tilted converter for picking and placing. In this 3-D object pose estimation method, the position and orientation information is computed using separate methods.

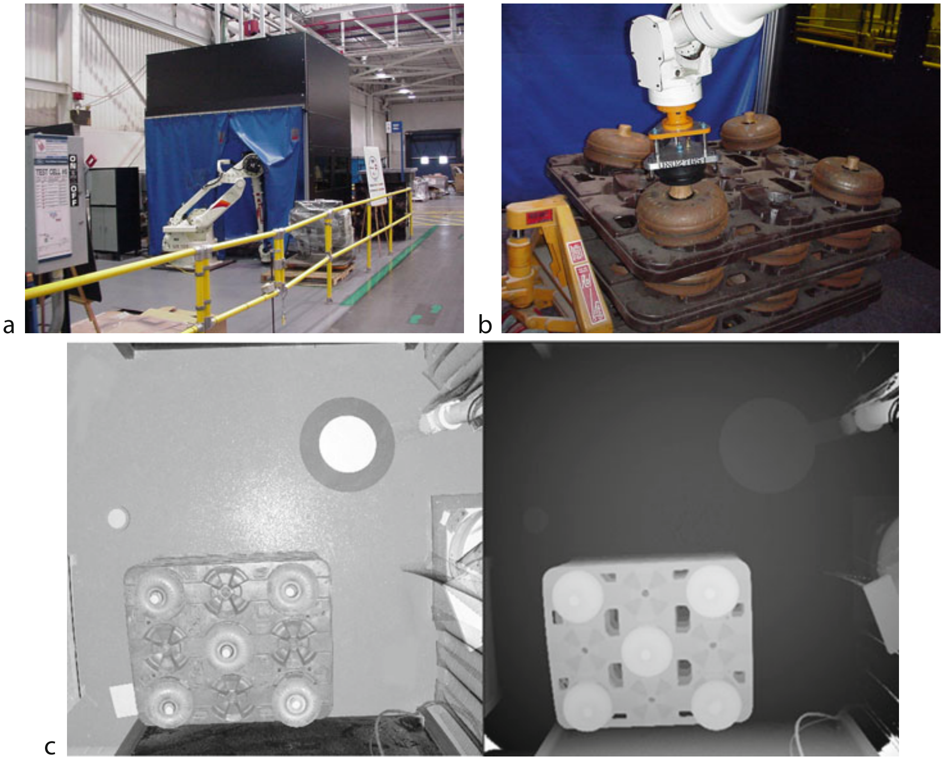


Fig. 15.16
(a): Robot and test bed: the red box on top of bed is the imaging device. (b): Robot moves to pick up a tilted converter. (c): *Left*: intensity image; *Right*: range image

15.3.3.1 Centre Position Estimation

The contour of torque converters is a circle from the camera's view direction. The circle centre is used as the centre position of each torque converter. In the low-level processing, each torque converter is segmented within each sub-area. The Hough transform is applied to extract the circle shape within each sub-area. The centre position of the extracted circle is taken as the position of the torque converter. ➋ *Figure 15.17* shows the centre position results of ➋ *Fig. 15.13b*.

15.3.3.2 Orientation Estimation Using ADM Descriptions

The torque converters appear in different sizes and directions due to various levels and positions of stacks and the different orientations of torque converters themselves. ➋ *Figure 15.18* shows two examples. ➋ *Figure 15.18a, b* shows the intensity images with size of 130 × 130 and size of 105 × 106 pixels. The top circular region of a torque converter is a ring-shaped band, as illustrated in ➋ *Fig. 15.18a, b*. This ring-shaped band is an area identified as

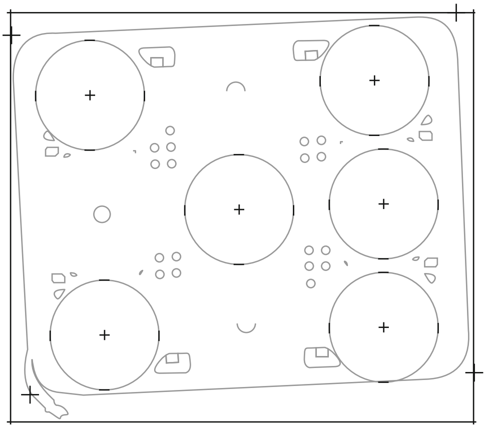


Fig. 15.17
Position estimation results of ➋ *Fig. 15.13b*

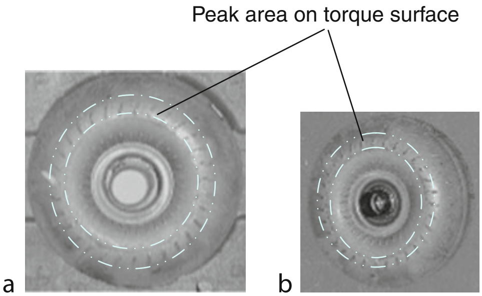


Fig. 15.18
Two torque examples and their peak areas. (a) Intensity image of example 1; (b) intensity image of example 2

a region that is most representative to the pose orientation of the object of interest. The surface normal direction of the tangent plane of the ring-shaped area is taken as the torque converter orientation.

In this orientation estimation approach, the proposed ADM descriptor is applied to represent the 3-D surface features. The correspondences between pairs of the features are obtained through the ADM descriptions. Since the ADM description is a local surface representation, the points belonging to the ring-shaped areas are extracted first. The tangent plane of the ring-shaped area is then generated through fitting the extracted feature points. Finally, pose information is estimated according to the surface normal of the extracted tangent plane.

For an efficient comparison between two original ADM descriptions, principle component analysis is applied to compress ADM descriptions. The PCA or the Karhunen-Loeve expansion provides a method to automatically determine a linear subspace with minimal number of dimensions. Such a representation is optimal in terms of the least-square error [13]. By using only the most important eigenvectors as defined by the corresponding eigenvalues, the dimension of the eigenspace is significantly reduced with a minimal loss in precision and reliability. This compression process greatly reduces computing complexity.

In this orientation estimation method, PCA is used to compress the original ADM descriptions consisting of all vertices on the object surface. Since the original ADM vectors are correlated, the Compressed ADM (CADM) descriptions are obtained through eliminating the eigenspace dimensions whose eigenvalues are relatively small. The projected CADM dimension, *s*, is determined according to the needed fidelity in reconstruction and the variance among ADM vectors.

The extracted subeigenspace constructed by *s* significant eigenvectors and the *s*-dimensional CADM vectors of model features is used to match the feature points in the pose estimation method.

As shown in *Fig. 15.19*, a general procedure of the 3-D object pose estimation is divided into two sections, the offline training and the online testing. In the offline training section, the feature points used to estimate orientation information are specified on the model object surface. The model object surface is first described in terms of the surface meshes. The ADM descriptions of mesh vertices are then obtained. The eigenspace of all ADM descriptions is constructed based

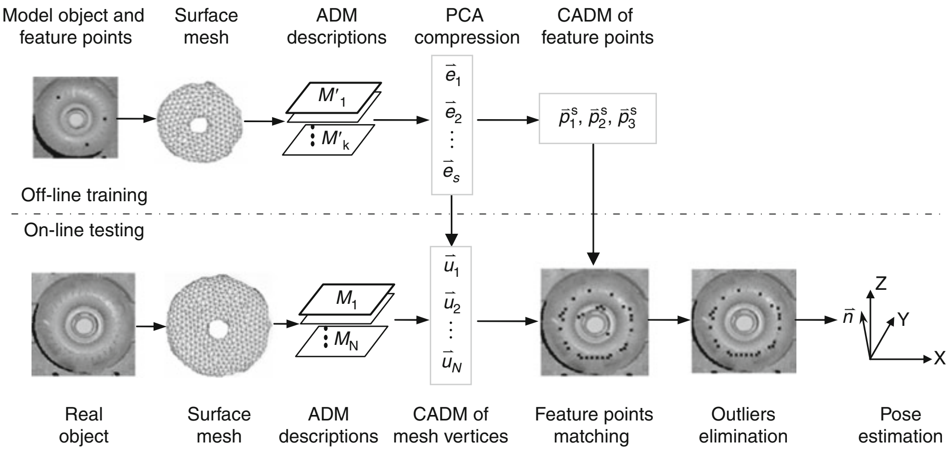


Fig. 15.19

A procedure of 3-D object pose estimation for an automated assembly application

on the PCA approach. According to the reconstruction accuracy and the variance among the ADM vectors, the CADM dimension s and corresponding s significant eigenvectors are generated. The CADM vectors of the specified feature points are then obtained by projecting the original ADM descriptions onto the subeigenspace constructed using the extracted s significant eigenvectors. The CADM vectors of the specified feature points and the s significant eigenvectors of the subeigenspace are the two components for the online testing process.

In the online testing section, the surface mesh representation for object surface is first generated. The ADM descriptions of the mesh vertices are then obtained. All ADM descriptions are projected into the subeigenspace using the s significant eigenvectors which are extracted during the offline training process. The projected CADM vectors are matched with the obtained CADM vectors of the specified feature points of the model object. The correspondences of the feature points are established by the maximal correlation coefficient between the training sampler and the testing objects.

Once the feature points are matched, the tangent plane of the ring-shaped area is estimated according to the least-square surface-fitting approach. Some points not belonging to the ring-shaped area are also extracted due to the sensor error or the difference between the model torque converter and the tested torque converter. Therefore, an iterative outlier elimination method is applied based on the 3-D projection distance between the points and the estimated tangent plane. For each elimination step, the point with the largest distance to the fitted tangent plane is found first. If the largest distance is also larger than a predefined threshold, the point is eliminated from the feature point set. This iterative process terminates only if no point is eliminated from the current feature point set. The tangent plane fitted by the final feature points is used to compute the orientation of torque converter.

15.3.4 Experiments and System

In this set of experiments, 45 torque converters placed in eight stacks with different positions and levels in a workcell are tested by the proposed orientation estimation approach. Ten torque converters with different positions and levels are used as model parts to obtain the pose information of all of the 45 torque converters. The orientation information of each torque converter is computed according to the Cartesian coordinates of range data. The errors between the computed accurate orientation and the estimated orientation of each torque converter are calculated to evaluate the accuracy of this approach. ➋ *Figure 15.20a* shows a one-level stack with five torque converters. A model torque converter with specified feature points is shown in ➋ *Fig. 15.20b*.

➋ *Figure 15.21* show the final matched feature points of the torque converters in ➋ *Fig. 15.21a* according to model part in ➋ *Fig. 15.21b*. The mean angle errors of these torque converters are shown in ➋ *Table 15.1*. The maximum mean angle error of these torque converters is 0.339°, and the minimum mean angle error is 0.020°.

For evaluating the proposed feature-based pose-determination approach, a general pose error E is defined to estimate the experimental results of all of the 45 torque converters obtained by 10 model torque converters. Let ε_{ij} represent the mean angle error of torque converter j obtained by model torque converter i. The estimation error E is computed by

$$E = \frac{1}{M \times N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \varepsilon_{ij}, \tag{15.26}$$

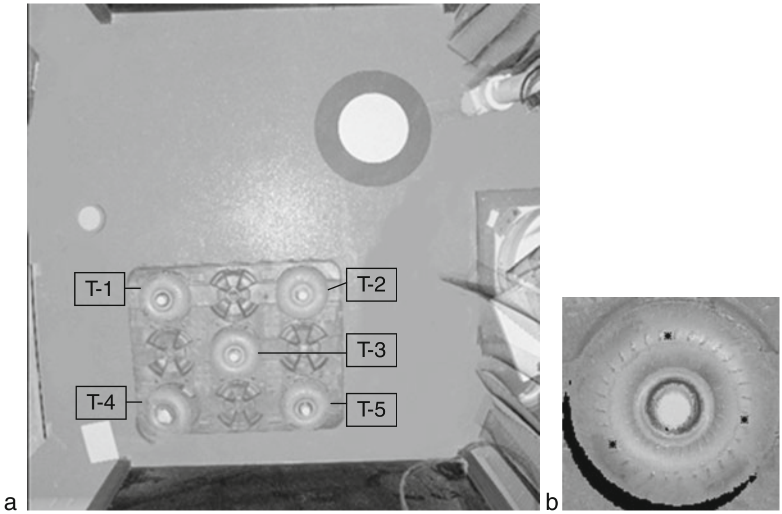


Fig. 15.20

An experimental example displayed in intensity image. (a) A one-level stack with 5 torque converters; (b) a model torque converter with specified feature points

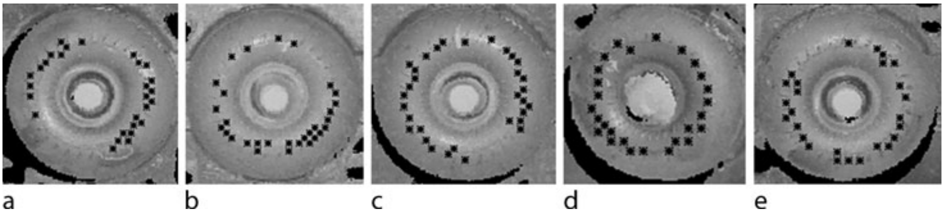


Fig. 15.21

Final matched feature points of the torque converters shown in Fig. 11a obtained by the model torque converter; (a) T-1; (b) T-2; (c) T-3; (d) T-4; (e) T-5

Table 15.1

The mean angle errors (degree) of the torque converters in *Figure 15.21*

Torque converter	T2-1	T2-2	T2-3	T2-4	T2-5
Mean angle error (in degree)	0.020	0.339	0.245	0.297	0.078

where *M* and *N* are the number of model torque converters and test torque converters. In our experiments, the estimation error *E* is 0.263° for the 45 test torque converters and 10 model torque converters.

The experimental results show that 3-D surface features of a local point can be described well using the proposed ADM descriptions in range images. In this automated assembling application, the feature points used to determine the pose information are extracted according to the projected ADM descriptions. Accurate 3-D orientation information of automobile parts is obtained by fitting the feature plane based on the refined feature points.

Robots and machine vision systems are frequently an integral part of industrial automation. The combinations of industrial robots and machine vision systems are typically found in manufacturing process automation. The advent of robotic technology has warranted the strenuous requirement for pick-and-place operations, and they account for majority of repetitive manoeuvres in manufacturing processes. Combined robotic instrumentations with factory assembly controlling and monitoring are key to achieving high productivities in the industry that need less human interventions, such as semiconductor and electronics assembly. In manufacturing processes, particularly in the processes that need human intervention, the success rate is small. Such manufacturing processes can be found typically in automotive assembly lines. In these assembly processes, e.g., transmission parts assembly, the positions and orientations of the parts deviate, frequently, from the positions and orientations defined by the designed procedure. Therefore, an operation process without the control based on the real-time online feedback is not suitable. The real-time online process, such as a precision-feedback controlled and monitored process, requires the additional assistance from a vision system that provides not only two-dimensional grid feature cues, but also three-dimensional orientation cues. The method developed in this study represents a solution to the problems, such as heavy parts pick-and-place, in the assembly processes.

In 3-D image processing, the range data of the environment are collected and analysed for completing a specified vision task. Using the traditional image processing methods, the target parts are segmented from the original range images. The centre position is then computed through the circle Hough transform algorithm. For the 3-D orientation estimation, a 3-D free-form surface descriptor, Angle Distance Map, is introduced. The principal component analysis is applied to make the ADM representation efficient for a large model database. Feature points used to determine the object pose information are matched according to the compressed ADM descriptions by PCA. For accurate pose estimation, the original extracted feature points are refined by an outlier elimination approach. 3-D orientation information is estimated according to the extracted feature points. The experimental results show that accurate object pose is obtained according to the proposed 3-D object pose estimation method.

15.4 Techniques in Range Image Processing, Part 1: Edge Detection

There are numerous approaches in range image processing. The techniques in this aspect are many, and for every imaging attribute description, there are various approaches. We present two approaches: (1) edge detection and (2) region segmentation.

15.4.1 Edge Detection

In range image processing, the 3-D geometrical features of objects are extracted for high-level machine vision tasks. Edge feature is an important 3-D feature in range images. Unlike traditional 2-D intensity image processing, there are primarily three types of edges in range images, and they are step edge which is discontinuous in depth value, roof edge which is discontinuous in surface normals, and curvature edge which is discontinuous in a variety of surface normals. ➲ *Figure 15.22* shows the three types of edges.

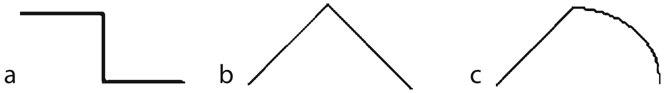


Fig. 15.22

Three types of edge in range images: (a) Discontinuities in depth; (b) discontinuities in surface normals; (c) discontinuities in the variety of neighbourhood surface normals

Surface normal is a first-order derivative feature. This type of local 3-D features is more robust to image noise than the features obtained using higher-order derivatives. The angle between neighbouring surface normals is one of the typical geometrical relationships used to extract 3-D surface feature. In this study, an angle attribute of neighbouring surface normals is defined. An edge detection method in range images is then developed for three different types of edges: step edge, roof edge and curvature edge. The test results on real range images show that our method correctly extracts all three types of edges simultaneously.

15.4.2 Angle Attribute of Neighbouring Surface Normals

In 3-D surface analysis, some geometrical features are obtained through the varieties of local surface normals. In many traditional approaches, such as 2-D intensity image processing, the gradients along X and Ydirections of a range image are used to extract the value discontinuities of surface normals. This approach does not distinguish the orientation variations of surface normals, while the orientation feature of surface normals is also important in 3-D surface analysis. Based on this point of view, an angle attribute of neighbourhood surface normals is proposed as a local surface geometrical feature. ➲ *Figure 15.2* shows a paradigm of some neighbourhood surface normals.

The proposed angle attribute is an angle vector used to describe the local surface feature on the central point. In ➲ *Fig. 15.23*, there are four pairs of surface normals with a centre to be the point of concern. The angle attribute $\vec{\theta}$ is defined as

$$\vec{\theta} = [\theta_0, \ \theta_1, \ \theta_2, \ \theta_3]^T, \tag{15.27}$$

where $\theta_i (i = 0, 1, 2, 3)$ is the angle value between the neighbourhood surface normals along the directions of 0°, 45°, 90° and 135°, as shown in ➲ *Fig. 15.2b*. The value of θ_i is computed by

$$\theta_i = \frac{(\mathrm{Angle}(\vec{n}, \ \vec{n}_i) + \mathrm{Angle}(\vec{n}, \ \vec{n}_{i+4}))}{2}, \ i = 0, \ 1, \ 2, \ 3, \tag{15.28}$$

where $\vec{n}$ is the surface normal of the central point in ➲ *Fig. 15.2a* and $\vec{n}_i$ is the surface normal of the neighbourhood points. Let (f_x, f_y) and (f_{xi}, f_{yi}) be the gradients of $\vec{n}$ and $\vec{n}_i$, and $\mathrm{Angle}(\vec{n}, \ \vec{n}_i)$ is then computed as

$$\mathrm{Angle}(\vec{n}, \ \vec{n}_i) = \arccos\left(\frac{f_x f_{xi} + f_y f_{yi} + 1}{\sqrt{f_x^2 + f_y^2 + 1}\sqrt{f_{xi}^2 + f_{yi}^2 + 1}}\right). \tag{15.29}$$

As shown in ➲ *Fig. 15.23b*, the specific parameter d is the coordinate distance value between the central point and its neighbourhood points. Analytic resolutions are adjusted using different values of parameter d. Obviously, angle features with small d result in a higher resolution for features. On the other hand, a lower resolution is obtained using large d in the

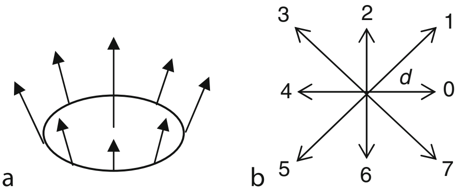


Fig. 15.23
(a) A paradigm of local surface normals; (b) a paradigm of computing the angle attribute of neighbourhood surface normals

angle feature. In normal cases, angle attributes with a larger *d* are more tolerant to noise in an image, and they are applied to extract roof edges that represent a smooth change between two regions of interest.

In this study, for the different types of edges in range images, an edge detection method based on the angle attribute is developed. According to the angle attribute, a Mean Angle Feature (MAF) $\bar{\theta}$ is defined as

$$\bar{\theta} = \frac{\sum_{i=0}^{3} \theta_i}{4}, \tag{15.30}$$

where θ_i, $i = 0, 1, 2, 3$ are computed using (15.28).

The Mean Angle Feature is applied to extract the three types of edges simultaneously in a typical range image: step edge, roof edge and curvature edge. For step edges, ideally there is no real surface normal at the step edge. In image processing, the value of the surface normal at this local area is still obtained by common approximation methods. Since the value of the approximated surface normal is random, the value of an MAF at this local area is also distinguishable. For roof edges, there is an orientation variation at the edges between two neighbourhood surface patches. The MAF describes the surface orientation variation along different directions. Therefore the orientation variation is extracted using an MAF operator. For curvature edges, the surface orientation varies not only within the neighbourhood patches but also at the curvature edge between the patches. Nevertheless, the characteristics of the orientation variation are consistent within one patch, and they are different from the adjacent patches. According to the differential values of the MAF, the curvature edges can be extracted through detecting the changes of the characteristics of the orientation variation. Based on the nature of edges in range images, an edge detection method is developed for extracting the three types of edges in range images simultaneously.

15.4.3 Edge Detection in Range Images

In this edge detection method, different types of discontinuities are detected based on the MAF of a local surface. According to (15.29), the angle values are computed using the gradients of surface normals. In traditional methods, the gradients are computed using the first-order partial derivative operators. For real range images, due to the effect of geometrical distortion, computed gradients do not represent the true geometry. Especially for the images with curved surfaces, the gradients cannot be correctly approximated using traditional gradient computing methods. In our edge detection method, a surface fitting algorithm is applied to obtain the accurate gradients in real range images. The estimated surface function $\widehat{f}(x,\ y)$ is described as

$$\widehat{f}(x,\ y) = a_{20}x^2 + a_{02}y^2 + a_{11}xy + a_{10}x + a_{01}y + a_{00}, \tag{15.31}$$

where a's are the parameters of a geometrical function. The subscript of a denotes the polynomial of that parameter. The gradient values are then given according to

$$f_x = a_{10},\quad f_y = a_{01}. \tag{15.32}$$

Prior to calculating the surface properties, a smoothing process is applied on the range images, which include detector noise, quantization and calibration errors. A Gaussian smoothing method is applied in this edge detection method. In a Gaussian smoothing, two-dimensional image data are convolved with the rotation-invariant Gaussian filter

$$g(x,y) = e^{-(x^2+y^2)/2\sigma^2} \tag{15.33}$$

where (x,y) is the coordinate of each point in the image and σ^2 is the variance. Alternatively, Gaussian smoothing is approximated through a convolution with a 3 × 3 operator

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 12 & 2 \\ 1 & 2 & 1 \end{bmatrix}. \tag{15.34}$$

After the gradients are obtained, the MAF of each point in a range image is computed according to (2) and (4). The result of an MAF map of range images is then generated. An initial edge result based on the MAF map is generated according to

$$\Delta\bar{\theta} = \left|\frac{\partial\bar{\theta}}{\partial x}\right| + \left|\frac{\partial\bar{\theta}}{\partial y}\right|. \tag{15.35}$$

A binary edge map of range image is obtained based on the initial edge result. The different types of edges in a range image are detected simultaneously according to the edge map.

In the edge map, parameter d of the MAF operator causes a double-edge effect at step edges in range images. There is a thin gap in the middle of the extracted edge area. A common expansion and thinning process is applied to eliminate this gap. In a binary edge map, the gap area is filled by expanding the initially extracted edge points. In this edge detection approach, for a non-edge point and its adjacent area, typically 5 × 5, if the pixel number of edge point within the neighbourhood area is larger than a predefined threshold, the non-edge point is changed into an edge point in the binary edge map. After the expansion process for all pixels in the binary map, a thinning process is realized using a normal skeleton algorithm. A distance attribute of each edge point is defined at first. For each edge point, the nearest non-edge point from it in all directions is detected. The distance between the nearest non-edge point and the concerned edge point is defined as the distance attribute of the edge point. The edge points with the local maximum distance attribute are considered as skeleton points. The thinning process is realized by eliminating the non-skeleton points.

The edge detection method for range images consists of the following steps:

1. Apply a smooth operator in (➋ 15.34) on original range image.
2. According to (➋ 15.31) and (➋ 15.32), compute the gradients along X and Y directions for each point in range image.

3. Based on (➋ 15.28)–(➋ 15.30), extract the MAF for each point in range image and generate an MAF map for range image.
4. According to the MAF map, the initial edges are detected based on (➋ 15.35).
5. Apply the expansion and thinning process on the edge map to obtain the edge as the final result of the method.

15.4.4 Experimental Results and Analysis

The experimental range images in this study are acquired using a scanner-based infrared laser radar imager. Noise is apparent in the acquired range images. The smoothing process and surface fitting method are applied to obtain accurate surface normal. To extract smooth roof edges and obtain a fine edge resolution, the value range of parameter *d* in the MAF computation is selected from 3 to 5. Based on the ground truth rule, the extracted edges are matched against the original range images. The results show that the edges match well with the gradient changes in the original range data.

➋ *Figure 15.3* shows the experimental results of a polyhedral object. ➋ *Figure 15.24a* shows the original range image with a size of 404 × 122 pixels. The corresponding intensity image and

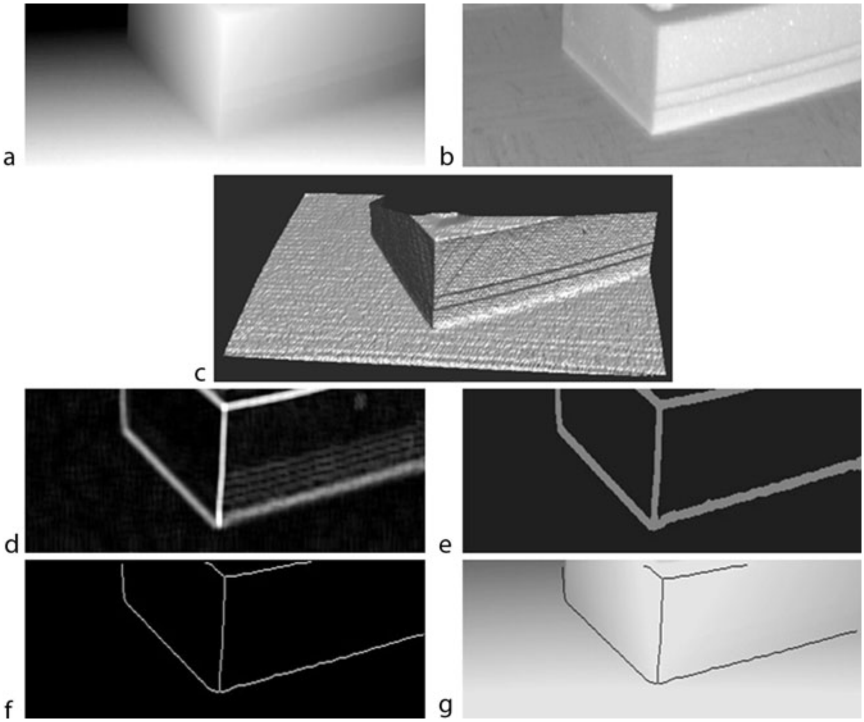


Fig. 15.24

(a) The original range image of a packaging part; (b) the corresponding intensity image; (c) the 3-D display for showing image noise; (d) the MAF map of (a); (e) initial edge result; (f) the final edge result after expansion and thinning processing of (e); (g) a superimposed image of (a) and (f) to show edge localization accuracy

a 3-D display using a light mode are shown in *Fig. 15.24b, c*. There are several types of edges in this range image, such as step edges and roof edges of concave and convex types. The MAF map is shown in *Fig. 15.24d*. The surface angle features can be seen in the MAF map. *Figure 15.24e* shows the initial edges extracted from the MAF map. After expansion and thinning processing, the one-pixel-width edges in range image are extracted and shown in *Fig. 15.24f*. To verify the localization of the edge points, the edge image in *Fig. 15.24f* is superimposed onto the original range image in *Fig. 15.24a* to produce an accuracy verification as shown in *Fig. 15.24g*. The edge points are checked based on the ground truth and they are found to be accurate.

Figure 15.25 shows the results from applying our method to a range image of a cylinder part. *Figure 15.25a, b* show the original range image and its corresponding intensity image with a size of 228 × 165 pixels. A 3-D display is shown in *Fig. 15.25c*. Noise in the image is apparent. There are step edges between the flat and curved surface in this image. *Figure 15.25d* shows the 3-D surface angle features extracted according to the MAF map. The initial edge map is shown in *Fig. 15.25e*. The double-edge effect is reflected in the initial edge result. *Figure 15.25f* shows the final edge result after the expansion and thinning process. *Figure 15.25g* shows the edge image being superimposed on the original range image. Although some image processes, e.g., smoothing process and expansion and thinning process are applied in this edge detection method, edge localization is found accurate for all three types of edges.

The result of detecting edges from a curved industrial part is shown in *Fig. 15.26*. The original range and intensity images with a size of 209 × 173 pixels are shown in *Fig. 15.26a, b*, respectively. *Figure 15.26c* shows the same mode 3-D display. In this range image, there are three types of edges: step edge, roof edge and the edge between the surfaces with different curvatures. The 3-D surface features are described according to the MAF map as shown in *Fig. 15.26d*. *Figure 15.26e* shows the initial edge map detected from the MAF map. The double-edge effect can also be seen in the initial edge result. The final result with different types of edges is shown in *Fig. 15.26f*. *Figure 15.26g* shows the superimposed image of the edges and the original. The experimental results show that the three types of edges in a range image are correctly detected simultaneously using the MAF of neighbouring surface normals.

In correspondence, an angle attribute of neighbourhood surface normals is proposed for extracting the 3-D geometrical surface features. Based on the geometrical features of this angle attribute, an edge detection method is developed for extracting the three types of edges in range images. Using the MAF map of an original range image, the different types of edges are distinguished and detected simultaneously. To eliminate the effect of noise in the real range images, some image processes are applied to extract the final edge results. The experimental results show that the proposed method produces the edges matched to ground truth verification.

15.5 Techniques in Range Image Processing, Part 2: Region Segmentation

Segmentation is an assignment of pixels into one of many disjoint sets such that the pixels in each set share a common property. Currently, many machine vision techniques use range images to obtain useful descriptions of 3-D scenes. Segmentation has been most often used in range image analysis and recognition as it helps partition an image array into low-level entities. In general, segmentation methods for range images are classified into two categories. One is based on region features, and the other is based on edge information.

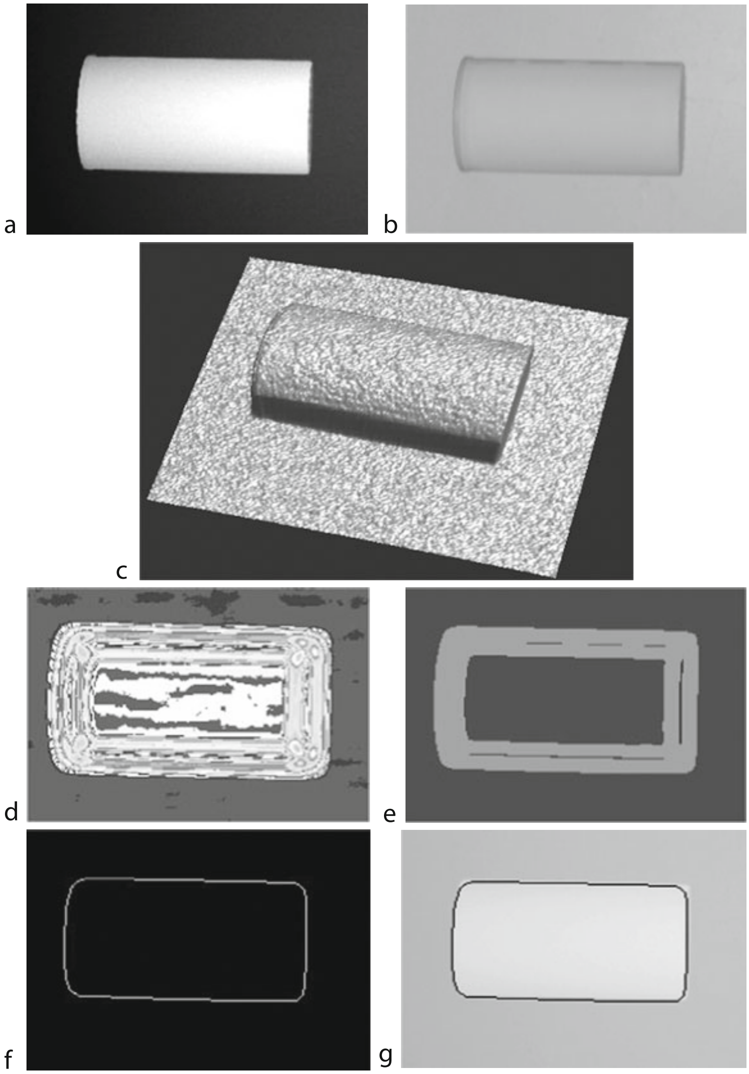
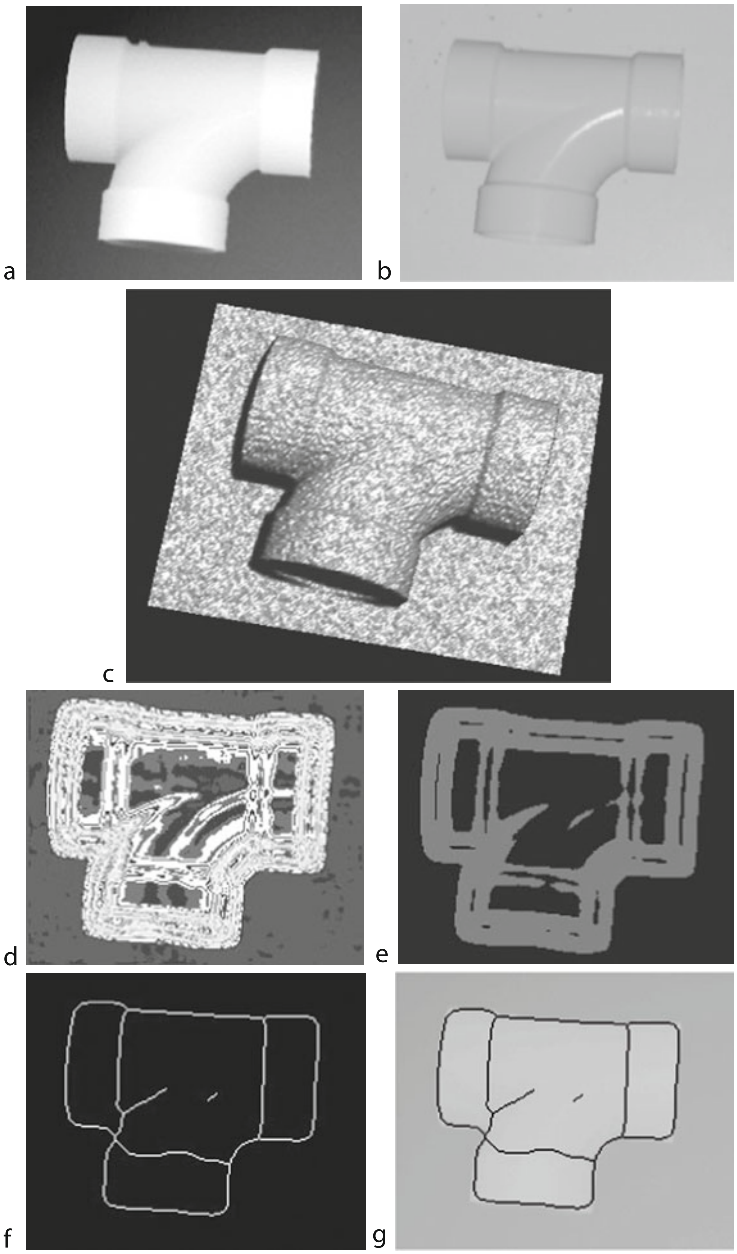


Fig. 15.25
(a) The original range image of a cylinder part; (b) the corresponding intensity image; (c) the 3-D display for showing image noise; (d) the MAF map of (a); (e) initial edge result; (f) the final edge result after expansion and thinning processing of (e); (g) a superimposed image of (a) and (f) to show edge localization accuracy

In region-based segmentation methods, the pixels having similar properties are grouped together. The techniques of surface fitting and feature clustering are two commonly applied to analyse the region properties for range image segmentation. Region growing techniques select a set of seed regions first and grow the seed regions by iterative merging of adjacent regions with similar properties.



■ **Fig. 15.26**

(a) The original range image of an industrial part; (b) the corresponding intensity image; (c) the 3-D display for showing image noise; (d) the MAF map of (a); (e) initial edge result; (f) the final edge result after expansion and thinning processing of (e); (g) A superimposed image of (a) and (f) to show edge localization accuracy

In this section, a segmentation approach based on two-dimensional gradient histogram is proposed and the images are taken from a range sensor. The gradients along directions of x and y coordinates are applied in the segmentation method. According to the distribution of the gradients histogram, a four-neighbourhood region growing algorithm is developed to achieve the preliminary planar patch segmentation based on a proposed grouping criterion. The gradient features are derived from the first-order derivatives. Compared with the second-order derivative, typically used in some methods, the first-order derivatives are insensitive to noise and easy to compute. Since some pixels may not be grouped into correct regions due to noise effect, a merge process is applied on the first-round segmentation results. The selection of the segmentation parameters is also discussed in this study. The tests show that this algorithm can be used to segment polyhedral objects in range images and it is less sensitive to noise and computationally efficient.

15.5.1 Gradient Based Region Growing

It is well known that the higher-order derivative features are more sensitive to noise than lower-order derivative features. To real range images, the results usually do not meet the expectation from theoretical aspects because of noise effects and quantization errors. Therefore, the first-order derivatives are used for the segmentation in this study. The segmentation features are one-dimensional slice gradients in two coordinate directions. For a given surface $\vec{r} = (x,\ y,\ f(x,\ y)),\ f(x,\ y) : (x,\ y) \in Z \times Z \rightarrow \Re$, according to the theory of differential geometry, the unit surface normal $\vec{n}$ of a point on a patch is given by

$$\vec{n} = \frac{1}{\sqrt{r_x^2 + r_y^2 + 1}} (-r_x, -r_y,\ 1) \tag{15.36}$$

where r_x and r_y are the partial derivatives of $\vec{r}$ with respect to x and y. The surface normal is uniquely determined by the two gradients along the two perpendicular axes.

In this gradient-based method, a gradient histogram is defined in a 3-D parameter space for the polyhedral segmentation. As shown in ➲ *Fig. 15.1*, x-axis represents f_x, y-axis represents f_y and z-axis represents the number of the pixels with the same f_x and f_y. For a given surface planar patch of a range image, it is assumed that the noise is uniform and Gaussian. Thus, the distribution of pixel numbers with respect to the two range gradients f_x and f_y is Gaussian. In the feature space, each peak corresponds to a group of regions with the same surface normal and the top of each peak corresponds to the mean gradient values of each region. A segmentation of planar surfaces is obtained by partitioning peaks in the gradient histogram space (➲ *Fig. 15.27*).

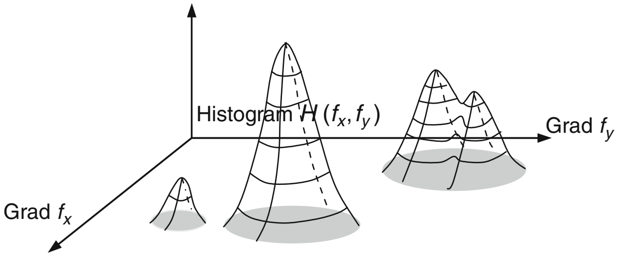


Fig. 15.27
The gradient histogram space

In the proposed region growing algorithm, two neighbouring pixels are grouped into the same patch if their surface properties, f_x and f_y, belong to the same Gaussian distribution in the gradient histogram space. A peak point searching algorithm is developed through iteratively searching the local maxima in the gradient histogram space.

For two neighbourhood points P_1 and P_2, with gradients (f_{x1}, f_{y1}) and (f_{x2}, f_{y2}), their corresponding histogram points, $H\ (f_{x1}, f_{y1})$ and $H\ (f_{x2}, f_{y2})$, are shown in *Fig. 15.28*. A grouping criterion is used: if $H\ (f_{x1}, f_{y1})$ and $H\ (f_{x2}, f_{y2})$ share the same peak area in the feature space, P_1 and P_2 are grouped into the same region; otherwise P_1 and P_2 belong to different regions.

A coarse segmentation result is obtained by applying a four-neighbourhood expanding grouping algorithm based on the grouping criterion. The four-neighbourhood expanding algorithm is an iterative grouping method beginning from a seed pixel. The seed pixels are the ungrouped pixels in range image scanned one by one. For each area grouping operation, the grouping algorithm expands in four directions from a seed pixel of this region. The ungrouped pixel in each direction is compared with the seed pixel according to the grouping criterion. The new grouped pixels are added to this region, and saved as new seed pixels for the future grouping operations. This iterative grouping algorithm terminates when there is no new seed pixel grouped into this region.

During grouping process, some points may be grouped into incorrect regions due to noise or other geometrical distortion. In this region growing approach, if the total pixel number of a grouped region is smaller than the threshold R_{Num}, then that region is considered as dominated by noise or geometrical distortion. For incorrect coarse segmentation results, a merge process between a noise region and its right neighbour regions is developed. For two adjacent regions R_1 and R_2, while the pixel number of region R_1 is smaller than R_{Num}, and the pixel number of region R_2 is larger than R_{Num}, a merge criterion is proposed: if $\left|\bar{f}_{x1} - \bar{f}_{x2}\right| + \left|\bar{f}_{y1} - \bar{f}_{y2}\right| < D$, R_1 and R_2 are merged into one region; otherwise R_1 and R_2 cannot be merged into one region. The variables $\bar{f}_{x1}$, $\bar{f}_{x2}$, $\bar{f}_{y1}$ and $\bar{f}_{y2}$ are the mean gradients of each region, and D is a threshold to gradient values. The selection of thresholds R_{Num} and D is discussed in Sect. 15.5.3.

Prior to calculating the surface properties, a smoothing process is applied on real range images, which include detector noise, quantization and calibration errors. A Gaussian smoothing method by Brady and Ronce [24] is applied in this segmentation method. In a Gaussian smoothing, two-dimensional image data are convolved with the rotation-invariant Gaussian filter

$$g(x, y) = e^{-(x^2+y^2)/2\sigma^2} \tag{15.37}$$

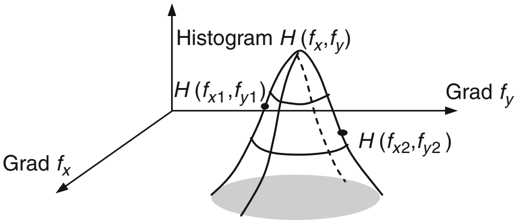


Fig. 15.28
A paradigm for grouping criterion

where (x, y) is the coordinate of each point in the image and σ^2 is the variance. Alternatively, Gaussian smoothing is approximated through a convolution with a 3 × 3 operator

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 12 & 2 \\ 1 & 2 & 1 \end{bmatrix}. \tag{15.38}$$

The gradients of each pixel are computed, and the histogram is then obtained according to the gradients. A coarse segmentation result is obtained using a four-neighbourhood iterative expanding algorithm according to the grouping criterion. For those grouped regions dominated by noise or geometrical distortion, a merge process is applied based on the merge criterion. In this region growing approach, the cluster region number is not required a priori. The procedure is described by the following steps:

1. The original range image is smoothed using the Gaussian mask shown in (15.38). The gradients of each pixel in the smoothed range image are computed, and the gradient histogram is obtained.
2. The segmentation result map Q is initialized by setting each point with a non-processing value -1, also setting the current region label as 0, and setting the current scanning position in Q as $i = 0$ and $j = 0$.
3. Scanning and checking the map pixels $\{(i, j)\}$ one by one.
4. If $Q(i, j) \neq -1$ go to Step 6.
5. The pixel (i, j) is taken as a seed of a new region. The current region label adds one. According to the gradients of this point in the range image, this seed grows along its four neighbouring directions based on the grouping criterion. The grouped points are taken as new growing point candidates. This four neighbouring expansion process is iterated until no new point is grouped into this region. At each grouping process, the value of the grouped pixel in the segmentation map Q is set as the current region label. The attributes of this region, such as region pixel number and mean gradient $\bar{f}$, are also modified.
6. If the scanning process is not over, go to Step 3.
7. The regions with its pixel number smaller than threshold R_{Num} are processed according to the merge criterion.

15.5.2 Threshold Selection

There are two thresholds, R_{Num} and D, used in this gradient-based region growing method. *Figure 15.29* shows a paradigm of selecting threshold R_{Num}. In this gradient histogram, the pixels whose gradient histogram values are larger than the R_{Num} are taken as effective region points; otherwise they are taken as noise points. When R_{Num} is selected to be smaller than a representative value for target regions, some pixels due to noise and some small blocks due to distortion are then taken as valid regions, thus causing under-thresholding. When R_{Num} is larger than necessary, some small but valid regions are treated as noise, thus causing over-thresholding. In this study, an area ratio of the noise points to the whole points in a range image is applied to decide the threshold R_{Num}. Normally the area ratio value depends on the areas

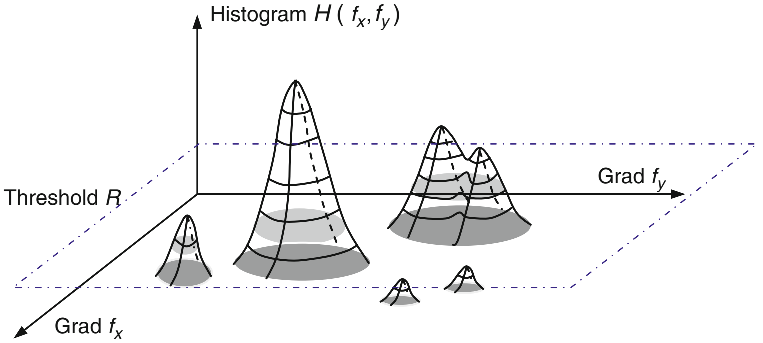


Fig. 15.29
A paradigm for selecting threshold R_{Num}

such as noise area, geometrical distortion area and jump edges. According to a candidate threshold R, the area ratio value, $f(R)$, is given by

$$f(R) = \frac{1}{Total}\sum_{f_x}\sum_{f_y} H(f_x, f_y), \quad H(f_x, f_y) < R, \tag{15.39}$$

where R is a candidate threshold value in the gradient feature space, and *Total* is the total pixel number of the processed range image. For a predefined area ratio value p, threshold R_{Num} applied in this segmentation method is derived by

$$f(R_{\text{Num}}) = p. \tag{15.40}$$

Figure 15.29 shows a paradigm of selecting threshold R_{Num}. The threshold R_{Num} is obtained by applying an iterative searching algorithm. The candidate threshold R is increased one by one from zero. For each candidate threshold R, the area ratio value is computed according to (15.39). If (15.40) is satisfied, the candidate R is taken as the threshold R_{Num}. The area ratio value p in (15.40) is a predefined parameter. The selection of value p depends on the contents of range images. If range images contain a lot of noise, distortion and edge areas, the noise ratio is large. Therefore, parameter p is selected as a large value. Otherwise, parameter p is selected as a small value. For normal-range images, parameter p is selected as 5% for the noise area inside the images.

Threshold D is another parameter in the merge process. After the coarse segmentation, for those regions whose region pixel number is smaller than threshold R_{Num}, a merge process is applied between the small regions and their adjacent larger regions whose region pixel number is larger than threshold R_{Num}. As defined in merge criterion, two adjacent regions are merged if the difference between their mean gradients is smaller than threshold D. The merge process is executed region by region. For two adjacent regions, the value of threshold D is decided by the gradient variation of the larger region within the pair of merging regions. Parameter σ_f is defined as the mean difference of gradients of the larger region, and is computed by

$$\sigma_f = \frac{1}{N}\sum_{i=0}^{N}\left[|f_{xi} - \bar{f}_x| + |f_{yi} - \bar{f}_y|\right], \tag{15.41}$$

where $\bar{f}_x, \bar{f}_y$ are the mean gradient values of the region, and N is the pixel number of this region, and f_{xi} and f_{yi} are the gradients on x-axis and y-axis, respectively, of pixel i. Based on the gradient difference parameter σ_f, threshold D is given by

$$D = 2\sigma_f. \tag{15.42}$$

15.5.3 Analysis

The range images used in this study are acquired using a laser range imager. There is noise in the images. A Gaussian smoothing process, defined in (➋ 15.37) and (➋ 15.38), is applied to obtain accurate gradient properties. The area ratio value of parameter p depends on the areas, such as geometrical distortion area and jump edges. In these experiments, parameter p is selected as 5%. Based on the ground truth rule, the extracted segmentation results are matched against the original range images. The experiments show that this gradient-based segmentation method generates good results for understanding 3-D polyhedral objects.

➋ *Figure 15.30a* is an original intensity image of wall and ground. Its size is 166 × 143 pixels. ➋ *Figure 15.30b* is the corresponding range image. Image noise is apparent in a 3-D display mode as shown in ➋ *Fig. 15.30c*. ➋ *Figure 15.30d* shows the gradient histogram of the smoothed range image. There are two major peaks corresponding to the directions of the wall and the ground regions. Some areas with small histogram values represent noise effect and geometry distortion. Through our region grouping approach, the neighbourhood pixels within the vicinity of the same gradient peak are grouped together. The region grouping result without the merge process is shown in ➋ *Fig. 15.30e*. Most of the pixels are grouped correctly. However, three small patches are grouped incorrectly because of noise or geometry distortion. ➋ *Figure 15.30f* shows the final result of applying the merge process.

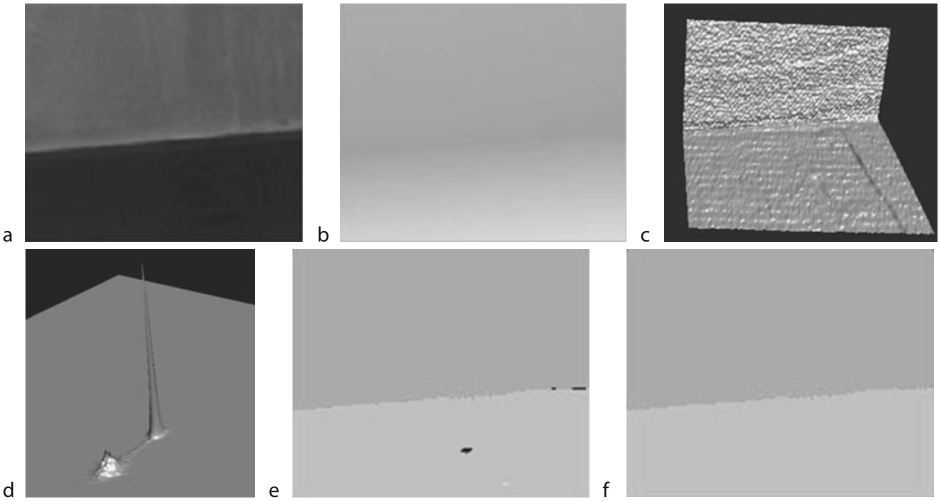


Fig. 15.30
A segmentation sample of wall and ground. (a) Original intensity image; (b) correspondent range image of (a); (c) 3-D display for showing image noise; (d) gradient feature space; (e) the result of initial region grouping; (f) the final segmentation result of applying the merge process

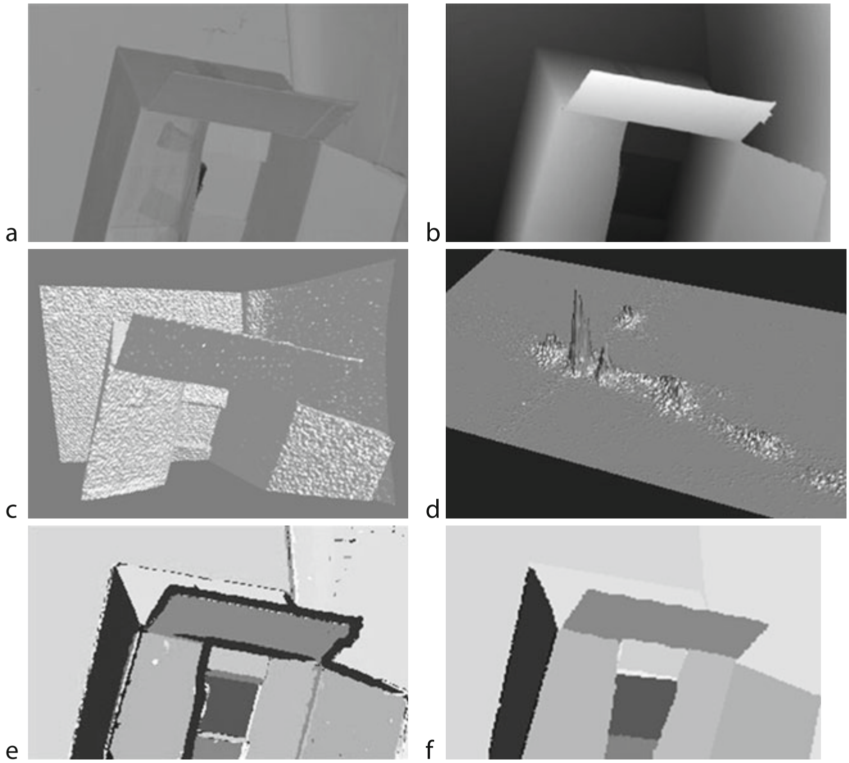


Fig. 15.31

A segmentation sample of a box. (a) original intensity image; (b) correspondent range image of (a); (c) 3-D display for showing image noise; (d) gradient feature space; (e) the result of initial region grouping; (f) the final segmentation result of applying the merge process

Figure 15.31 shows another segmentation process. *Figure 15.31a, b* shows the original intensity and range images of a box. The image size is 298 × 187 pixels. A 3-D display mode shows the range image in *Fig. 15.31c*. Image noise can be seen clearly. *Figure 15.31d* shows the gradient histogram. The main peak areas and some random distribution areas due to the image noise and jump edges can be seen. The initial region grouping result is shown in *Fig. 15.31e*. Some small patches and jump edges are grouped incorrectly. These regions are merged into their neighbourhood regions using the merge process. *Figure 15.31f* shows the final segmentation result.

Figure 15.32a shows an original intensity image of a packaging box. The image size is 364 × 248 pixels. *Figure 15.32b* shows the corresponding range image. A 3-D display mode is shown in *Fig. 15.32c*. *Figure 15.32d* shows the feature space of the range image. Three major peaks corresponding to the three directions of the planar regions. The region grouping result of the coarse segmentation is shown in *Fig. 15.32e*. *Figure 15.32f* shows the final segmentation result after applying the merge process. The noise area and jump edges in range images have been correctly grouped through the proposed gradient-based region grouping method.

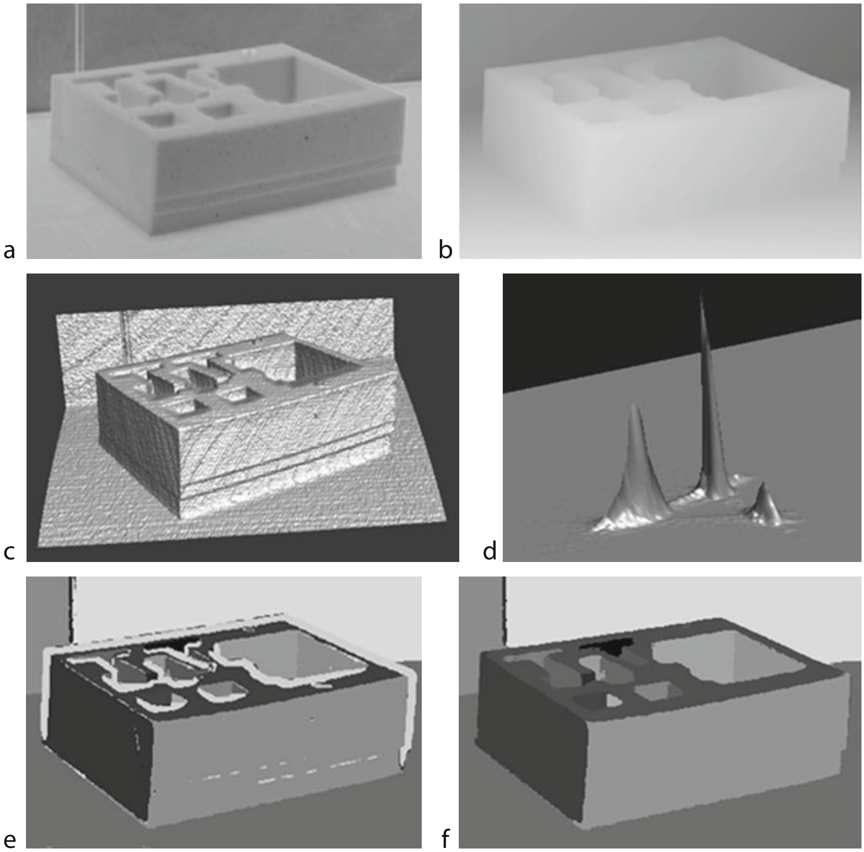


Fig. 15.32

A segmentation sample of a part. (a) original intensity image; (b) correspondent range image of (a); (c) 3-D display for showing image noise; (d) gradient feature space; (e) the result of initial region grouping; (f) the final segmentation result of applying the merge process

In this approach, two thresholds used in the merge process are derived from (15.39) to (15.42). The experimental results show that the thresholds have been selected properly. *Figure 15.33* shows some segmentation results of the object in *Fig. 15.32b*, with different threshold values. It is known that if R_{Num} is set too small, some small noise regions are taken as real separate regions and then lead to an incorrect segmentation as shown in *Fig. 15.33a*. If R_{Num} is set too large, some valid regions whose region pixel number is smaller than R_{Num} are taken as noise regions. If these small regions and their adjacent large regions satisfy the merge criterion, these valid regions are merged incorrectly into the adjacent large regions. As shown in *Fig. 15.33b*, two planes of the hole of the lower-left corner of the box surface are merged into one region. The different values of the threshold D also affect the final segmentation result. If D is selected too large, some real separate neighbouring regions with smaller gradient difference will be merged together. There is no such instance in this image. If D is selected too small, some small regions cannot be grouped correctly into their neighbourhood because of their larger

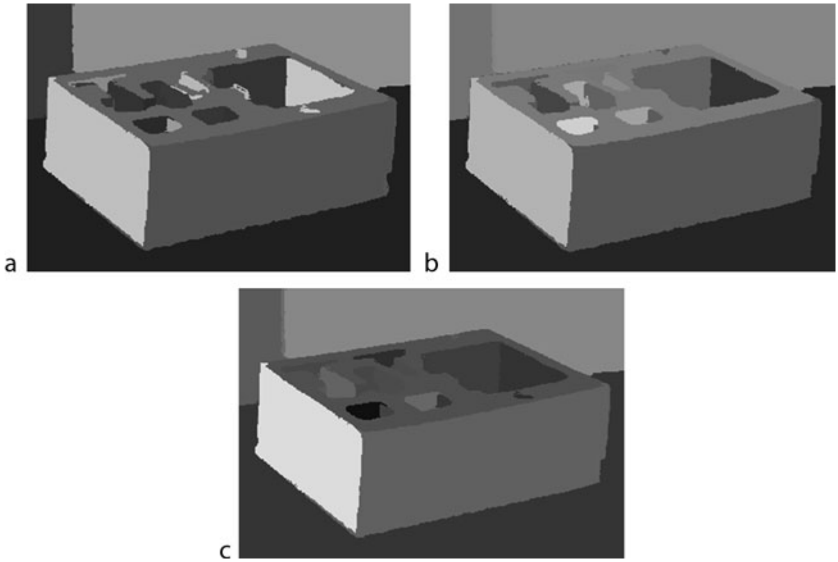


Fig. 15.33
Some incorrect segmentation results with different threshold selections: (a) the segmentation result with a smaller RNum; (b) the segmentation result with a larger RNum; and (c) the segmentation result with a smaller D

gradient difference. As shown in *Fig. 15.33c*, compared with the correct segmentation results shown in *Fig. 15.32f*, there is an incorrectly merged region on the top surface of the box.

Compared with other popular segmentation methods, the above discussed approach is much simpler and more robust because only the first-order features are used. The gradients along *x* and *y* directions and their histogram are applied as the segmentation features. The problem of surface segmentation becomes a problem of points clustering in a feature space. A grouping criterion is proposed for the region growing based on the gradient histogram. A four-neighbourhood iterative expanding algorithm is developed in region grouping. The regions with arbitrary shape can be grown in one expanding process. For a certain type of regions in range images, such as noise, geometrical distortion area and jump edges, a merge process is applied to the initial region growing results according to a proposed merge criterion. The experiments show that the proposed algorithm generates good results for understanding polyhedral objects in range images.

References

1. Anthes J, Carcia P, Pierce J, Dressendorfer P (1993) Non-scanned LADAR imaging and applications. In: Applied laser radar technology. SPIE, vol 1936, pp 11–22
2. Jelalian AJ (1992) Laser radar systems. Artech House, Norwood
3. Scott M (1990) Range imaging laser radar. United States Patent no. 4,935,616, 19 June 1990
4. Scott M, Garcia P (1988) Use of predetection mixing of intensity modulated optical beams. Appl Opt 27:5119
5. Brophy C (1990) Effect of intensity error correlation on the computed phase of phase-shifting interferometry. J Opt Soc Am A 7(4):537–541
6. Freischlad K, Koliopoulos C (1990) Fourier description of digital phase-measuring interferometry. J Opt Soc Am A 7(4):542–551

7. Morgan C (1982) Least-squares estimation in phase-measurement interferometry. Opt Lett 7:368–370
8. Bruning J, Herriot D, Gallagher J, Rosenfeld D, White A, Brangaccio D (1974) Digital wavefront measuring interferometer for testing optical surfaces and lenses. Appl Opt 13:2693–2703
9. Stann B, Ruff W, Robinson D, Potter W, Sarama S, Giza M, Simon D, Frankel S, Sztankay Z (1999) Scannerless imaging ladar using a laser diode transmitter and FM/cw radar principles. In: Laser radar technology and applications IV. Proceedings of SPIE, vol 3707
10. Burns H, Yun ST, Kimmet J, Keltos ML (1999) Compact, multichannel imaging-laser radar receiver. In: Laser radar technology and applications IV. Proceedings of SPIE, vol 3707
11. Arman F, Aggarwal J (1993) Model-based object recognition in dense-range image – a review. ACM Comput Surv 25:5–43
12. Besl P (1989) Active optical range imaging. In: Sanz JLC (ed) Advances in machine vision. Springer, New York
13. Bolle R, Vemuri BC (1991) On three-dimensional surface reconstruction methods. IEEE Trans Pattern Anal Mach Intell 13:1–13
14. Besl P, Jain R (1985) Three-dimensional object recognition. Comput Surv 17:75–145
15. Pentland A (1987) Recognition by parts. In: Proceedings of the 1st international conference in computer vision, London, England, pp 612–620
16. Dickinson S, Metaxas D (1997) The role of model-based segmentation in the recovery of volumetric parts from range data. IEEE Trans Pattern Anal Mach Intell 19:259–267
17. Dickinson S, Metaxas D, Pentland A (1994) Constrained recovery of deformable models from range data. In: Proceedings of 2nd international workshop visual from, Capri, May 1994
18. Dorai C, Jain A (1997) COSMOS-a representation scheme for 3D free-form objects. IEEE Trans Pattern Anal Mach Intell 19:1115–1130
19. Johnson A, Hebert M (1999) Using spin images for efficient object recognition in cluttered 3D scenes. IEEE Trans Pattern Anal Mach Intell 21:433–449
20. Sim R, Dudek G (1999) Learning visual landmarks for pose estimation. In: Proceedings of the 1999 IEEE international conference on robotics and automation, Detroit, May 1999, pp 1972–1978
21. Burtnyk N, Greenspan M (1994) Signature search method for 3-D pose refinement with range data. In: Proceedings of the 1994 IEEE international conference on multisensor fusion and integration for intelligent systems, Las Vegas, Oct 1994, pp 312–319
22. Stark K, Fuchs S (1996) A method for tracking the pose of known 3-D objects based on an active contour model. In: Proceedings of the 13th international conference on pattern recognition, Vienna, Austria, Aug 1996, pp 905–909
23. Crowley J, Schiele B (1998) Position estimation using principal components of range data. In: Proceedings of the 1998 IEEE international conference on robotics and automation, Leuven, Belgium, May 1998, pp 3121–3128
24. Fan T, Medioni G, Nevatia R (1987) Segmented descriptions of 3-D surface. IEEE Int J Robot Automat 3(6):527–538
25. Gruss A, Kanade T (1990) A fast lightstripe range finding system with smart VLSI sensor. In: Machine vision for 3-D scenes. Academic, San Deigo, pp 381–397
26. Fu K, Gonzalez R, Lee C (1987) Robotics: control, sensing, vision, and intelligence. McGraw-Hill, New York
27. Besl P, Jain R (1988) Segmentation through variable-order surface fitting. IEEE Trans Pattern Anal Mach Intell 10:167–192
28. Kang S, Ikeuchi K (1993) The complex EGI: a new representation for 3-D pose determination. IEEE Trans Pattern Anal Mach Intell 15:707–721
29. Liu X, Wang D (1999) Range image segmentation using a relaxation oscillator network. IEEE Trans Neural Netw 10:564–573
30. Lee K et al (1998) Robust adaptive segmentation of range images. IEEE Trans Pattern Anal Mach Intell 20:200–205
31. Inokuchi S, Nita T, Matsuday F, Sakurai Y (1982) A three-dimensional edge-region operator for range pictures. In: Proceedings of 6th international conference on pattern recognition, Munich, West Germany, pp 918–920
32. Lim A, Teoh E, Mital D (1994) A hybrid method for range image segmentation. J Math Imaging Vis 4:69–80
33. Zhao D, Zhang X (1997) Range data based object surface segmentation via edges and critical points. IEEE Trans Image Process 6:826–830
34. Baccar M, Gee L, Abidi M (1996) Segmentation of range images via data fusion and morphological watersheds. Pattern Recognit 29:1673–1687
35. Deng S, Yuan B (1995) Range image segmentation based on mathematical morphology. ACTA Electronica Sinica 23:6–9
36. Jiang X, Bunke H (1999) Edge detection in range images based on scan line approximation. Comput Vis Image Underst 73:183–199
37. Amano T, Hiura S, Inokuchi S (1996) Eigen space approach for a pose detection with range images. In: Proceedings of ICPR'96, pp 622–627
38. Li S, Zhao D (2003) Gradient-based polyhedral segmentation for range images. Pattern Recognit Lett 24(12):2069–2077

39. Zaharia T, Preteux F (2004) 3D versus 2D/3D shape descriptors: a comparative study. Image Process Algorithms Syst III Proc SPIE 5298:70–81
40. Flynn P, Jain A (1989) On reliable curvature estimation. In: Proceedings of CVPR, pp 110–116
41. Krishnapuram R, Gupta S (1992) Morphological methods for detection and classification of edges in range images. J Math Imaging Vis 2:351–375
42. Brady M, Ronce J, Yullie A, Asada H (1985) Describing surfaces. Comput Vis Graph Image Process 32:1–28
43. Wani M, Batchelor B (1994) Edge-region-based segmentation of range images. IEEE Trans Pattern Anal Mach Intell 16(3):314–319
44. Li S, Zhao D (2001) Three-dimensional range data interpolation using B-spline surface fitting. J Electron Imaging 10(1):268–273

16 Colour Recognition

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_16,

Abstract: Colour is an important feature of many industrial artefacts. Moreover, a very large proportion of packaging materials are coloured. The inspection of coloured objects is therefore of considerable importance. In addition, natural products and raw materials are often inspected, graded or sorted on the basis of their colours. As so often happens in Machine Vision, the prime requirement is for verification rather than unassisted recognition. Sometimes the range of allowed colours can be specified quite precisely. On other occasions, it has to be learned from carefully selected samples. Our scope for flexibility is limited by the need to use commercially available image sensors. These almost invariably provide RGB output signals. We shall therefore concentrate primarily on pixel-by-pixel recognition of colours based on RGB data. Several techniques exist, including "brute force" methods that rely on standard Pattern Recognition algorithms, while others are based more firmly on Colour Science. Multispectral data allows a more refined approach to colour recognition than is possible with RGB. It requires more sophisticated image acquisition methods but can use well-established Pattern Recognition algorithms. Our goal is to provide image segmentation techniques that can be used to identify regions according to "standard" colours ("red," "yellow," "green," etc.) and colours that are specific to one particular situation (i.e., product or factory). A colour recognition program generates one or more binary images that can then be measured using the techniques described in ➋ Chaps. 14 and ➋ 18–20. In this sense, colour recognition serves simply as a means of segmenting an image.

16.1 Why We Need to Recognise Colours

Colour recognition is just one step in the bigger process of product inspection. To begin, let us consider a typical and surprisingly important task: inspecting a simple package. Most low-value products and many high-values ones too are packaged with labels printed in a few block colours. (That is, large areas of constant colour with no significant amount of blending between them. An example is shown in ➋ *Fig. 16.1*) A colour recognition system must partition the image, so that each colour imprint can be analysed separately. Once the segmentation has been completed, we can rely on standard binary image processing operators to measure them individually and thereby complete the inspection process. In some applications, colour separation can be performed very effectively using optical filtering (➋ Chaps. 4 and ➋ 5) and this should never be forgotten as an alternative to the software-based techniques that we shall discuss in this chapter. Optical colour-separation techniques are robust, fast and cheap but unfortunately are inflexible compared to software methods. (Fig. 4.11) Our theme throughout the remainder of this chapter is the use of colour recognition algorithms for image segmentation. Some of these are grounded in Colour Science, while others use a standard Pattern Recognition methodology, with spectral decomposition being performed optically by a bank of narrow band-pass filters.

Colour recognition algorithms have been devised around RGB, HSI and YUV tristimulus models. The question therefore arises as to which is best. It is reasonable to expect that the answer to such a question would be based on the large body of research that is encompassed by the term *Colour Theory.* However, practical considerations must prevail, as in any other branch of engineering. Video cameras separate the spectral components by optical filtering, thereby generating RGB signals first. If we were to base colour recognition algorithms on other tristimulus models, we would need to transform the signals first. (Some cameras contain internal circuitry for doing this but we should not suppose that cameras can measure

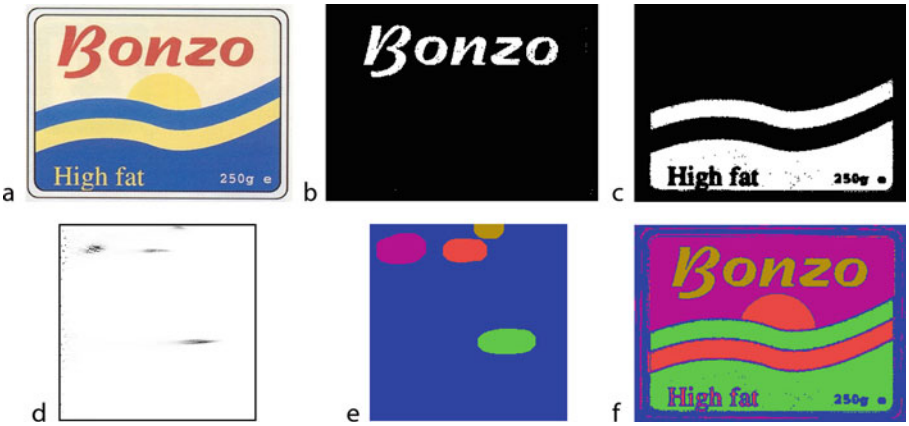


Fig. 16.1

Segmenting an image using colour. Note: parts (d)–(f) illustrate points explained later this but are not meaningful at this point in the chapter. (a) Original image: simulation of a container for a butter substitute, printed in block colours. (b) Output of a programmable colour filter (PCF, Sect. 16.3.1) designed to recognise "red". (c) Output of a PCF designed to recognise "blue". (d) HS scattergram, negated for clarity. (e) Clusters were identified by filtering and thresholding to form grey-scale blobs. The map image is displayed in pseudo-colour for clarity of illustration. (f) Programmable colour filter programmed using the map image in (e). Notice that the pseudo-colours have no direct relationship to the true colours in the original image; it is important point to understand that the original image has been segmented is a way that is subjectively judged to be reasonable

HSI or YUV directly.) Conversion formulae between RGB and other colour models are given in [3]. If we can separate two colour classes in, say, HSI space, we can also do so in RGB space. The partitioning surface may be much more complicated in one space than the other but the two classes can nevertheless be separated perfectly in both. The reason is that the RGB-to-HSI transformation is reversible.

16.2 Applying Pattern Recognition Methods in RGB Space

The reader is referred to the Sect. 16.5 of this chapter, where the Pattern Recognition Techniques mentioned here are described in more detail.

One way to recognise colours in a machine is to employ a standard pattern classifier to partition RGB space. This is effective for discriminating between block colours, since large areas of nearly constant colour generate dense compact clusters, with very points lying between them. It is often instructive to analyse the structure and relationship between clusters using interactive software. For example, MATLAB [4] enables users to "fly around" clusters in a 3D space. Users can view the clusters from different directions and thereby gain a better understanding of the nature and difficulty of the discrimination task (*Fig. 16.2*).

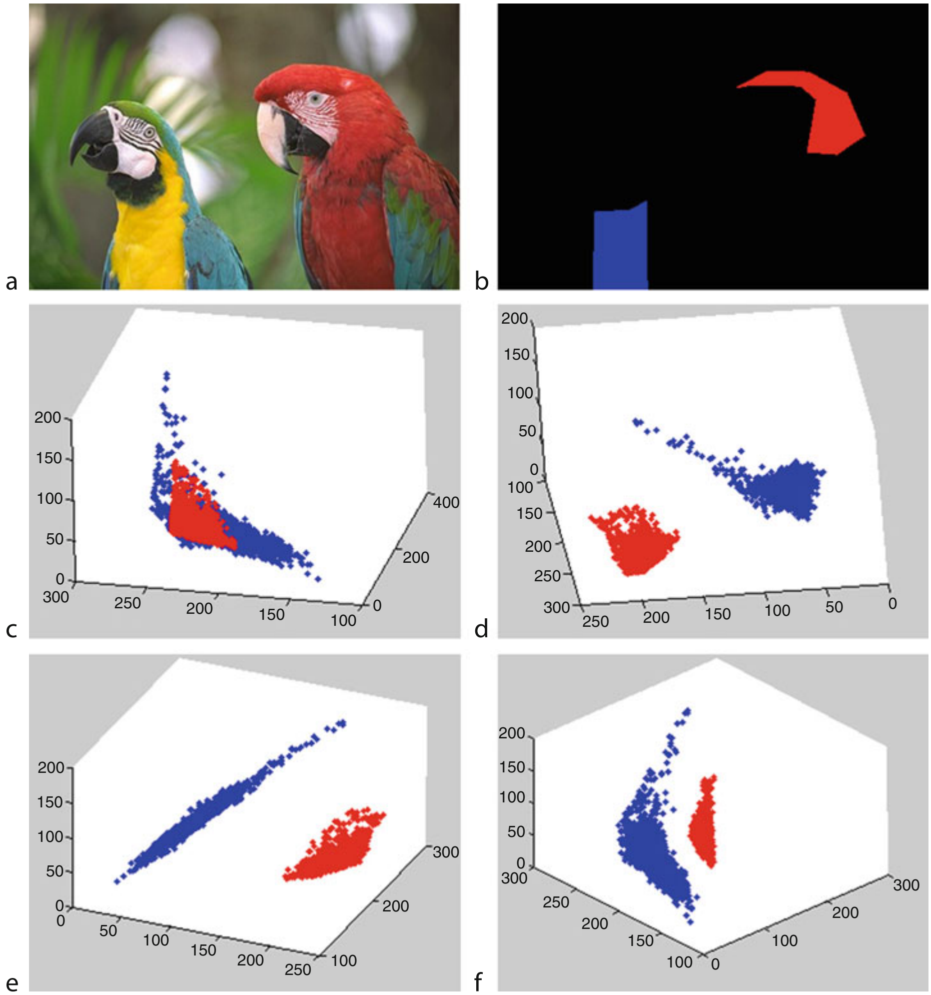


Fig. 16.2

Interactive analysis of clusters formed in RGB space by two regions of nearly constant colour. (a) Original colour image. (b) "Yellow" and "red" mask regions, selected by the author. (c)–(f) Screen-shots captured during an interactive MATLAB session, as the author "flew around" RGB space. (Points in RGB space corresponding to yellow are displayed in blue here, for clarity.) Notice that the clusters are well separated, although they appear to overlap in (c). It would be a trivial exercise to design a colour recognition system that distinguishes between these particular samples of "yellow" and "red"; a linear classifier, Compound Classifier or Nearest Neighbour Classifier, or Neural Network would all do this, with ease. (These are all discussed later and in Chap. 24.)

Standard Pattern Recognition methods can be used to partition RGB space so that "universal" and application-specific "named" colours can be identified automatically. Both supervised and unsupervised learning algorithms can be applied. The low dimensionality (3 for RGB space) makes it possible to use human intuition, guided by interactive data-visualisation software, to guide the choice/design of a suitable classifier. The following pattern classifiers are described in the Sect. 16.5 to this chapter. Also see [1].

(a) Compound Classifier (CC)
(b) Nearest neighbour classifier (NNC)

These classifiers are both capable of forming decision surfaces of unlimited complexity and effective learning rules have been devised. For both of these classifiers, there is a clear and easily understandable relationship between the values of the stored parameter and the position of the decision surface. This advantage is not enjoyed by some other classifiers. This is particularly valuable as human beings are able to visualise the clustering of data points in low-dimensionality spaces, such as RGB and HSI spaces. As a result, a vision engineer is able to develop an intuitive "feel" during the classifier design process and can assist the learning algorithms in adjusting the placement of the decision surface. It is manifestly obvious from visual examination of the clustering in RGB space that linear discrimination methods are not adequate for all but the simplest colour recognition tasks. (*Fig. 16.2* shows an extremely simple distribution of data points.) Separating one compact class (e.g., *torquoise*) from its complement (i.e., *not torquoise*) obviously requires a more sophisticated approach that either the CC or NNC can fulfil but a linear classifier cannot (*Fig. 16.3*). In *Fig.16.4*, results are shown for a Compound Classifier that was designed to recognise a single ill-defined class: *apple red-yellow*. (This involves *colour recognition*, not *colour discrimination*.) The author intervened in this design process only to define a polygonal region to represent this class. In practice, a person might be able to improve recognition accuracy and computational efficiency.

Multi-layer neural networks provide an alternative solution, as they are also able to implement complex decision surfaces. The author has not needed to employ neural networks for colour recognition, since they do not confer any obvious advantages over the CC and NNC, which more easily allow the vision engineer to retain an intuitive involvement in the classifier design process.

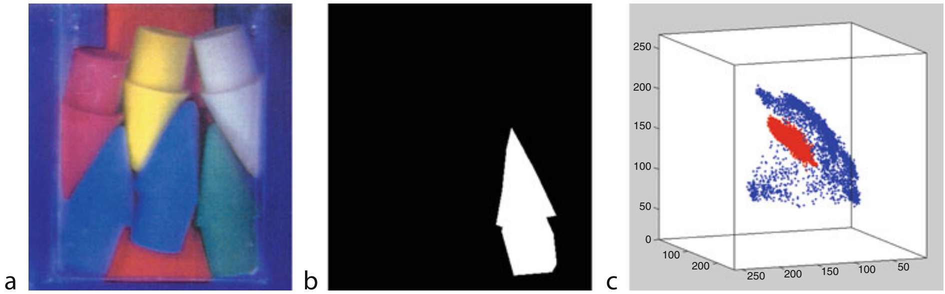


Fig. 16.3

Showing a compact colour class ("torquoise") and its complement ("not torquoise") (a) Original image. (Packet of rubber pencil erasers.) (b) Binary mask, drawn by the author. This defines the "torquoise" class. After dilation, its logical inverse defines the "not torquoise" class. (c) HSI plot for points in the two classes: "torquoise" (red points) and "not torquoise" (blue points). Interactive exploration of this space shows that the "torquoise" class lies inside a hollow shell (i.e., the class "torquoise")

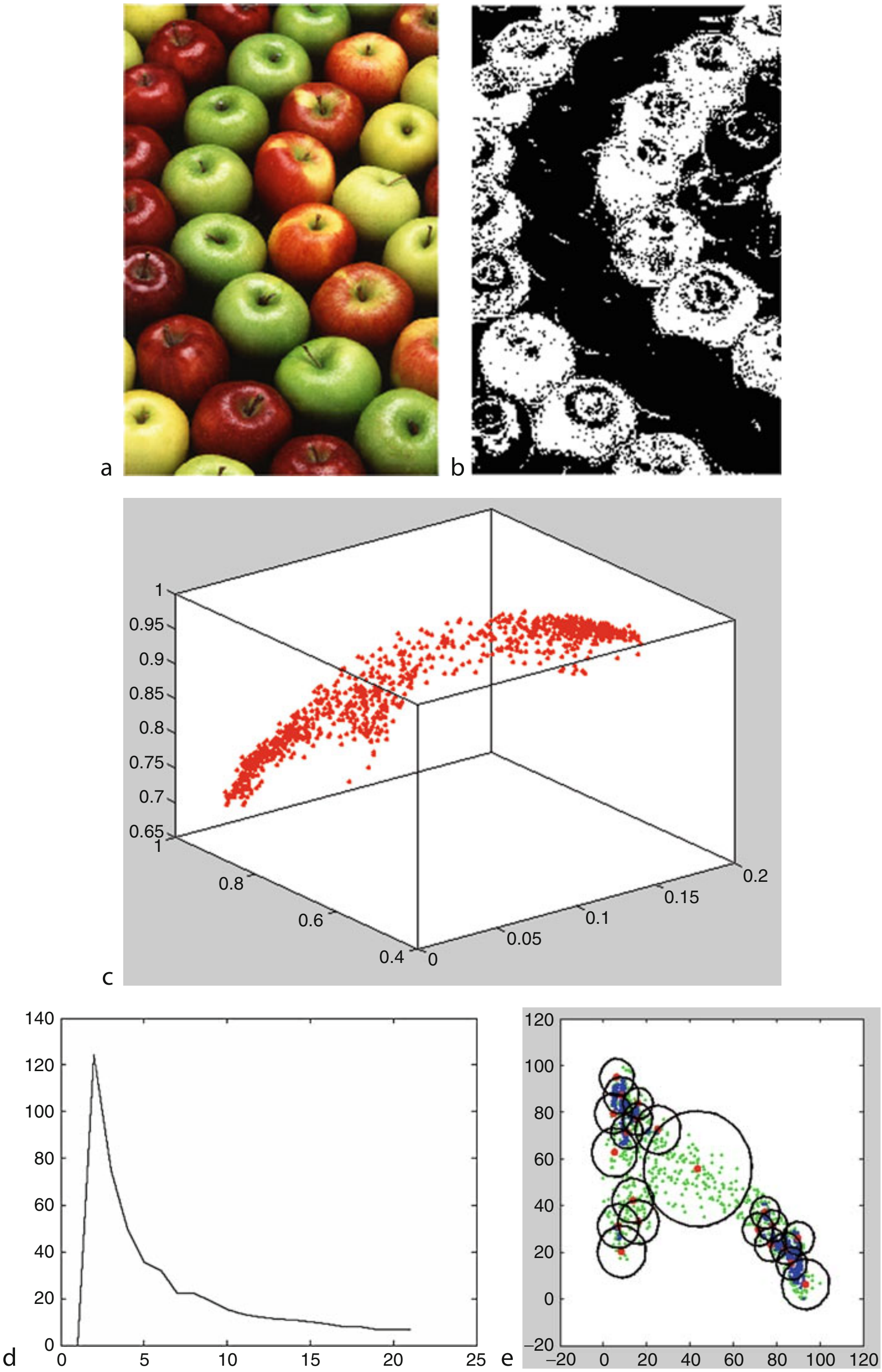


Fig. 16.4 (Continued)

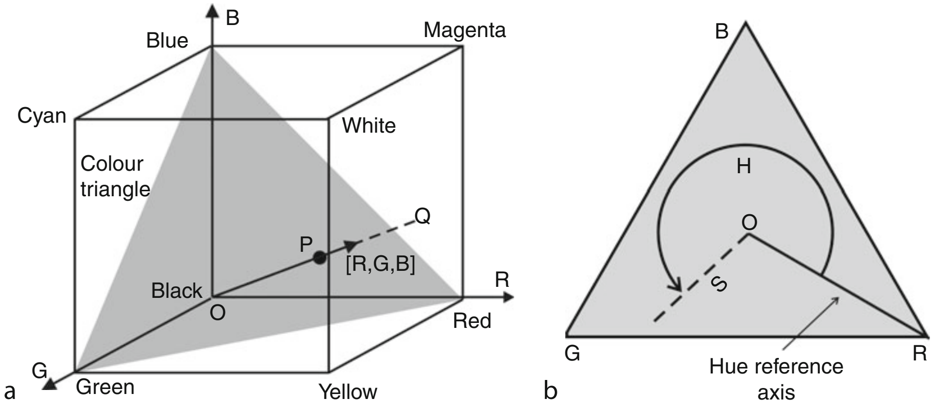
16.3 Colour Triangle

In this section, we move towards an alternative approach to colour recognition. This is described first in terms of RGB space but is, in reality based on an HSI model.

Consider ➋ *Fig.16.5*, which shows a diagram representing RGB space. The cube defined by the inequality

$$0 \leq R, G, B, \leq W$$

is called the *Colour Cube.* (W is a positive constant.) The Colour Cube shows the allowed range of variation of the point [*R,G,B*]. The *Colour Triangle,* also called the *Maxwell triangle,* is defined as the intersection of that plane which passes through the points [*W,0,0*], [*0,W,0*] and [*0,0,W*], with the Colour Cube. Now, the orientation of the line joining the point [*R,G,B*] to the



■ **Fig. 16.5**
RGB Colour space. (a) Relating the Colour Triangle to the Colour Cube. (b) Hue (H) and saturation (S) use polar coordinates to define a point in the Colour Triangle. Q is the point at which the line OP extended crosses one of the faces of the Colour Cube

■ **Fig. 16.4**
Using a Compound Classifier to recognise "apple red-yellow". (a) Original image. A small region of this image (about the same size as one apple) was selected by the author to generate the training data. (b) Result of applying the Compound Classifier to each pixel in the input image (i.e., including but not limited to the training set). (c) The distribution of points in HSI space. (d) Outliers were removed and Maximindist was then applied. (See ➋ Sect. 16.5 to this chapter.) This is a plot of the maximin distance versus the number of stored points. (e) HS space. Green points belong to the original training set. Outliers were then removed (blue points). Maximindist was then applied to the blue points. This generates the starting positions used by the one-class learning rules for the Compound Classifier. The circles form the decision surface of the Compound Classifier after learning. Notice that the large circle in the centre could be replaced by a two, or more, smaller circles. This would improve the performance of the classifier

origin could be specified uniquely by two angles (not shown). An alternative, and much more convenient way to do this is to define the orientation of this line, by specifying where the vector [R,G,B] intersects the Colour Triangle (P in ➋ *Fig. 16.5*) The vector [R,G,B] is extended if necessary. Then, by specifying two parameters (i.e., defining the position of point P in the Colour Triangle), the orientation of the [R,G,B] vector can be defined. All points lying along the line OPQ are associated with the same sensation of colour in a human being. (Very close to the origin there is a loss of colour perception but, for simplicity, we shall ignore this.) Hence, all RGB vectors which project onto the same point in the Colour Triangle are associated with the same colour name. For the purpose of setting the PCF's look-up table, colour names are assigned by an authority, called the *teacher*, under controlled lighting conditions. The following points should be noted:

(a) Points in the Colour Triangle that are very close together are very likely to be associated with the same colour name.

(b) The quantity ($R + G + B$) is a useful estimate of the intensity perceived by a human being when he/she views the same scene as the camera. This is a demonstration of *Grassman's Law* [2]. (The length of the RGB vector, measured by Euclidean distance between its ends, is not nearly so useful as a predictor of human perception of intensity.)

(c) The position of the point P defines the *chromaticity*. This is the perceptual quality of a colour, independently of its intensity. Chromaticity requires the specification of two parameters. In the HSI model, these are hue and saturation.

(d) The Colour Triangle allows us to use a graphical representation of the distribution of colours, expressing them in terms of the spatial distribution of blobs in a monochrome image. This is a particularly valuable aid to our understanding the nature and interrelationship between colours. It also permits us to use standard image processing software, to perform such operations as generalising colours, interpolating between colours, merging colours, performing fine discrimination between similar colours and simplifying the boundaries defining the recognition limits of specific colours.

(e) Since the colours perceived by a human being can be related to the orientation of the [R,G,B] vector, a narrow cone, with its apex at the origin, can be drawn to define the limits of a given "named" colour. The intersection of the cone with the Colour Triangle generates a blob-like figure.

(f) Not all colours can be represented adequately in the Colour Triangle. The Tristimulus model provides a useful working hypothesis, but it does not guarantee that perfect colour recognition is possible using just the RGB primaries. For example, "gold" and "yellow" are mapped to the same region of the Colour Triangle. Vivid purple is mapped to magenta, while vivid cyan is mapped to a paler (i.e., less saturated) tone. In relative terms, these are minor difficulties, and the Colour Triangle remains a very useful concept for colour vision. If the reader still doubts this, remember the satisfaction that numerous people have with the rendering of colours on television, which is, of course, based on RGB sensing.

(g) Fully saturated mixtures of two primary colours are found on the outer edges of the Colour Triangle, whereas the centre of the triangle, where all three primary components are balanced, represents white. We shall refer to the centre of the Colour Triangle as the *white point*, since it represents achromatic (*neutral*) tones. Other, unsaturated colours are represented by nearby points within the triangle. Hue (H) is represented as the angular position of a point relative to the centre of the Colour Triangle, while the degree of saturation (S) is measured by the distance from the centre (➋ *Fig. 16.5b*).

(h) In order to calculate hue and saturation from the RGB variables, we use the following transformations:

$$H = \cos^{-1}\left((2R - G - B) \Big/ \left(2\sqrt{((R - G)^2 + (R - B)(G - B))}\right)\right) \qquad (1)$$

$$S = 1 - 3\mathrm{MIN}(R, G, B)/(R + G + B) \qquad (2)$$

$$I = (R + G + B)/3 \qquad (3)$$

16.3.1 Programmable Colour Filter

The Programmable Colour Filter (PCF) provides an electronic/software method of recognising colours. It does not require the use of any optical devices, except those built into a standard RGB camera. A PCF operates under software control and hence is more versatile than optical filters, which have a fixed spectral response. We shall develop the theoretical basis for the PCF later but first we describe how it may be implemented in low-cost electronic hardware. The system outlined in *Fig. 16.6* is capable of recognising colours based on an RGB video signal in real-time. It is able to recognise 65,536 (= 2^{16}) colours. Notice that the memory required is modest: 32 MB per look-up table (LUT). (The memory requirement is halved if we limit ourselves to working with not more than 256 colour classes.) Since several LUTs can be stored in a single high-capacity memory chip, such a system is capable of rapid switching from one colour recognition function to another, simply by altering the *LUT selection* inputs.

As evidence that this seemingly naive approach actually works (see *Figs. 16.7* and *16.8*).

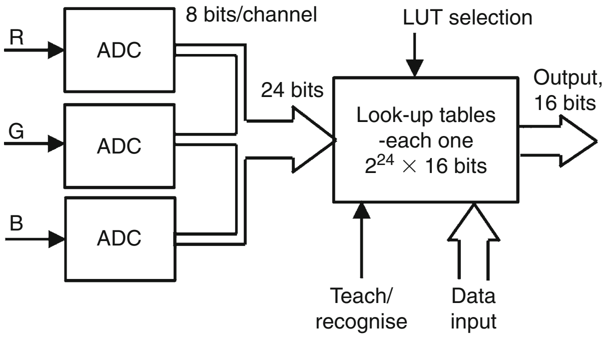
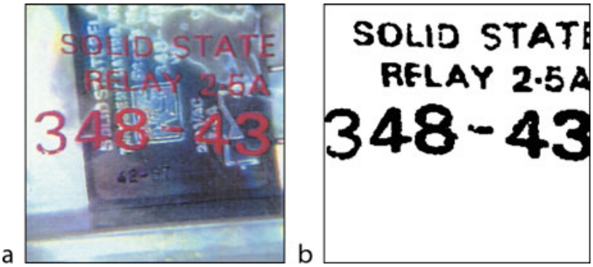
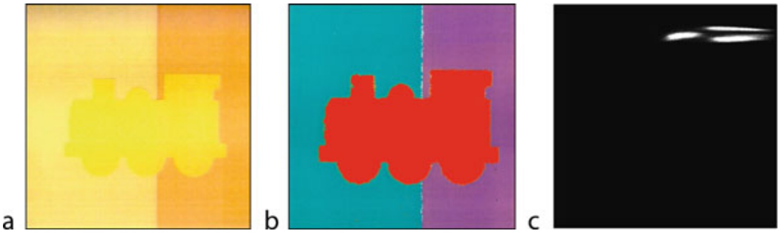


Fig. 16.6
Programmable Colour Filter (PCF). The inputs are the RGB video channels (analogue). The output effectively defines the intensity in a monochrome image and hence can be manipulated using standard grey-scale image processing functions [4]



■ Fig. 16.7
Colour recognition. (a) Original image. Black electronics component with shiny silver printing. It is in a transparent plastic bag with red printing. The background is white. (b) Output of the PCF



■ Fig. 16.8
Another example of colour recognition. (a) Original image. The original artwork showed lower contrast than is evident here; the three shades of yellow are just distinguishable by eye. (b) Output of the PCF, displayed in pseudo-colour. (The PCF output has only four different levels.) (c) HS scattergram, no processing. The significance of this will be become clear later

16.3.2 Programming the PCF

We have not yet explained how the contents of the PCF look-up table are established. This may be a very slow process but need only be performed once. Thereafter, the PCF is very fast. The colour filter is normally "programmed" interactively (➲ Fig. 16.9). First, a sample colour image is displayed and the user draws a mask around a region of interest. Colours in this region are then analysed and a colour scattergram is generated in the Colour Triangle. To understand this, let us assume that each point $[i,j]$ in the region of interest yields a colour vector $[R_{i,j},G_{i,j},B_{i,j}]$. Furthermore, this vector, or its projection, will be assumed to intersect the Colour Triangle at that point defined by the polar co-ordinates $[H_{i,j},S_{i,j}]$ (P in ➲ Fig. 16.5). These hue and saturation values can be calculated using the equations just given. The point to note is that, each address $[i,j]$ in the region of interest defines a point in the Colour Triangle. The *Colour Scattergram* is generated thus

Rule 1: Clear the working image, I_{work}.
Rule 2: Select a point $[i,j]$ in the region of interest within the colour image.
Rule 3: Compute values for the hue and saturation, $[H_{i,j}, S_{i,j}]$. These are polar co-ordinates.
Rule 4: Add 1 to the intensity stored at that point defined by $[H_{i,j}, S_{i,j}]$ in image I_{work}. (Hard limiting occurs if we try to increase the intensity beyond the white level, 255.)
Rule 5: Repeat Rules 2–4 for all $[i,j]$ in the region of interest.

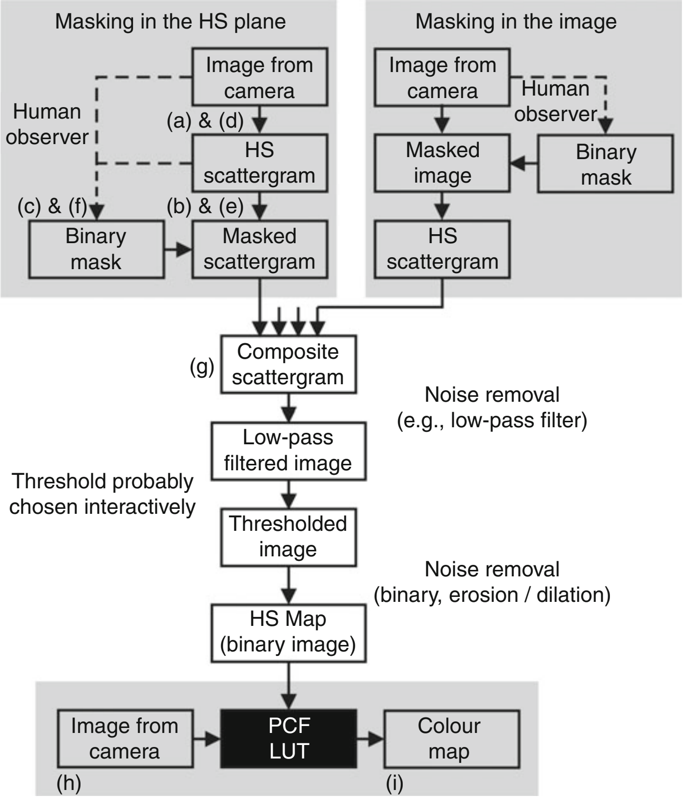


Fig. 16.9
Programming a colour filter and using it for colour recognition

Dense clusters in the colour scattergram are associated with large regions of nearly constant colour. A large diffuse cluster usually signifies the fact there is a wide variation of colours in the scene being viewed, often with colours blending into one another. On the other hand, a printed sheet containing only block colours (i.e., regions of constant colour with step-wise colour changes) generates a series of small dense clusters that are distinct from one another. The colour scattergram is a very useful method of characterising colour images and hence has a central role in programming the colour filters described below. As we shall see later, it is a valuable aid to understanding and communicating ideas about colour to another person, or to a computer.

Programming the PCF proceeds in one of two ways:

(a) The user defines a set of colours to be recognised, by interactively drawing a closed contour, usually around the main cluster in the colour scattergram. Other clusters and outlier points are normally ignored (see ➋ *Fig. 16.4*). The result is a binary image containing a single blob.

(b) The colour scattergram is processed using a (grey-scale) image processor, such as QT (➋ Chap. 21). Processing will probably involve

- Grey-scale filtering (This is usually needed to smooth the scattergram).
- Thresholding.

- Binary noise reduction.
- Selection of the blob(s) to be retained.
- Binary morphology, to enlarge the blob(s). (Needed for generalising colour recognition).
- Blob labelling. (QT function *ndo*) Each blob is assigned a different intensity value. The result is a multi-level image containing one or more blobs.

The next step is common to both manual or automated definitions of blobs in the Colour Triangle. It is called *Back-projection*, since it involves defining values for a set of points in the Colour Cube from those that were previously projected into the Colour Triangle. Consider ➲ *Fig. 16.10*. The position of a point in the Colour Triangle can be specified by two Cartesian coordinates *U* and *V*, where:

$$U = (R - G)\Big/\Big[\sqrt{2}(R{+}G + B)\Big]$$

and

$$V = (2B - R - G)/[\sqrt{6}(R + G + B)]$$

(It is easier to use Cartesian Coordinates ([*U,V*], here instead of hue and saturation [*H,S*], although one pair of values uniquely defines the other.) The set of points within the Colour Cube that give a constant value for *U* and *V* lies along a straight line and will be denoted by *Φ([U,V]* (➲ *Fig. 16.11*). All members of *Φ*([*U,V*]) are perceived as being the same colour as one another and map onto the same point in the Colour Triangle. Let *θ* be a set of connected points in the Colour Triangle (i.e., forming a blob)

$$\theta = \{[U_1, V_1], [U_2, V_2], [U_3, V], \ldots\}$$

If all members of *θ* are associated with the same colour name (ω), then all points in the set

$$\psi = \{\Phi(\mathbf{X})|\mathbf{X} \in \theta\}$$

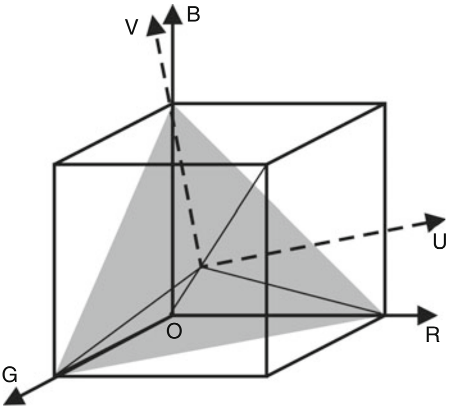


Fig. 16.10
Defining Cartesian axes U and V in the Colour Triangle

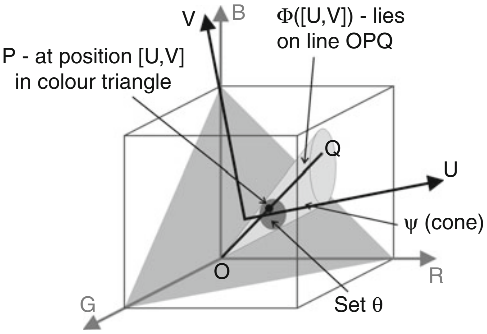


Fig. 16.11
Calculating the PCF look-up table (LUT) values by back-projection. The point at position [U,V] in the Colour Triangle is a member of the blob (i.e., connected set) θ. All points in the set Φ([U,V]) are perceived by an observer as being of the same colour as each other and lie on the line OPQ. The cone ψ has its apex at O and has cross-section θ in the Colour Triangle. All points in ψ are therefore associated with colour name ω. This partially determines the values for the PCF LUT. Other blobs in the Colour Triangle are treated in the same way. Points in the Colour Triangle that have not been specifically associated with a name are assigned to the class "colour not recognised"

should also be assigned to that same colour class. Notice that ψ is a cone with its apex at the origin of the colour space. The intersection of that cone with the Colour Triangle is the set θ. Back-projection is the process of setting all cells in the PCF look-up table that have addresses in ψ to the value ω.

We have just described the rationale for the back-projection procedure, not the computational details. In fact, the same operation is performed for all points in the Colour Triangle. This is necessary, so that we can define values of all points in the Colour Cube and hence all entries in the PCF's look-up table. There may be several blobs, with different values of ω, in the Colour Triangle, so the PCF will, in future, be able to recognise several classes. Points that are not within one of the blobs in the Colour Triangle are treated in the same way. In this case, the value for ω is a number representing *"colour not recognised".*

To distinguish this from the method described in the following section, we shall refer to this as *PCF1*. Note that it is based on the UV plane. Practical results based on this approach will be presented later

16.3.3 Programmable Colour Filter in the HS Plane

The previous section described colour recognition techniques based on RGB colour space and coordinates, $[U,V]$, to define points in the Colour Triangle. We shall now discuss methods based on a different but closely related form of scattergram: hue and saturation are plotted using Cartesian coordinates (see *Fig. 16.5b*). In order to calculate *HSI* from *RGB* variables, we use Eqs. 16.1 –16.3. Once again, we assume that we can safely discard intensity and that look-up tables could be used to improve the time to "calculate" *H* and *S*. Once we have derived

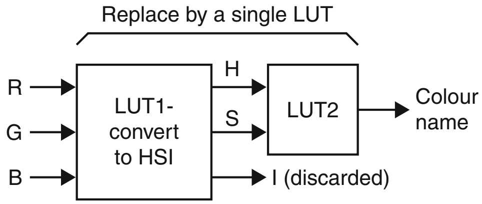


Fig. 16.12
Colour recognition based on [H,S] uses RGB inputs

a method for recognising colours based on HS values, it is possible to use another LUT for colour recognition. We then have two LUTs in tandem:

- LUT1 converts from RGB to HSI representation (LUT1 output defines the inputs to LUT2).
- LUT2 performs the colour recognition based on [*H*,*S*].

(*Figure 16.12*) These can be replaced by a single LUT. It is justified conceptually on an HS model but in reality it uses RGB inputs. The system shown in *Fig. 16.12* is also capable of performing real-time colour recognition, whether we use one or two separate LUTs.

We begin by calculating a colour scattergram in HS space. This is very similar to the process described in the previous section. The result is again a grey-scale image in which intensity indicates how many pixels have those particular values of [*H*,*S*] (see, for example, *Fig. 16.13b*). The scattergram can then be processed in a very similar way to that described in the previous section; thresholding, filtering and blob analysis (to eliminate the effects of minor peaks in the scattergram) are appropriate (*Fig. 16.9*). The HS scattergram can be calculated by the QT function *shs* and the recognition is performed using *pcf* (see Chap. 21). To distinguish this from the method described in Sect. 16.3.2, we shall refer to this as *PCF2* and refer to the fact that it is based on the HS plane.

16.3.4 Incorporating Human Intuition

The PCF1 and PCF2 procedures described so far work reasonably well but do not always produce sufficiently accurate colour discrimination. Matters can often be greatly improved, by allowing a vision engineer to adjust the blobs in the [*U*,*V*]/[*H*,*S*] plane interactively. For both the PCF1 and PCF2 colour recognition procedures, we have simplified the task of designing a filter, by creating a simple visual representation that a person can readily understand and adjust. We shall call this the *map image*. By inspecting the blobs in the map image, an experienced engineer can predict the behaviour of the PCF that will result from it. For example, when a blob representing a colour class (*C*) is enlarged, the set of colours that the PCF will associate with C also becomeṣ larger. This leads us to the concept of *colour generalisation*. We shall describe it and other procedures that an engineer can employ to adjust the behaviour of the PCF later. These all rely on manipulating a collection of blobs in the map image.

16.3.5 Colour Generalisation

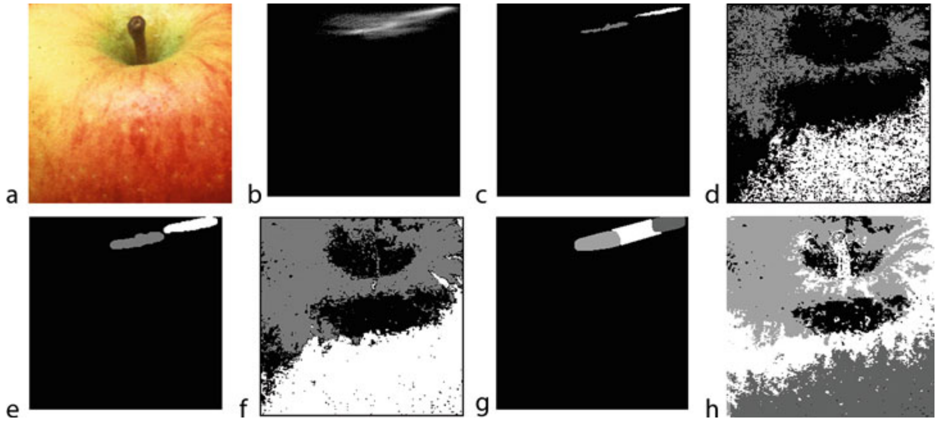
For the sake of simplicity, let us consider just one blob (Z) in the map image. When we use Z to define the LUT contents, a certain set (S) of colours will be recognised. When Z is enlarged, say by binary dilation, S and a set of "similar" colours will be recognised. By changing the amount of dilation, the degree of generalisation can be varied. In the case of the PCF2 procedure, dilation can be applied in different amounts along the H and S axes, by using a rectangular, rather than square structuring element.

We can extend this idea, so that the blobs become very large indeed, but we cannot, of course, allow them to overlap. Suppose that there are several separate blobs in the map image, representing a number of distinct colour classes. In some applications, it may be important to distinguish between colour classes, rather than identify the precise limits of any one of them. For example, a widget may have only achromatic, yellow and red features. The map image would then have three blobs. By dividing the plane on a nearest neighbour basis, all colours can be attributed to the "achromatic," "red" or "yellow" class. "Orange" would be divided between "red" and "yellow," while "green" would be assigned to "yellow." (It is more similar to "yellow" than it is to "red" [➔ *Fig. 16.13*].) To partition the $[U,V]/[H,S]$ plane, we can use the watershed function (➔ *Fig. 16.14*). The following operations are required. (QT commands are set in brackets. See ➔ Chap. 21)

```
Watershed function (wsh)
Negate (neg)
Label blobs. Shade each blob so that it has a unique intensity. (ndo)
Dilate. Apply a small amount of dilation, to fill the gaps between blobs. (dil)
```

16.3.6 Colour Interpolation

Suppose there are two small separate blobs (sets of points, S_r and S_y) close together in the map image, representing "red" and "yellow". Then, we can reasonably expect the points within the



■ **Fig. 16.13**

Colour generalisation and interpolation. (a) Original image. Natural products, such as apples, often have colours blending into one another. (This tends to produce indistinct clusters in the colour scattergram.) (b) Hue-saturation (HS) scattergram. (c) HS scattergram thresholded. (d) PCF based on map image (c). (e) Binary dilation applied to (c). (f) PCF based on (e). (g) Mask image, contrived to show colour interpolation. The two grey blobs are derived from (c). The white region was obtained from their convex deficiency. (h) PCF based on (g)

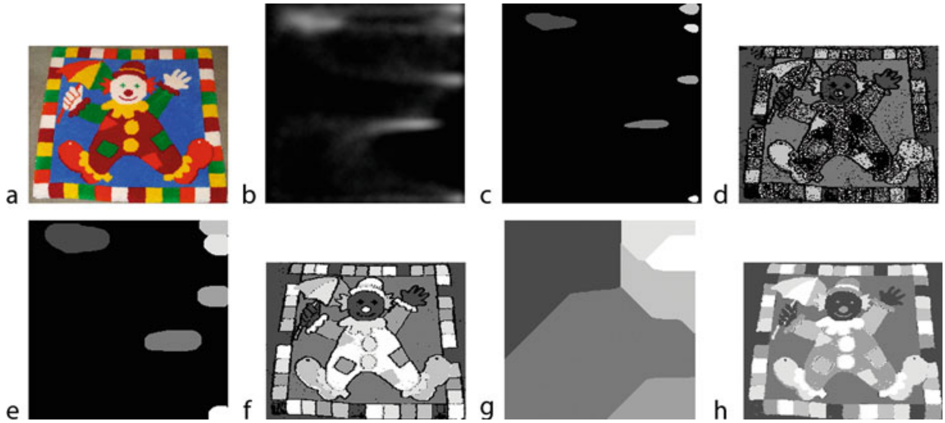


Fig. 16.14

Programmable colour filter (PCF) applied to an image with no blending of colours. (a) Original image: hand-knotted rug. (b) HS scattergram. (c) Threshold applied to (b), with filtering to eliminate noise and blob shading. (d) PCF based on (c). (e) Dilation applied to (c). (f) PCF based on (e). (g) HS space partitioned by applying the watershed function to (c). (h) PCF based on (g)

convex hull of the combined set $S_r \cup S_y$ but not in either of them to be called "orange." (*Fig. 16.13*). Here is the algorithm expressed in pseudo-code for an image processor. (Again, QT commands are given in brackets.)

```
Convex hull (chu_all)
Exclusive OR (Find points in the convex hull but not originally in the [U,V]/
[H,S] plane. (exr)
```

16.3.7 Logical Operations on Colour

Suppose that we have analysed five images, containing samples of "butter," "canary," "sulphur," "banana" and "lemon" and that this has produced five separate binary map images; there is one blob in each map image, representing a different subset of "yellow." In order to obtain a PCF that can recognise the more general class "yellow," we can combine these images by forming their logical union. The result is a map image containing a single blob that we can then use to build PCF1, or PCF2, so that it will recognise the general colour class called "yellow." In this way, we are able to form the logical union of two, or more colour sets. However, we can perform fuzzy logic on colour sets, by combining grey-scale map images. This is implicit in the flow chart for building a PCF shown in *Fig. 16.9*. Colour union is particularly useful for combining colours derived from natural products; since they are highly variable. For example, we might want to program a PCF to recognise the colour *leaf green*. This will require us to combine results from many different samples of foliage (*Fig. 16.15*). This is simply an extension of the principle outlined in *Fig. 16.9*. (Notice the multiple inputs to the box labelled *Composite scattergram*.) This diagram also shows a binary mask, drawn by a vision engineer, ANDed with a (grey-scale) map image, to suppress "unwanted" colours. "Red" and "orange-yellow" are eliminated in *Fig. 16.15*.

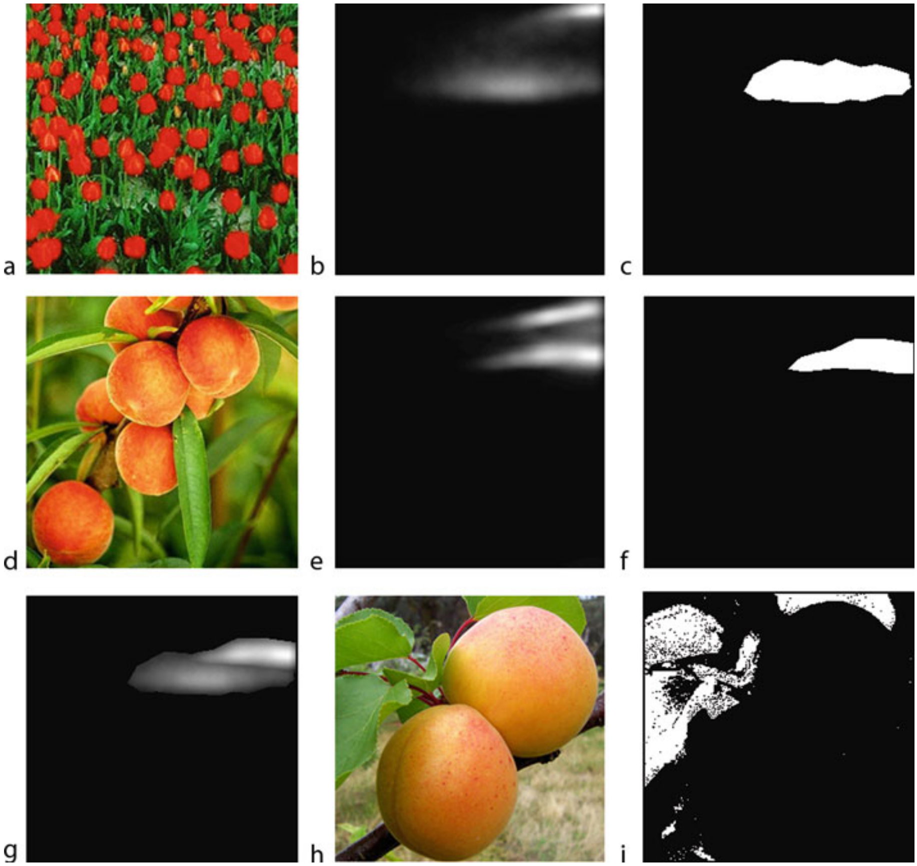


Fig. 16.15
Building a general concept of “green” using fuzzy logic operations in HS space. In practice, many more than two clusters should be merged to build a PCF that will have a sufficiently general concept of “green”. (a) Original image I_1. (b) HS scattergram of I_1. (S_1) The upper cluster is generated by the red flowers; the lower cluster is created by the green leaves. (c) Binary mask, M_1. (Drawn by the author; encloses the “green” cluster and nothing else.) (d) Original image I_2. (e) HS scattergram of I_2. (S_s) The lower cluster is created by the green leaves; the yellow-orange fruit generates the upper cluster. (f) Mask M_2 was drawn by the author to enclose the “green” cluster in S_2. (g) Merging the two “green” clusters. First, M_1 and M_2 were used to mask S_1 and S_s respectively. The resulting images were then merged, using pixel-by-pixel addition. The result (S_g) shown here, is a more general concept of “green” than is represented in either I_1 or I_2. (h) Original image I_3. (i) The PCF was programmed using a thresholded version of image S_g and was then applied to I_3. Applying suitable noise-removal operations and colour generalisation, much better results could be obtained for a PCF that recognises “green”

16.3.8 Interaction and Human Intuition

A useful trick when designing a PCF is to use a cursor to select blobs one at a time from a binary map image. In this way, the contribution of each blob to the total behaviour of the PCF can be understood. Individual blobs can then be dilated/eroded, separately, to improve the desired colour recognition accuracy. Drawing around an individual intensity peak (cluster in the scattergram) in a grey-scale map image produces a polygonal mask that can achieve a similar result. This is illustrated in ➲ *Fig. 16.15*. Interactive analysis like this can be particularly useful in identifying diffuse low-intensity clusters in the colour scattergram that might otherwise go unnoticed. Some of these can make a significant contribution to the performance of a PCF, yet be difficult to "capture" successfully by an automated process.

While ➲ *Fig. 16.9* summarises the likely sequence of steps involved in designing a PCF, it should not be taken as the only possible way to proceed. An engineer may need to experiment for some considerable time before he/she is fully satisfied with the results. Once the PCF design phase has been completed, the colour recognition process is automatic and needs no further human intervention. An engineer should never apologise for using human intuition in the design process, provided the results are verified properly later. In this chapter, the author has tried to present a logical case for each step in the design of a PCF but human experience and low cunning are far more powerful than prescriptive rules in situations like this.

16.3.9 Colour Similarity and Special Effects Filters

The PCF software devised by the author (*pcf* in QT, ➲ Chap. 21) can use, as its input (i.e., the map image), any binary or grey-scale image, provided it is of the correct size. We can, for example, generate the map image using a graphics program, in order to create special effects. Four examples are illustrated in ➲ *Figs. 16.16–16.19*.

16.3.10 Relating Physical, Graphical and Symbolic Representations of Colour

We are close to defining an hierarchy of different representations for colour. Most but not all of its elements have been described but two remain unexplained. The first is the approximation of each blob in the $[U,V]/[H,S]$ plane by a set of overlapping discs. (Squares or rectangles could be used instead.) These can be fitted manually, or algorithmically, perhaps by the rules defined in the ➲ Sect. 16.5 to this chapter). For the sake of argument, let us assume that we have already fitted a set of discs thereby approximating the blob (Z) that represents a certain colour, *C*. These circles can be defined by a set of data triples of the form:

$$S = \{[X_1, Y_1, R_1], [X_2, Y_2, R_2], [X_3, Y_3, R_3], \ldots\}$$

$[X_i, Y_i]$ is the centre of the *i*th disc and R_i is its radius. Of course, we need to modify the PCF slightly, so that it accepts as input a set of data values (S), rather than the map image containing the blob Z. This is convenient because S requires far less data storage than Z. Notice that we can apply generalisation easily, by increasing all of the circle radii ($\mathrm{R_i}$).

The last element that we need to build the hierarchy is symbolic programming (➲ Chap. 23). A Prolog program can readily associate symbolic colour names (e.g., *carpet blue* or *cola can red*) with a list of circle parameters (S); only a simple Prolog fact of the following form is needed:

$$colour(Name, [[X1, Y1, R1], X2, Y2, R2], \ldots])$$

We can now introduce the hierarchy of colour representations (➲ *Fig. 16.20*). Beginning with a pattern of light, we progress through the PCF design phase, to create symbolic references

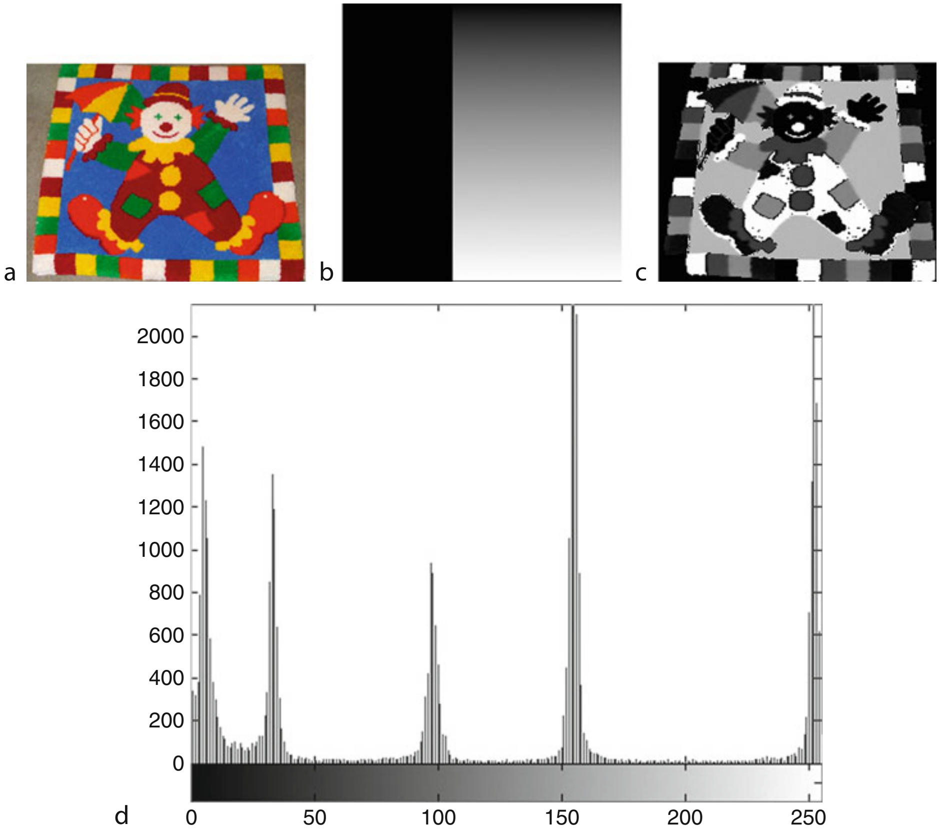


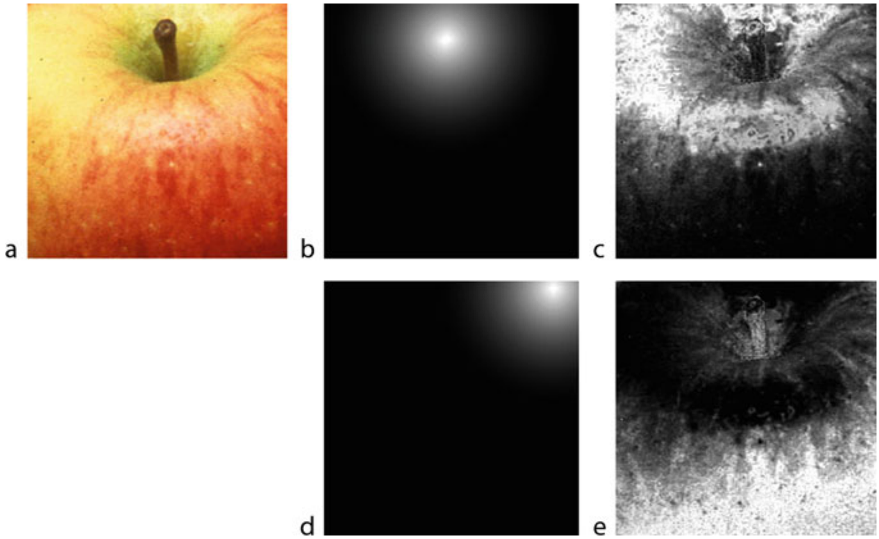
Fig. 16.16
Using a PCF to measure hue, subject to the condition that saturation is above a given limit. (a) Original image. (b) Map image. (c) Output of the PCF based on (b). (d) Intensity histogram of (c). The sharp peaks indicate that there are large regions of nearly constant hue that can be separated by simple thresholding

to colour that can be incorporated into a Prolog program. Thereafter, it is possible to identify regions containing colours that have previously been learned, in a straightforward way.

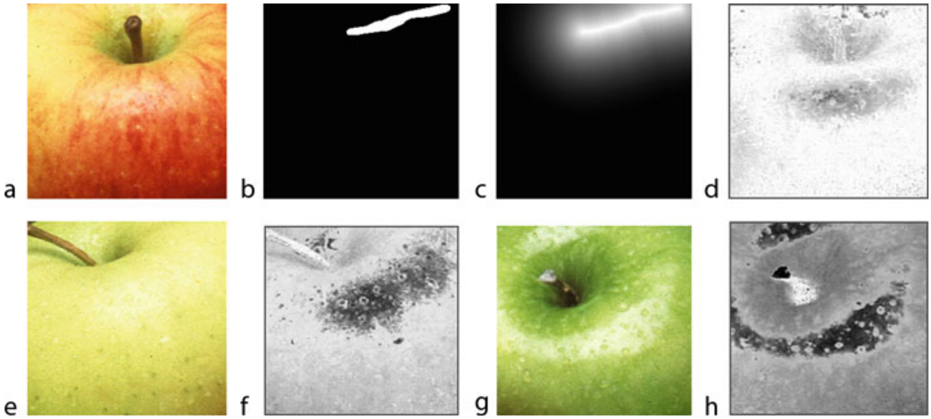
16.3.11 Prolog Programming with Colour

We now have the ability to discuss colour in Prolog, which is an AI language that is well suited to logical reasoning. We can express relationships between colours in a simple way. For example, suppose that we have four PCFs that identify "butter yellow," "canary yellow," "lemon yellow" and "saffron yellow" and that these have been associated with the following disc-parameter lists

```
List_butter
List_canary
List_lemon
List_saffron
```



■ Fig. 16.17
Programmable colour filter used to determine colour similarity. Similarity is referred to a single point, defined by given pair of HS values. (**a**) Original image. (**b**) Map image. The peak intensity corresponds to a local maximum in the ''yellow'' region of the colour scattergram (➜ *Fig. 16.13*). (**c**) Output of the PCF based on (**b**). (**d**) Another map image. The peak intensity corresponds to a local maximum in the ''red'' region of the colour scattergram. (**e**) Output of the PCF based on (**d**)



■ Fig. 16.18
Using a PCF to measure colour similarity, referred to a set of points in the HS plane. (**a**) Original image. (**b**) Processed HS scattergram derived from (**a**). (Also see ➜ *Fig. 16.13*.) (**c**) Severe blurring using the grass-fire transform applied to (**b**). (**d**) Output of the PCF based on (**c**). (**e**) New image; second variety of apple. (**f**) Same PCF applied to (**e**). (**g**) Another new image; third variety of apple. (**h**) Same PCF applied to (**g**)

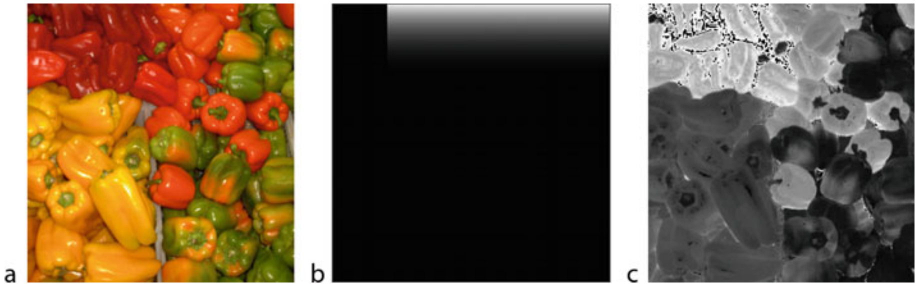


Fig. 16.19
PCF emphasising colour differences in a narrow part of the spectrum. (a) Original image. (b) Map image. (c) Output of the PCF based on (b)

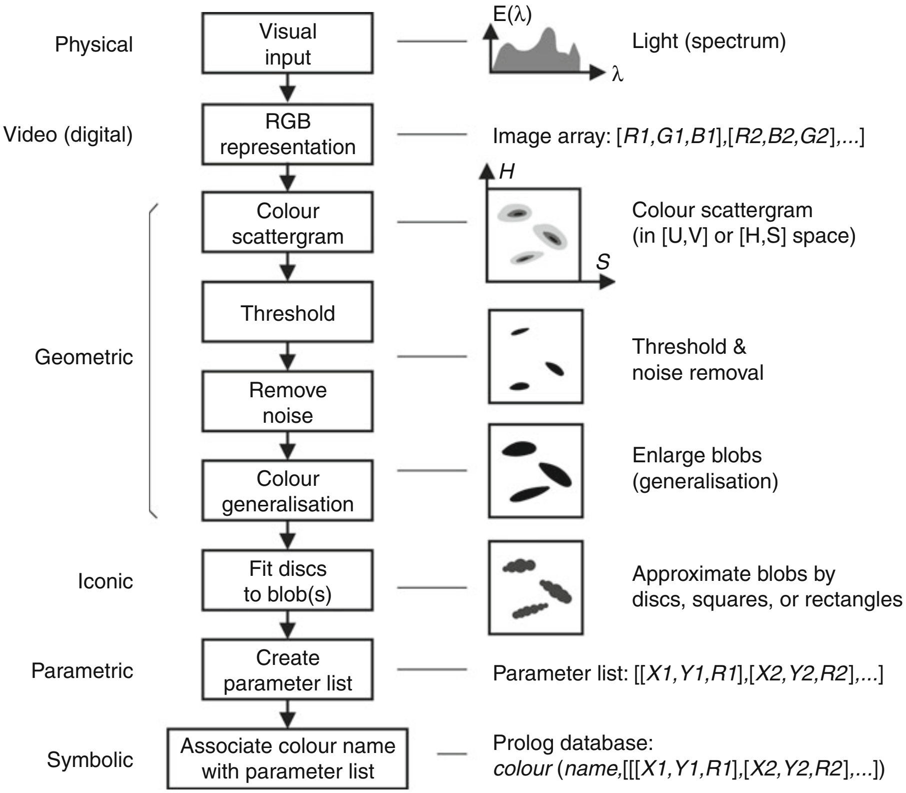

Physical
Visual input
E(λ)
λ
Light (spectrum)
Video (digital)
RGB representation
Image array: [R1,G1,B1],[R2,B2,G2],...]
Geometric
Colour scattergram
H
S
Colour scattergram (in [U,V] or [H,S] space)
Threshold
Remove noise
Threshold & noise removal
Colour generalisation
Enlarge blobs (generalisation)
Iconic
Fit discs to blob(s)
Approximate blobs by discs, squares, or rectangles
Parametric
Create parameter list
Parameter list: [[X1,Y1,R1],[X2,Y2,R2],...]
Symbolic
Associate colour name with parameter list
Prolog database:
colour (name,[[[X1,Y1,R1],[X2,Y2,R2],...])

Fig. 16.20
Various representations of colour are made possible by the PCF

as explained above. Then, "yellow" might be expressed as the logical union of these colour sets, using four very simple Prolog rules:

```
colour(X,yellow) :- colour(X, List_butter).
colour(X,yellow) :- colour(X, List_canary).
colour(X,yellow) :- colour(X, List_lemon).
colour(X,yellow) :- colour(X, List_saffron).
```

We can express colour set intersection in an equally simple way:

```
colour(X,List_lemon) :-
      colour(X,List_orange_and_lemon),% Obvious meaning
      not(colour(X,orange)).          % Recognises the set ''not orange''
```

Of greater importance is Prolog's ability to accept declarative statements (➜ Chap. 23). A simple program is listed below. (Recall that a Prolog consists of a series of tests, not instructions.)

```
object(X,banana) :-
         colour(X,yellow),      % X is yellow - this relies on the PCF to
                                  identify ''yellow'' objects
         curved(X),             % X is curved
         length(X,L),           % Length of X is L mm
         L > 100,               % Simple arithmetic test
         L < 250,               % Simple arithmetic test
         width(X,W),            % Width of X is W mm
         W > 20,                % Simple arithmetic test
         W < 50,                % Simple arithmetic test
```

16.4 Multi-spectral Image Data

Conventional colour cameras generate just three output channels (RGB), although there is no fundamental reason why more channels cannot be provided in a solid-state camera. The present limitations are due to commercial rather than technical factors. One manufacturer has recently produced a camera with a fourth VIS channel ("emerald green"). Earth resources satellites have multi-channel scanners, with sensors for visible and infrared radiation. It is possible to build a multi-channel sensor incorporating one of the following.

- Automated monochromator relying on the physical movement of a narrow slit
- As above, but using an LCD optical modulator
- Switched LED arrays providing illumination over multiple narrow wavebands
- Multiple narrow band filters mounted on a rotating wheel
- Digital micro-mirror devices

Many multi-spectral imagers are slow, requiring several video frame periods to complete a multiband scan. It is worth mentioning that data projectors change the spectrum of the light they emit in a rather limited way, by altering the proportions of three primary colours. They do not ever generate "orange" light but only a mixture of red, green and (very little) blue. For this reason, they may be unsuitable for multi-spectral imaging.

Each pixel in a multi-spectral image, covering N distinct wave-bands, may be represented by a point in N-dimensional space. Colour recognition then becomes a standard Pattern Recognition problem and is concerned with partitioning this space. We can employ well-established classification techniques, such as Neural Networks (➜ Chap. 24), Nearest Neighbour and Compound Classifiers (➜ Sect. 16.5 to this chapter). The same mathematical

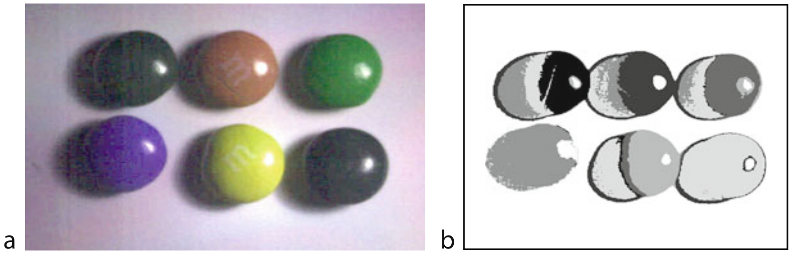


Fig. 16.21
Colour recognition of M&M candies using an NNC, operating in 13-dimensional space. (a) The scene was lit using an intense "point" source placed to the right. (This produces one small highlight on each object). Intensity values from 13 images (obtained by sampling at wavelengths of 460, 480, 500, ..., 700 nm) were combined to give a 13-element vector description of the optical spectrum of each point in the scene. One locate was defined manually on the well-lit side (right-hand side) of each object. Hence, there was only one locate representing each of the six colour classes. This simple method of designing an NNC gives good results in the well-lit areas but does less well where the intensity is low. (The camera is overloaded by the highlights, which therefore create classification errors.) To improve recognition accuracy, it would be necessary to add extra locates, by sampling in the darker areas, as well.

decision making techniques can be used with a mixture of any of the following sensory data and might reasonably be included in the term "multi-spectral":

- Visible light (VIS)
- Ultra-violet illumination and UV sensing
- Fluorescence (This might be UV lighting and VIS viewing, VIS lighting and VIS viewing)
- Phosphorescence
- Infra-red, including but not restricted to thermal imaging
- X-ray
- Range sensing
- Microwave
- Magnetic-field mapping

A simple experiment using a Nearest Neighbour Classifier to recognise colours based on a 13-element representation of colour is shown in ➲ *Fig. 16.21*.

16.5 Supplement

16.5.1 Compound Classifier

This classifier makes a binary decision M = ±1, according to the following rules:

- **X** is a Q-dimensional vector describing the colour at a single point in the input image. It is called the *test vector*. For an RGB image, Q = 3 and
$$\mathbf{X} = [\mathrm{R}, \mathrm{G}, \mathrm{B}]$$
- $\mathbf{X}_i$ ($i \in [1,N]$) is a Q-dimensional reference vector.
- $D(\mathbf{X},\mathbf{X}_i)$ is the Euclidean distance between vectors **X** and $\mathbf{X}_i$.

- C_i ($i \in [1,N]$) is the decision of a sub-classifier and is computed thus

$$C_i = \mathbf{sign}(F_i - D(\mathbf{X}, \mathbf{X}_i))$$

where F_i is a positive real number. This produces a decision surface for the sub-classifier that is a

circle, for $Q = 2$
sphere, for $Q = 3$
hypersphere for $Q \geq 4$

- $M = \pm 1$ is the decision of the Compound Classifier and is equal to

+1 if any one of the $C_i = +1$
−1 if all of the $C_i = -1$.

- The $\mathbf{X}_i$ are called *locates*, since they determine the positions of the circles/spheres/hyperspheres.
- F_i is the radius of the *i*th circle/sphere/hypersphere.

To understand the power and flexibility of the CC, we need only consider the two-dimensional case: drop a handful of mixed coins (all circular) onto a table (➲ *Fig. 16.4e*). The shadow they produce is the decision surface of a CC. Notice that the coins may but need not cast one connected shadow.

By adjusting the $\mathbf{X}_i$ and F_i, the position and size of just one circle/sphere/hypersphere sub-classifier can be altered, leaving all others unchanged. Learning rules for the CC have been devised for two quite different conditions:

(a) When two classes A and B (or A and A′) have to be distinguished (*two-class* learning rule, ➲ *Fig. 16.22*)
(b) When the bounds of a single class have to be defined (*1-class* learning rule, learning to detect anomalies, ➲ *Fig. 16.23*)

It is worthwhile recording some additional points about the CC.

(a) So far, no mathematical proof of convergence for the CC learning rules has been found. However, they have been studied extensively by experimentation and have been found to be effective in a wide variety of situations.
(b) Additional rules have been devised for automatically adding/removing subclassifiers, as needed.
(c) A procedure has been devised for finding a good starting state for the CC, so that the learning time is minimised.
(d) Points should be chosen at random from the image(s) used to generate the input to the learning classifier. It is unwise to scan the image in raster fashion, since there may be slow variations in the colour parameters across the image. Failure to comply with this requirement may result in the classifier learning one set of parameter values ($\{\mathbf{X}_i, F_i\}$) near the beginning of the raster scan of the image, only to be forced to revise then later. (This is the same requirement as is applied when computing parameters of a time series iteratively, rather than globally.)
(e) The CC can employ other distance metrics, instead of the Euclidean Distance. So, for example, it can be based on the *Manhattan* (or *City Block*) *Distance*, or *Square Distance*.
(f) The CC has been implemented in fast electronic hardware [6]. It is possible to connect several boards together, to increase the number of subclassifiers almost indefinitely, without increasing the computation time at all.

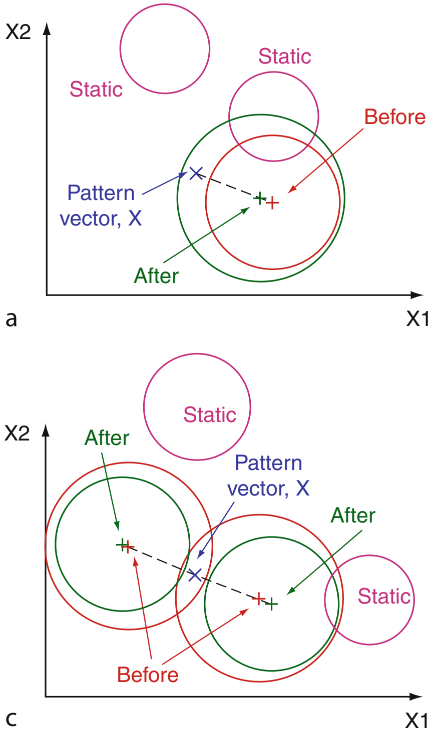


Fig. 16.22

Two-class learning rule for a CC. (a) When M disagrees with T (Teacher decision) and M = −1, the closest circle/sphere/hypersphere is moved towards the vector X and is made larger. Both movement and size changes are very small indeed. (b) When M disagrees with T (M = +1, T = −1) and only one of the subclassifiers produces a decision C_i = +1, that circle/sphere/hypersphere enclosing X is moved away from X and is made smaller. (c) As (b) except that more than one circle/sphere/hypersphere encloses X. In this case, all of them are treated in the same way as described earlier. This process is repeated iteratively, with X vectors selected at random. Additional rules have been devised for adding/removing subclassifiers, as needed [1].

(g) Conceptually, the CC is attractive as it is able to recognise a single compact class among a mass of others. Hence, it is ideally suited to recognising anomalies: a group of hypersphere sub-classifiers encloses the “normal” (i.e., “good”) class. Any deviation outside that region, in whatever direction, is “wrong” and should be signalled as an error.

(h) There is no reason why the inputs to the CC should not be the HSI, YUV or some other colour parameters, instead of the RGB signals that we have proposed. However, little is to be gained by this and it necessarily involves more computation in converting from one set of colour coordinates (RGB) to another. Any two sets that are distinguishable in RGB space are also distinguishable in HSI space, since the transformation between them is reversible. The shape of the surface that separates them will be different. At worst, we will have to use more hypersphere subclassifiers in RGB space but we are saved the effort of converting from HSI to RGB colour coordinates.

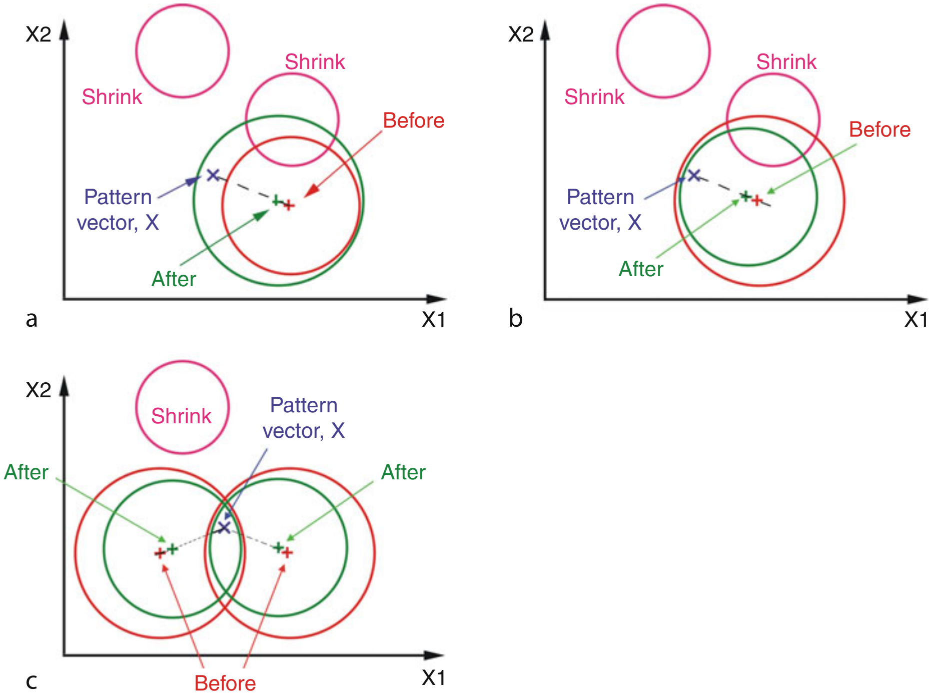


Fig. 16.23

One-class learning rule for a CC. In practice, all changes are much smaller than shown here. (a) When M = −1, the closest circle/sphere/hypersphere is made larger and is moved towards X. All other circles/spheres/hyperspheres are made smaller. (b) When M = +1, and just one circle/sphere/hypersphere encloses X, it is moved towards X and is made smaller. All other circles/spheres/hyperspheres are made smaller. (c) When M = +1 and more than one circle/sphere/hypersphere encloses X, they are all moved towards X and are all made smaller. All other circles/spheres/hyperspheres are also made smaller

16.5.2 Nearest Neighbour Classifier

Mathematically, the NNC is closely related to the Compound Classifier and similar learning rules have been defined. However, the attraction of the NNC is the possibility that exists for an extremely simple design procedure that can make use of very small amounts of data. Like the CC, it has a strong conceptual appeal that owes its origins in the inverse relationship that we instinctively feel should exist between similarity and distance. (This is the *Compactness Hypothesis* and is a central tenet of Pattern Recognition.)

The NNC also stores a set of Q-dimensional vector reference vectors (locates), which we shall again denote by $\mathbf{X}_i$ ($i \in [1,N]$). The Euclidean distance $D(\mathbf{X},\mathbf{X}_i)$ from the input vector $\mathbf{X}$ to each of the $\mathbf{X}_i$ is computed. Each of the $\mathbf{X}_i$ is associated with a particular class label, H_i. (Typically, $H_i \in${“red”,“green”, “yellow”, ...}.) The decision of the NNC is H_j if

$$D\left(\mathbf{X},\mathbf{X}_j\right) < D(\mathbf{X},\mathbf{X}_i), \forall j \neq i$$

(See ➲ *Fig. 16.24*)

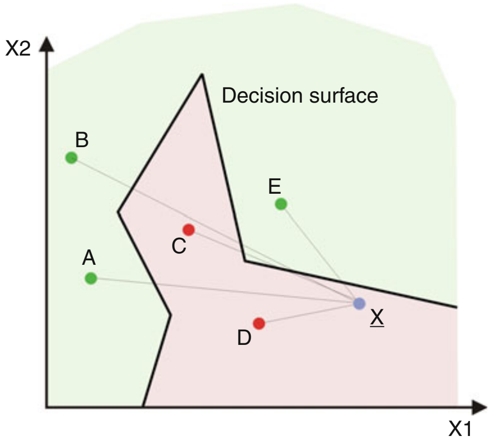


Fig. 16.24
Nearest Neighbour Decision Rule. Locates A, B and E are representatives of class 1; locates C and D are representatives of class 2. X is closer to locate D than it is to any other locate. Hence, it is classified as belonging to class 2, as are all other points in the pink region. The decision surface is piece-wise linear. In 3D space, it consists of planar segments

There are several ways to calculate suitable values for the locates:

(a) For very simple situations where it is clear that there is only one compact cluster for each colour category, the $\mathbf{X}_i$ may be taken to be the means of those classes. This will only be effective if all intra-cluster distances are smaller than every inter-cluster distance. That is, if **X** and **Y** are both associated with class H_i but **Z** is associated with class H_j $(i \neq j)$ then the following condition should be met

$$D(\mathbf{X}, \mathbf{Y}) < D(\mathbf{X}, \mathbf{Z}).$$

(see ➲ *Fig. 16.25*)

(b) In those situations where the colour classes are well separated but do not satisfy the condition just defined, it is possible to employ any one of a variety of cluster analysis techniques. For example, using a procedure called *Maximindist* (described below), it is possible to identify a small set of points that are guaranteed to be representative of each cluster. An elongated cluster is represented by a set of points scattered along its longest axis (➲ *Fig. 16.4e*).

(c) Iterative rules for moving the locates when the decision is wrong have been devised for the NNC. These are very similar to those already detailed for the CC and are summarised in ➲ *Fig. 16.26*. The adjustment made in response to an individual error is very small. Learning is made faster by starting the process with a set of locates found by Maximindist.

A few points should be noted about the NNC:

(i) It is unwise to use any metric other than the Euclidean distance, as the decision surface is unstable under certain conditions.

(ii) The decision surface of the NNC is piece-wise linear.

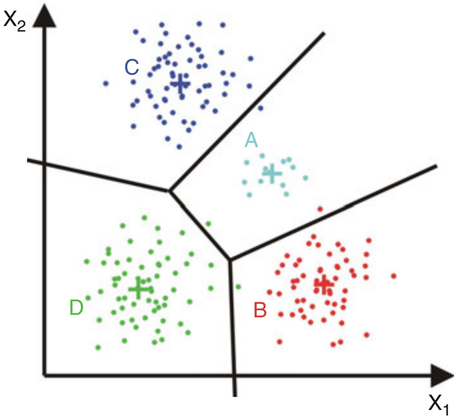


Fig. 16.25

Using class means to define the locates for an NNC is only effective if the colour classes are small, compact and are well separated from one another. In this case, there are some errors, since cluster diameters are not taken into account.

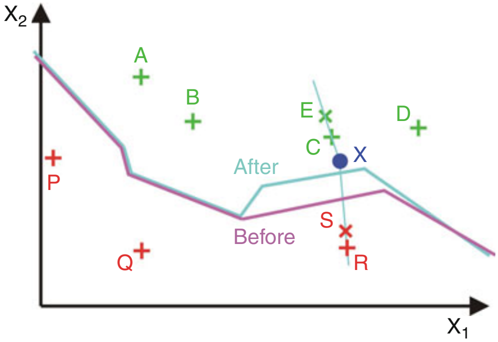


Fig. 16.26

Error correction learning in an NNC. A, B, C, D are locates representing class 1, while P, Q, R represent class 2. The pattern vector X is assumed to have been misclassified as belonging to class 1 instead of class 2. The locate closest to it (C) is moved a short distance away from it (to E), while the closest locate from class 2 (R) is moved a little closer (to S). Although X is not reclassified as a result of this single adjustment of the locate positions, the movement is exaggerated here for clarity. This process is repeated iteratively, with X vectors selected at random

(iii) Any decision surface generated by a multi-layer neural network of Perceptron-like devices can be implemented by an NNC.

(iv) It is possible to implement the NNC using a look-up table (LUT) to perform subtraction and squaring (in a single step) and without taking square roots. This can assist both hardware and software implementations (➜ Chap. 17).

(v) Sometimes, it is possible to identify possible locates by simple visual examination of the three-dimensional scattergram of RGB values.

(vi) The NNC can be applied directly to the RGB signals generated by a colour camera, or to the HSI/YUV parameters. (Some linear rescaling may be necessary to ensure that each input to the NNC has the same range.)

Both the CC and NNC can produce good results in colour discrimination tasks, provided enough care in taken in their design/learning phases.

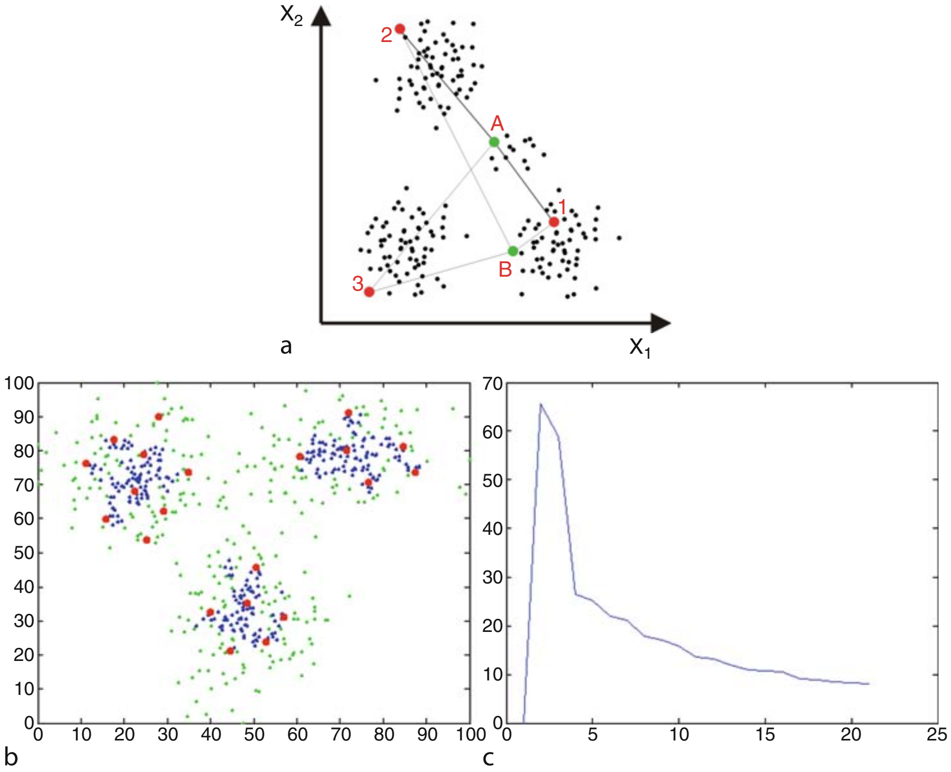


Fig. 16.27

Removing outliers and the Maximindist procedure. (a) Maximindist takes its name from the fact that it relies on finding the maximum value of the minimum distance between a candidate point and those points already stored, in this case {1,2,3}. When trying to find the fourth point, A is preferred to B because it is further away from its closest neighbour (1) than B is from its closest neighbour (also 1). The procedure begins by choosing the first point (1) at random from the given data set. (b) Green: points lying in three clusters, generated using pseudo-random number generators. Blue: points remaining after outliers were removed. (K = 1) Red: 20 points remained after Maximindist was run on the set of blue points. (c) Maximin distance plotted against the number of stored points

16.5.3 Removing Outliers and Maximindist

A point is regarded as an outlier if the distance between it and its nearest neighbour is greater than some pre-defined constant (K) times the average distance between all other points in the data set and their nearest neighbours. Typically, K is about 1 (➲ *Fig. 16.27b*).

It is much easier to appreciate Maximindist from a simple diagram, rather than trying to define it in formal mathematical terms (➲ *Fig. 16.27*).

References

1. Batchelor BG (1974) Practical Approach to Pattern Classification. Plenum, London/New York
2. Grassmann's Law (2009) Wikipedia, http://en.wikipedia.org/wiki/Grassmann%27s_law_(optics). Accessed 1 Dec 2009
3. http://www.brucelindbloom.com/index.html?ColorCalculator.html. Accessed 23 June 2006; http://www.easyrgb.com/math.html. Accessed 23 June 2006; http://www.cs.rit.edu/∼ncs/color/a_spaces.html. Accessed 18 July 2006
4. MATLAB, Mathworks, Inc. http://www.mathworks.com. Accessed 26 Nov 2009
5. Plummer AP (1991) Inspecting coloured objects using grey-scale vision systems. In: Proc. SPIE conf, Boston, MA, Nov 1990, vol CR-36, pub as "Machine vision systems integration," SPIE, Bellingham, WA, pp 64–77, 1991, ISBN 0-8194-471-3
6. Zero Instruction Set Computer (2009) Wikipedia http://en.wikipedia.org/wiki/Zero_instruction_set_computer. Accessed 17 Dec 2009

17 Algorithms, Approximations and Heuristics

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_17,

Abstract: When a vision procedure has been devised, perhaps using an interactive image processing system, it must be engineered before it is ready for use in a factory floor system. A prototype procedure found in this way has probably not been optimised for speed, compatability with other parts of the system, or ease of implementation. It may not even be the most effective procedure of the same general type. For example, it might employ one image filter, such as an edge detector, whereas another would perform just as well and be much easier to implement. This chapter explores a range of grey-scale and binary image processing functions, demonstrating that many algorithms present parallel options. It is well to investigate every stage of an image processing procedure to see if it can be replaced by a more convenient, or more efficient, alternative. Each part of an image processing procedure must be compatible with all of the others. Sometimes, the way an image is represented can make a large difference in the ease of implementation, or speed of execution. On some occasions, a mathematically rigourous procedure (algorithm) can be replaced by a much more convenient "rule of thumb" (heuristic), without serious loss of performance. If used sensibly, heuristics may make a system work in a very much more cost-effective way than would otherwise be the case. Heuristics can sometimes replace algorithms, with little or no loss of performance. In some instances, heuristics can greatly improve the speed compared to an algorithm, so that slightly reduced accuracy is acceptable. It is important to bear in mind what the real objective of a vision system is. Suppose we wish to build a machine to count large numbers of objects like dice. Should we count white rectangles, representing the top surfaces of the dominoes, or black spots and then divide the result by three? On a sufficiently large sample, these two procedures produce similar results. Thinking in terms of statistical performance is an important part of designing an effective vision system. Heuristics can often be made to cope safely with situations that have never been seen before. A "catch all" rule, such as "*If all else fails, switch off the power,*" can guarantee the safe operation of a complex plant, even if it occasionally reduces production efficiency unnecessarily. Emotional objections to procedures that are know to fail occasionally can sometimes be very strong but it is well to consider the frequency of such an event. However costly a mistaken decision is, its overall significance may be small if it occurs very infrequently. Heuristics can often be biased deliberately to improve safety, or to improve productivity. One particularly useful heuristic rule for an automated inspection system is to refer doubtful cases to a human being for examination and/or rework. There is another very useful heuristic that we should always apply: try to improve the quality of the "input" images by adjusting the lighting-viewing conditions!

17.1 Introduction

This chapter is based on three axioms:

1. Machine vision is an engineering discipline, not a mathematical or philosophical exercise.
2. There is no unique way to perform any given image processing function; every algorithm can be implemented in many different ways.
3. A given calculation may be very difficult using one implementation technology but very easy with another.

Given a certain method of implementation, simply reformulating an algorithm can often convert a "difficult" calculation into one that is straightforward, perhaps even trivial. It is important, therefore, that we gain some understanding of the way that algorithms can be reformulated before we consider the choice of implementation technology.

In every engineering design exercise, there are numerous factors specific to the task in hand. These must be carefully considered and properly balanced if we are to achieve a robust and effective system. In the case of machine vision, these include the shape, material and optical properties of the object that is to be examined. Moreover, the modus operandi of the inspection system, its environment within the factory, and the skills and attitudes of the people who will work with it all have a critical role in system design. There are three critically important constraints, which are present in almost every engineering project. These relate to cost, performance and operating speed. Another common constraint is the desire to use technology that is familiar, tried and trusted. Together, these constraints may well preclude the use of certain implementation techniques and hence certain algorithms too. We cannot simply rely on mathematical analysis alone to provide sufficient commendation for using a given algorithm. This point is of fundamental importance for our discussion in this chapter. Our experience has shown that operational, management and performance constraints, such as those just listed, almost invariably impose far tighter control than "academic" discussions about whether we should use, say, one image filter rather than another.

We should never take it for granted that an algorithm must be performed in a given way. In particular, we should not expect that the initial explanation we gave in ➋ Chap. 14 necessarily defines how we are to implement a given algorithm. Indeed, we should always ask ourselves whether the algorithm is expressed in the best way possible and question whether reformulating it would lead to a better implementation. We should also investigate whether a similar algorithm, an approximate method, or an heuristic procedure is good enough for the given application. In the context of designing machine vision systems, *sufficiency*, rather than optimality, is the key concern. We must always be very wary of claims about optimality, because these are almost invariably based on some spurious assumption about what constitutes an "optimal solution." As in any engineering design process, all assumptions must be based on experience, judgement and common sense. We must not rely blindly on mathematical rigour, since mathematics only very rarely provides a true model of the real physical world. An old adage states that "*If you can perform the mathematics, then you don't understand the engineering.*" This is, of course, a gross simplification, but it paraphrases a vitally important truth. The reader should make no mistake about the authors' commitment to using rigourous mathematical analysis, wherever and whenever it is appropriate to do so. The most important point to note, however, is that there is a balance to be obtained: using algorithms, heuristics and approximate methods in a sensible reasoned way can lead to great improvements in the overall efficiency of a machine. The two latter categories of computational procedure may provide the basis for the design of a cost-effective machine vision system that is reliable and robust. On the other hand, mathematical analysis often yields deep insights that are not intuitively obvious. We must not forget, however, that as far as an engineer is concerned, one of the most foolish uses of mathematics is to prove what has been self-evident for years. More irritating still is the abuse of mathematics by using an incorrect or deficient physical model to "prove" that certain things are not true in the real world. To illustrate this point, we need only remind the reader that for many years the known laws of aero-dynamics were not able to "prove" that bumble bees can fly [11]. In other words, formal mathematical analysis based on erroneous/deficient physical or engineering models would lead us to believe that bumble bees cannot fly. The "proof" that this is not so is easily obtained: simply go into a garden on a summer’s day. Sticking doggedly to only those algorithmic techniques with absolutely certain mathematical pedigrees is both limiting and likely to give us a false sense of confidence.

The choice of metric used to quantify algorithm performance almost inevitably involves our making an arbitrary and oversimplified choice. This can seriously affect the conclusions we reach about the performance of a vision system. There is an outstanding example of this in a paper that we encountered recently. An “objective” performance criterion was defined in terms of an *arbitrarily chosen* metric. A number of image filtering methods were then analysed mathematically. As a result, one of them was found to be “superior” to all others. If a slightly different algorithm performance criterion had been selected, then a different filter might have been judged to be “best.” In some situations, the ability to implement an image filter using an electronic circuit module with a power consumption of less than 100 μW might be a more appropriate criterion to apply, rather than some abstract mathematical performance metric. The latter might, for example, be based on the familiar concept of “least squares fit,” but there cannot be any guarantee that this is in any way meaningful in any absolute sense. The reader should understand that we are not arguing that mathematical analysis has no place in the design of machine vision systems. It most certainly has a very important role, but mathematics must take its place alongside many other algorithm selection criteria, in order to obtain a balanced engineering compromise. We are simply arguing that we must not judge the design of a machine vision system using any one-dimensional cost performance criterion. (Even “cost” is a multi-dimensional entity.)

Our objective in this chapter is to explore the possibilities for expressing vision algorithms in a variety of ways. There is also the possibility of using approximate computational methods. Finally, we point out that techniques that are known to “fail” in some situations (i.e., heuristics) can sometimes provide a perfectly acceptable solution to the broader engineering design task.

We will find it convenient to use short mnemonic names to denote what might be quite complex image processing operations. For this reason, frequent use is made of the names of functions implemented in the QT image processing language described in ➲ Chaps. 21 and ➲ 41. The operations of these functions will be described as we encounter them here.

17.1.1 One Algorithm, Many Implementations

We will begin by discussing ordinary arithmetic multiplication. This will allow us to introduce the main points that we wish to emphasise in this chapter. By discussing an apparently straightforward task, that is clearly relevant to many different areas of computation, in addition to machine vision, we hope to explain why we felt it necessary to include this chapter in our book.

Consider the deceivingly simple task of multiplying two numbers together:

$$C = AB \tag{17.1}$$

Multiplication may be performed using a rather complicated digital hardware unit based on an adder, shifter and recirculating data register. In some situations but by no means all, this may be the most convenient implementation technique. Special purpose integrated circuits, based on this idea or arrays of gates, are readily available but in certain applications may not provide the best solution.

Logarithms can, of course, be used to replace the process of multiplication with the simpler operations of table look-up and addition. We can use the well-known relationship:

$$C = \text{antilog}(\log(A) + \log(B)) \qquad (17.2)$$

If the variables A and B are short integers, the log and antilog functions can be performed conveniently using look-up tables, implemented in ROM or RAM (➲ *Fig. 17.1*). The reader will, no doubt, have already anticipated the complications that arise when the variables A and/or B are allowed to assume values close to zero. However, we will not dwell on this topic here.

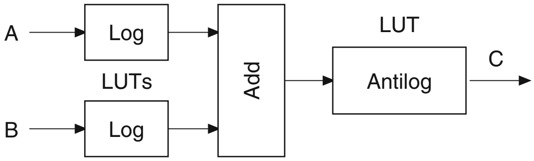
Another possibility is to use the so-called *quarter-squares method*, which makes use of the following identity:

$$A\,B = \left[(A + B)^2 - (A - B)^2\right]/4 \qquad (17.3)$$

Suppose that we are proposing to use an implementation technology which finds it easy to perform addition, subtraction and squaring, perhaps by using a look-up table (➲ *Fig. 17.2*). In this case, the quarter-squares method can be very effective indeed. (Older readers may recall that this method was used in the heyday of analogue computers to perform multiplication.) In certain circumstances, the quarter-squares or logarithm methods, implemented using look-up tables, might be more effective than the direct implementation of➲ Eq. 17.1.

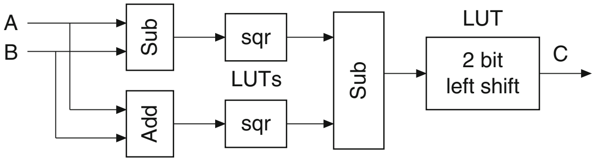
So far, we have tacitly assumed that both A and B are variables. However, if either of them is a constant, the implementation can be simplified quite significantly. If we wish to multiply a given number, represented in digital parallel form, by a constant, the use of a simple look-up table implemented using a Read Only Memory (ROM), is probably the simplest implementation technique (➲ *Fig. 17.3*).

Suppose, however, that the constants are always integer powers of 2. In this case, "multiplication" reduces to a simple bit shift operation, which may be implemented using nothing more complicated than a piece of wire! (➲ *Fig. 17.4*).



■ **Fig. 17.1**

Multiplying two numbers, A and B (typically 8-bit parallel) using logarithms. The *log* and *antilog* functions are implemented using look-up tables, possibly using ROMs



■ **Fig. 17.2**

Multiplying two numbers, A and B (typically 8-bit parallel) using the quarter-squares method. The *sqr* (square) functions are implemented using look-up tables, possibly using ROMs

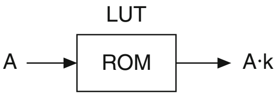


Fig. 17.3
Multiplying a number (typically 8-bit parallel) by a constant

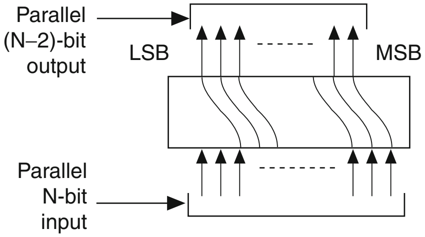


Fig. 17.4
Multiplying a number (typically N-bit parallel) by a constant of the form 2^M, where M is an integer. The figure shows the configuration for M = −2 (i.e., dividing by 4)

The reader might well question whether this discussion has any direct significance for image processing. To see one straightforward example where this relevance can be demonstrated, consider the arithmetic operations needed to implement the linear local operators, viz:

$$E \Leftarrow K_1(W_a A + W_b B + W_c C + W_d D + W_e E + W_f F + W_g G + W_h H + W_i I) + K_0 \quad (17.4)$$

At first sight this seems to indicate that we need to use 10 multiplications and 10 additions. Do we really need to use floating-point, or long-integer, arithmetic? Can we work with, say, an 8-bit representation of intensity (i.e., the variables, A, B, ..., I)? Can we obtain sufficiently accurate results using weights (W_a, W_b, ..., W_i) that are all integer powers of 2? If so, the problem reduces to one where we could use just one multiplication (i.e., K_0 (...)) and nine shift operations. Even if we must use general integer values for the weights, we can probably still manage to use a series of nine look-up tables, implemented using ROM/RAM, to compute the intermediate products $W_a A$, ..., $W_i I$ (➋ *Fig. 17.5*).

17.1.2 Do Not Judge an Algorithm by Its Accuracy Alone

We cannot afford to make a decision about which algorithm to use on the basis of its performance alone. To do so would be nothing short of stupid! There are many different technologies available to us when we come to build a vision system. Indeed, in a single vision system there are inevitably several different image/information processing technologies coexisting and cooperating together. These include optics, opto-electronic signal conversion (camera/scanner), analogue electronics, digital electronics, software. They all have their own peculiar strengths and weaknesses. An image processing algorithm is a mathematical prescription for a computational process which describes the operation of part of the data-processing chain. (When we use the term "image processing algorithm" in connection with a vision

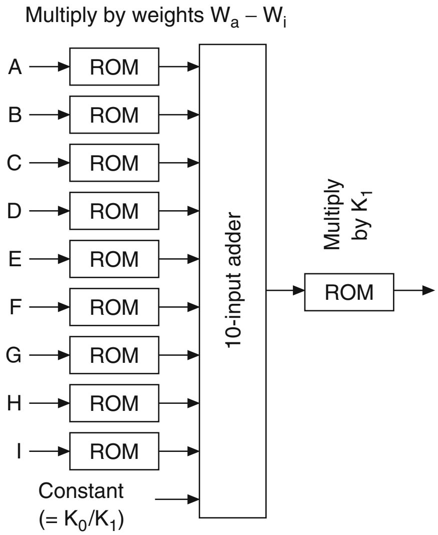


Fig. 17.5
A ROM-based implementation of the 3 × 3 linear local operator (Eq. 17.4) This diagram does not explain how the inputs A, B, ..., I are generated

system, we normally intend it to refer to the later stages in this chain, probably just the digital electronics and the software.)

Clearly, the speed of a machine is of great importance, but even this seemingly simple term has complications, because there are two components:

1. *Throughput rate.* (Number of images processed, or objects inspected, per second.)
2. *Latency.* (Time delay from digitisation of an image to appearance of result.)

An on-line inspection system, working in conjunction with a continuously-moving conveyor, can usually tolerate a high latency, whereas a robot vision system cannot [4]. Hence, the term “real time” needs some qualification, in practice.

A popular way of representing algorithm complexity uses the O(N) notation. Suppose that a certain algorithm requires an execution time of

$$A_0 + A_1N + A_2N^2 + A_3N^3 + \cdots + A_mN^ms,$$

where the A_i ($i=1, ..., m$) are constants and N is the number of data samples. (N is very likely to be related to the number of pixels in the image.) When N is very large, the term A_mN^m dominates all others and hence the algorithm is said to be of complexity $O(N^m)$. We do not always find that a simple polynomial function describes the execution time. For example, certain number-sorting algorithms require an execution time that is proportional to N log(N) (for very large values of N) and hence are said to have complexity O(N log(N)). While some number-sorting algorithms have a higher complexity than this (typically $O(N^2)$), they may actually be easier and faster to implement in digital electronic hardware. For example, a simple *bubble-sort* algorithm (complexity $O(N^2)$) can be implemented in an iterative digital array. *Figure 17.6* shows the organisation of a hardware unit for implementing 3 × 3 *rank filters*

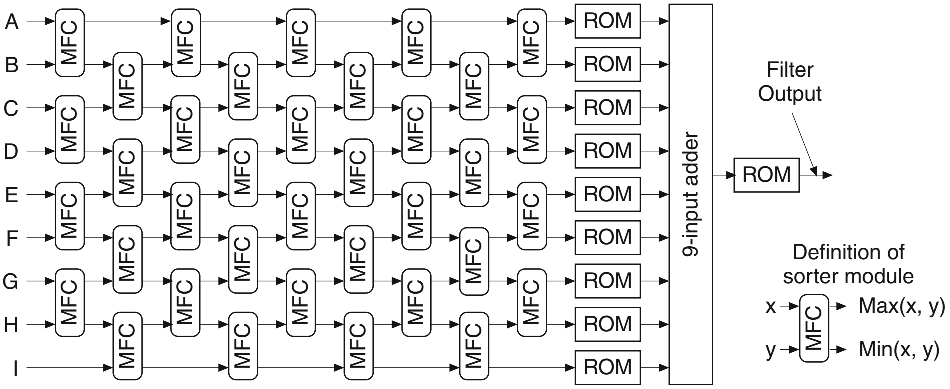


Fig. 17.6
Rank filter implemented using an iterative array performing a 9-element *bubble sort*

(Chap. 14), which require the ranking of a small number of numeric items. This iterative array uses the ideas implicit in the bubble sort. On the other hand, the *merge sort* has complexity O(N · log(N)) and for this reason is often preferred in software implementations. However, this cannot be implemented in digital hardware in such a neat manner. The general point to note is that, although the O(N) notation is very popular with mathematicians, it is not always a useful indicator of speed in a practical system.

17.1.3 Reformulating Algorithms to Make Implementation Easier

We have already made the point that there are many ways to implement a given algorithm and, in the latter parts of this chapter, we will give several examples to illustrate this. There are likely to be many similar image processing algorithms resembling a given procedure. For example, there is a wide range of edge detection operators [e.g., QT operators *sed, red, gra* and *edd*]. The point has been made elsewhere that, if the choice of edge detection operator is critical, then there is a problem somewhere else in the vision system – probably in the lighting! (Appendix C) Bearing this in mind, we may be able to improve the quality of the input image by choosing the best possible lighting, optics, viewing position and camera, and thereby allow ourselves the luxury of using a sub-optimal image filter that is easy to implement.

An experienced vision engineer very often realises that one algorithmic "atom" can be substituted for another in a processing sequence, producing very little effect on the overall system performance. To evaluate possible substitutions of this kind, it is essential to work with a convenient (i.e., *interactive*) image processing "tool kit." An experienced system designer may well be able to anticipate which algorithms can safely be substituted in this way, but "blind" predictions like this are frequently mistaken. To be certain about the wisdom of employing the proposed revised algorithm, it must be assessed in a rigourous experimental manner. Much of the knowledge about what substitutions are sensible might, one day, be incorporated into an expert system but, to date, this has not been done, except at the most trivial level. At the moment, an experienced engineer armed with appropriate tools (such as an interactive image processor like QT or NEATVISION (➲ Chaps. 21 and ➲ 22)) remains the best way to perform the high-level reasoning needed to design/redesign a vision algorithm.

In the previous chapter, we introduced several different types of image representation. As we will soon witness, these are particularly useful because certain algorithms can be reformulated very effectively using one form of image representation (e.g., chain code, run-length code, etc.), rather than the standard array representation. A prime example of this is to be found in the algorithms available for computing the convex hull of a blob-like figure. (Incidentally, this can be implemented even easier using the polar vector representation. See ➲ Chap. 14)

17.1.4 Use Heuristics in Lieu of Algorithms

A heuristic is a "rule of thumb" which is known to achieve a desired result in most cases, without necessarily being able to guarantee that it is always able to do so. On the other hand, an algorithm is always able to achieve the desired result. Why should we be prepared to accept a procedure which is known to fail in some situations? In our normal everyday lives, we use heuristics freely and without any serious misgivings. For example, a person drives to work each morning along a road that experience has shown is able to get him there quickly. However, there are exceptions: one day last week, a car had broken down on the road and the author of this chapter was delayed for several minutes before he could complete his journey. That day, his heuristic for getting to work on time failed but it worked well this morning and it does so on most working days. There simply is no algorithm for getting to work on time. It is clearly impossible, in any practical way, to collect all of the information that would be needed to predict accidents or the breakdown of vehicles on the author's route to work.

17.1.5 Heuristics May Actually Be Algorithmic

Heuristics can be very accurate and may actually be algorithmic without our knowing it. (We may not be clever enough to do the necessary mathematics to prove that a given procedure is algorithmic.) Very general heuristic rules can often cope with unexpected situations more comfortably and often more safely than "algorithms." Once we violate the conditions of use of an algorithm, we cannot guaranteed that it will generate valid results. Mathematical criteria defining conditions of use exist for good reason. If we violate them, we cannot properly anticipate the results. We might, for example, wish to relax the Dirichlet conditions for Fourier series expansion. It is highly unlikely that we could ever know for certain whether the result is mathematically valid, although by experimentation and/or computer simulation, we may be able to satisfy ourselves that it is "good enough." The important points to note are that what we regard as an algorithm may not be so, and that a procedure that is not yet proven to achieve a "valid result" in every situation may in fact do so. The reader will observe a parallel here with Gödel's theorem. There are provable truths (algorithms that are known to be true), unprovable truths (procedures that are not known to be algorithmic but are true), unprovable untruths (e.g., procedures that fail in some as yet undiscovered way), and provable untruths (heuristics that are known to fail occasionally). However, we should not take this analogy with Gödel's theorem too far, since it has no rigourous basis in logic.

17.1.6 Primary and Secondary Goals

Suppose that we are faced with the task of designing a machine that counts coffee beans, given an image such as that shown in *Fig. 17.7a*. Now, we know that we can count blobs in a binary image using the QT operator *cbl*. This, combined with the prior use of a thresholding operator (*thr*), seems highly appropriate, since the latter converts the grey-level input image into a binary image. A simple experiment (*Fig. 17.7b*) soon shows that these two operators [*thr(_), cbl*] are unable to provide the needed precision. So, we soon conclude that we should have used some suitable image filtering (smoothing), to pre-process the images before thresholding. This seems a very reasonable approach and is certainly worth investigating further. However, we must be clear that the real (i.e., primary) objective is counting coffee

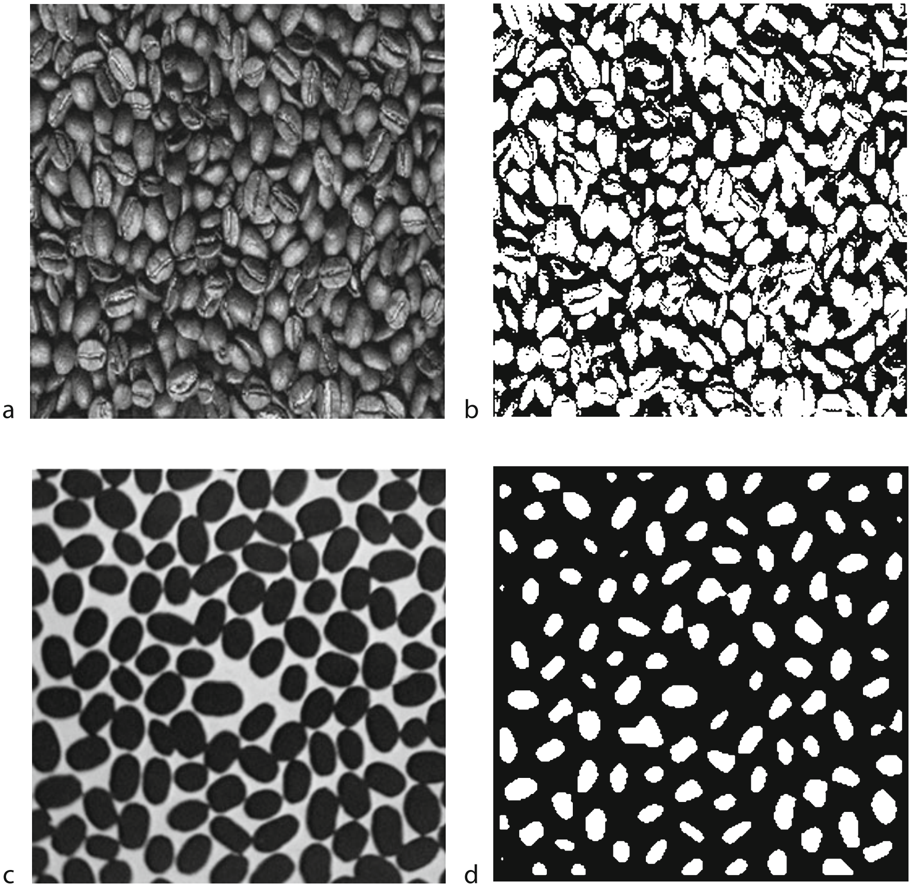


Fig. 17.7

Counting coffee beans. (a) Original image. Front lighting yields an image that is difficult to process. (b) Simple fixed parameter thresholding. Filtering can improve matters but not well enough to give a reliable count. (c) A different original image. Back lighting is superior. (d) Better separation, obtained using *[QT: ecn, thr, neg, gft, enc, raf, thr (100)]*

beans, not blobs in a binary image. In order to make some progress, we were forced to make *some* assumptions but, as we will show, this one is not the best. In this instance, the assumption we have made is that the primary task (counting beans) can be replaced by two or more secondary tasks: filtering, thresholding and blob counting. This is a reasonable working hypothesis but nothing more. After struggling for a long time to find a suitable filter, most people would probably conclude that they cannot generate a suitable binary image for this process to be reliable. (It is impossible, in practice, to obtain a "clean" binary image in which each bean is represented by just one blob.)

Eventually, we reassess our initial assumption and realise that we do not need to produce one blob for every bean. In this type of application, a statistically-valid answer is acceptable. We need to reassess our initial assumption that *cbl* is the way to count blobs. Suppose that a certain filtering and thresholding operation together produces on average 1.53 white blobs per bean, with a small standard deviation. Then, we can count blobs using *cbl*, the Euler number (*eul*), or any of a number of other procedures. In other words, there is nothing special about *cbl*. We might well find that we can substitute other techniques and still obtain a statistically-valid result. (We may find, of course, that the scaling factor relating the "blob count" to the "bean count" has to be revised. Recall that this was 1.53 for the operator *cbl*.) Our initial assumption that *cbl* is "the way" to count blobs and that we could obtain just one blob per bean was inappropriate, even though it seemed so reasonable when we began. Few image processing operations have a better claim to being "algorithmic" than *cbl*, yet we are forced here to concede that this "guarantee of performance" means nothing; we are using an *algorithm* in a process which is being *judged statistically*. By substituting *eul* or another "blob count" procedure for *cbl*, we may be making little or no sacrifice in the overall performance of the bean counting system, or even conceivably improving it. The QT operator *cbl* counts eight-connected blobs, whereas we might have chosen to base the definition on four-neighbours. Whichever we choose, the decision is quite arbitrary. For practical purposes, when counting coffee beans we might well find that *eul* produces results that are just as accurate in estimating the bean count.

In practice, we would probably find it more convenient to alter the method by which beans are presented to the camera. The image processing task is then much simpler and more reliable (➜ *Fig. 17.7c*, *d*).

17.1.7 Emotional Objections to Heuristics

Many people have a serious emotional objection to using heuristics, because they cannot be guaranteed to give the correct result always. Such objections are more likely to arise when there is a known, often deliberately contrived, counter-example which makes a given procedure "fail." People often become confused at such a discovery, however unlikely or unimportant that "pathological" situation may be in practice. Even very intelligent people are often very poor at judging the statistical frequency and/or absolute importance of rare events. One has only to look at everyday life to see how frequently rational judgement is suspended, despite incontrovertible evidence to the contrary. (For example, the popularity of gambling and driving fast, as well as the fear of flying and crime provide four such examples.) If an inspection system is known to ignore or give an inappropriate response, even on very rare or unimportant cosmetic faults, many managers will be unhappy to accept it. This caution will be "justified" on the grounds that senior management will not condone installing a machine with *any* known shortcomings. This thought process is not always rational, since there may be no alternative solution that is able to provide the "correct" results with 100% certainty. The important point to note is that, in reality, there is no sharp dichotomy between algorithms which *always* work and crude, approximate heuristics which do not. The concept of an algorithm-heuristic continuum is a more appropriate model for the situation in the real world. Another point is that the performance of a vision system is often judged *statistically* not "*logically*." This is not a new idea, since we judge people on "statistical" evidence. All that we are asking is that machine vision systems and their component computational processes be judged in the same way.

17.1.8 Heuristics Used to Guide Algorithms

Heuristics can be used to guide algorithms. For example, we may use a heuristic simply to speed up an algorithm. In this case, when the heuristic fails, the algorithm simply slows down but still yields a known result. A prime example of this is to be found in hill-climbing. Heuristics can be very effective at finding local maxima, whereas an exhaustive search algorithm will always find the true maximum. Similarly, a heuristic may be used to speed up the process of finding "good" routes through a maze but an algorithm is able to guarantee that the final result is optimal. Heuristics are useful in those situations where knowledge of one good solution can lead to a rapid means of finding a better solution.

17.1.9 Heuristics May Cope with a Wider Range of Situations

A well-designed set of heuristic rules can often cope with a far wider range of situations than an algorithm is known to cover. This is possible because heuristics are able to encapsulate human experience, using the same language that people use to understand and express them. Such rules of thumb are usually not definable in strict mathematical form and are expressed in terms of a natural language, such as English. Normally such rules are then recoded into a computer language. (Prolog and hence PQT are very appropriate for use in this role. ➋ Chap. 23.) It is entirely inappropriate to try to force mathematics where it does not fit naturally.

Lastly, we simply pose an important question: Is an algorithmic solution really necessary? We often have to overcome ignorance, prejudice and ill-informed opinion in order to overcome the emotional barrier that prevents many people from accepting "solutions that are known to fail." When a non-technical manager is involved in making decisions about the use of heuristics rather than algorithms, there is always the danger that his natural inclination will be to reject heuristics in favour of algorithms, believing that the latter inevitably gives a better result. We hope to show in the pages that follow that this is often not the case.

17.2 Changing Image Representation

In ➋ Sect. 14.1 we discussed a number of ways of representing images in digital form. It will become clear later that some algorithms and heuristics can be expressed more naturally in one form rather than another. For this reason, we will explain first how images can be transformed from one representation to another.

17.2.1 Converting Image Format

The array representation arises naturally from standard array-scan cameras and line-scan cameras which are repeatedly scanned to build up a 2-dimensional image. It is natural to associate each photo-detector site on the camera with one pixel in the digital image. The array representation therefore represents a reasonable starting point for image processing function, and the other coding methods can be derived from it.

17.2.1.1 Array Representation to Run Code Representation

The run code (or run-length code) is a more compact form of representation for binary images than the array code and can be derived from a video signal using fast low-cost electronic hardware (see ➋ *Fig. 17.8*). The operation of this circuit is quite straightforward: Counter 1 indicates how many pixels have been encountered in a raster scan through the image since the last black-white or white-black intensity transition. This value is stored in the RAM. (The box labelled "MUX" selects the input arising from Counter 1.) This counter is reset at the beginning of each new line, and whenever a black-to-white or white-to-black intensity transition occurs. Counter 2 indicates how many items have been stored in the RAM. Notice that a special symbol, called the *inter-line marker*, is stored in the RAM to indicate the end of each row of the image. This symbol is presented to the RAM's *data* input via, the multiplexor (MUX), when the *line sync/new line* signal is high. The format for data stored in the RAM is as follows:

$$\{R_{1,1}, R_{1,2}, \ldots, R_{1,n1}, *, R_{2,1}, R_{2,2}, \ldots, R_{2,n2}, *, R_{m,1}, R_{m,2}, \ldots, R_{m,nm}\}$$

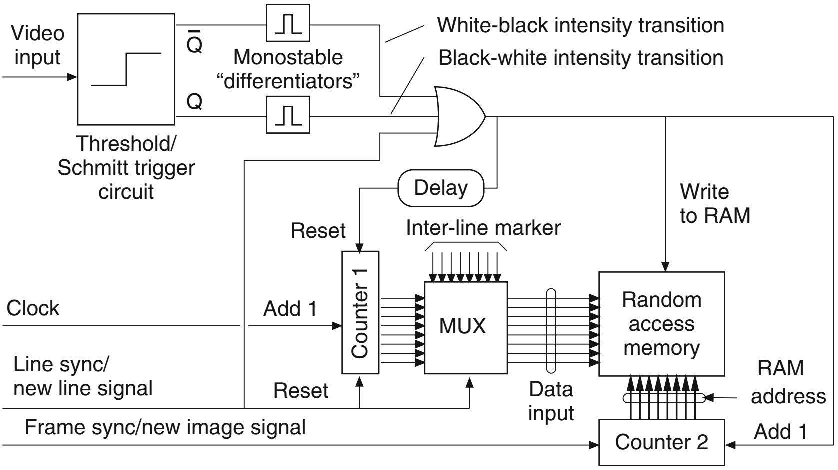

Video input
$\bar{Q}$
Q
Threshold/ Schmitt trigger circuit
Monostable "differentiators"
White-black intensity transition
Black-white intensity transition
Delay
Reset
Inter-line marker
Write to RAM
Clock
Add 1
Counter 1
MUX
Random access memory
Line sync/ new line signal
Reset
Data input
RAM address
Frame sync/new image signal
Add 1
Counter 2

Fig. 17.8

The run code can be generated in simple digital or analogue hardware. A simple low-cost circuit such as this can perform run-length encoding in real time on a video signal. This is important because the run code transforms many "difficult" calculations into trivial ones

where $R_{i,j}$ denotes the ith run in row j and ‘ ∗ ’ denotes the Inter-line marker. Notice that $R_{i,1}+R_{i,2} \cdots + R_{i,ni} \le N$, where N is the number of pixels along each row. The same circuit can be used for both analogue and digital video inputs with very little modification.

17.2.1.2 Array Representation to Chain Code Representation

The chain code can be derived from the array representation of a binary image in at least two ways:

(a) Perform a raster-scan search for an edge. Then, follow that edge until the starting point is revisited. This process is repeated until all edges have been traced (➋ *Fig. 17.9*). In this procedure, each edge point is visited not more than twice. If, as in some applications, we can be sure that there is only one object silhouette (i.e., only one blob) in the binary image, then the algorithm need visit only $(m \cdot n+N_{edge})$ pixels, where the image resolution is $m \cdot n$ pixels and there are N_{edge} edge pixels.

(b) Use pre-processing to label each edge point in the image. The labels (➋ *Fig. 17.10*) indicate the chain code. (Some special symbol is used to indicate non-edge points.) Then, trace the edge to “thread” the chain code elements together into a string [3].

While procedure (b) seems to be more complicated, it is well suited to implementation using dedicated hardware. Either a parallel processor or a pipe-line processor can perform the initial edge labelling at high speed. The latter can, for example, perform the initial edge coding in a single video frame-period. There then remains a serial process in which N_{edge} pixels are visited, as they are “threaded” together.

In algorithm (b), tracing the edges is performed by a serial process, which uses the binary neighbourhood coding scheme explained in ➋ *Fig. 17.10*. The preprocessing engine, whether it be a pipe-line or parallel processor, assigns a number (a pseudo-intensity value) to each pixel to describe its eight-neighbourhood. The serial processor then uses this value as an index to a look-up table, in order to locate the next edge point in the edge to be chain-coded. The same binary neighbourhood coding will be encountered in several places later in this chapter.

17.2.1.3 Chain Code Representation to Polar Vector Representation

The polar vector representation (PVR) can be derived quite easily from the chain code. To understand how, let us consider a typical chain code sequence:

. . .,2,2,2,3,3,3,3,3,3,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,3,3,3,3,4,4,4,5,5,5,5,5,. . .

Consider ➋ *Fig. 17.11*, from which it can be seen that a chain code value 2 is equivalent to a movement around the edge of one unit of travel at an angle of 90°. In polar vector form, this can be represented as (1, 90°). However, a chain code value 3 is equivalent to a movement around the edge of √2 units of travel at an angle of 135° (√2, 135°). The chain code sequence just listed is therefore equivalent to the following sequence of polar vectors:

. . ., (3, 90°), (6√2, 135°), (16, 180°), (4√2, 135°), (3, 180°), (5√2, 225°),. . .

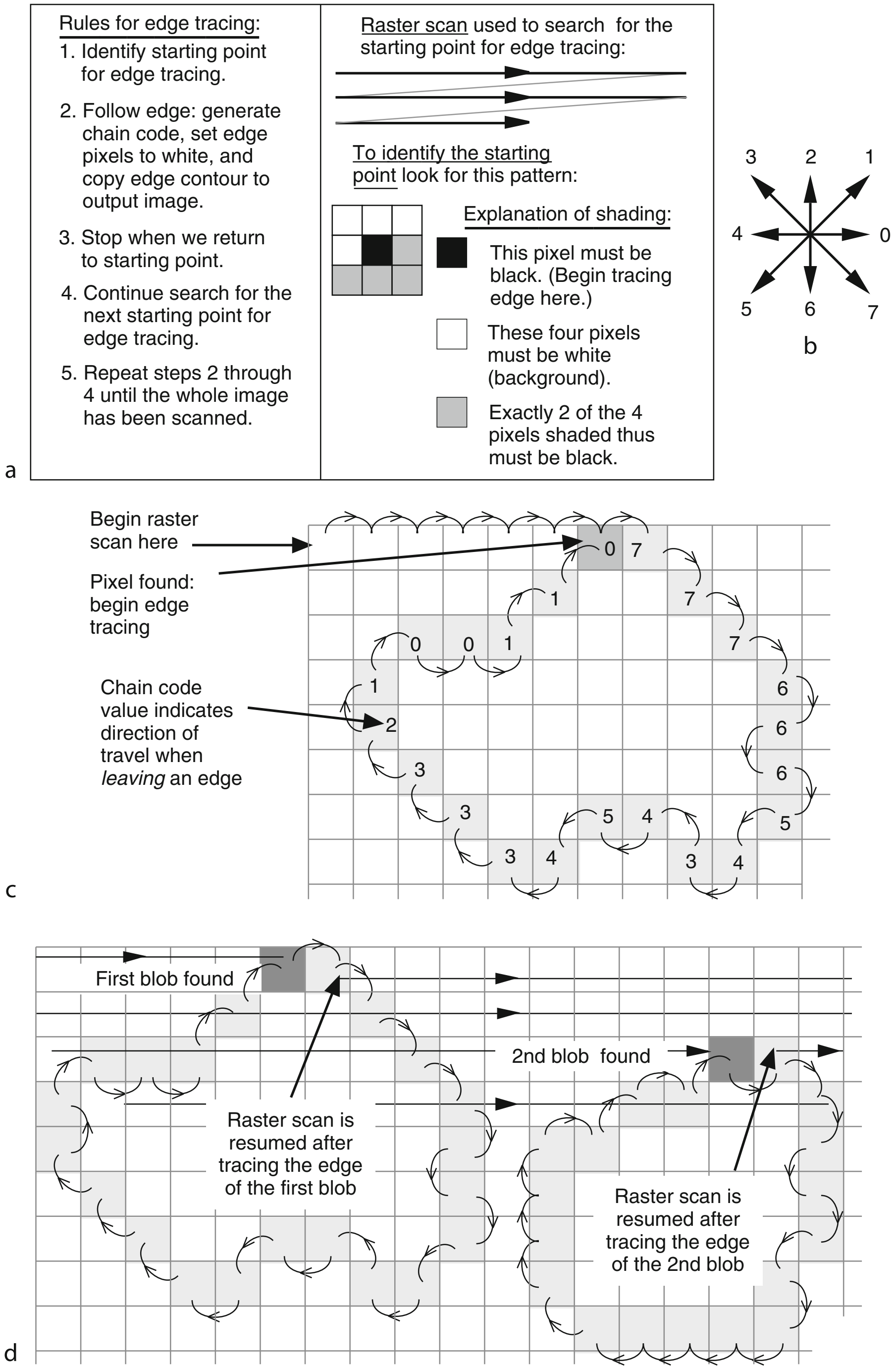

Rules for edge tracing:
1. Identify starting point for edge tracing.
2. Follow edge: generate chain code, set edge pixels to white, and copy edge contour to output image.
3. Stop when we return to starting point.
4. Continue search for the next starting point for edge tracing.
5. Repeat steps 2 through 4 until the whole image has been scanned.
Raster scan used to search for the starting point for edge tracing:
To identify the starting point look for this pattern:
Explanation of shading:
This pixel must be black. (Begin tracing edge here.)
These four pixels must be white (background).
Exactly 2 of the 4 pixels shaded thus must be black.
a
3
2
1
4
0
5
6
7
b
Begin raster scan here
Pixel found: begin edge tracing
Chain code value indicates direction of travel when *leaving* an edge
0 7
1 7
0 0 1 7
1 6
2 6
3 6
3 5 4 5
3 4 3 4
c
First blob found
2nd blob found
Raster scan is resumed after tracing the edge of the first blob
Raster scan is resumed after tracing the edge of the 2nd blob
d

Fig. 17.9

Edge tracing. (a) Rules (b) Edge code directions (c) Locating the starting point and following the edge of an isolated blob. (d) Tracing the edges of multiple blobs

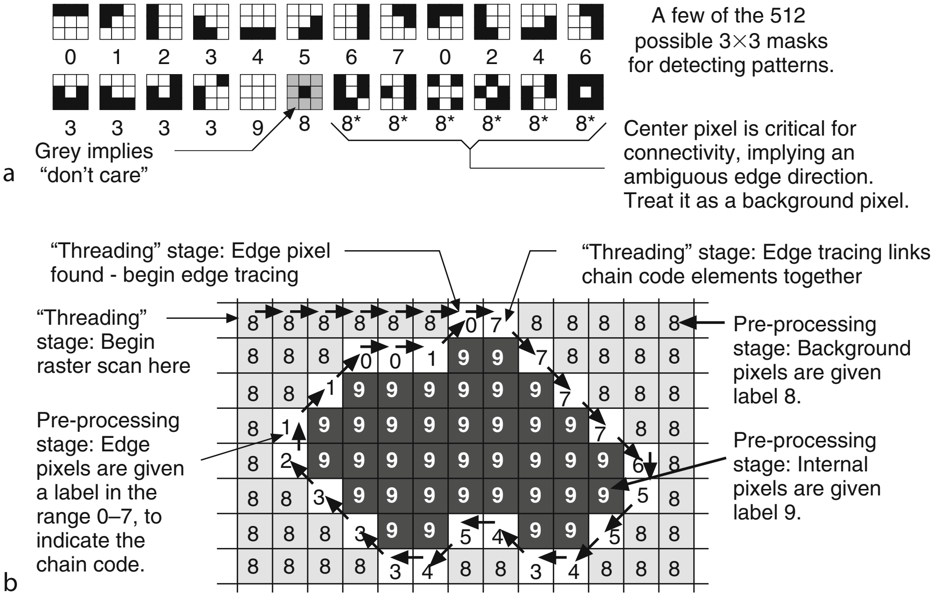

0 1 2 3 4 5 6 7 0 2 4 6
A few of the 512 possible 3×3 masks for detecting patterns.
3 3 3 3 9 8 8* 8* 8* 8* 8* 8*
Grey implies "don't care"
Center pixel is critical for connectivity, implying an ambiguous edge direction. Treat it as a background pixel.
a
"Threading" stage: Edge pixel found - begin edge tracing
"Threading" stage: Edge tracing links chain code elements together
"Threading" stage: Begin raster scan here
Pre-processing stage: Background pixels are given label 8.
Pre-processing stage: Edge pixels are given a label in the range 0–7, to indicate the chain code.
Pre-processing stage: Internal pixels are given label 9.
8 8 8 8 8 8 0 7 8 8 8 8 8
8 8 8 0 0 1 9 9 7 8 8 8 8
8 8 1 9 9 9 9 9 9 7 8 8 8
8 1 9 9 9 9 9 9 9 9 7 8 8
8 2 9 9 9 9 9 9 9 9 9 6 8
8 8 3 9 9 9 9 9 9 9 9 5 8
8 8 8 3 9 9 5 4 9 9 5 8 8
8 8 8 8 3 4 8 8 3 4 8 8 8
b

Fig. 17.10

Chain code generation (a) Preprocessing for the initial labelling of each pixel. Twenty-four out of the 512 possible 3*3 masks are shown here. (b) Threading the edge pixels together to generate the chain code

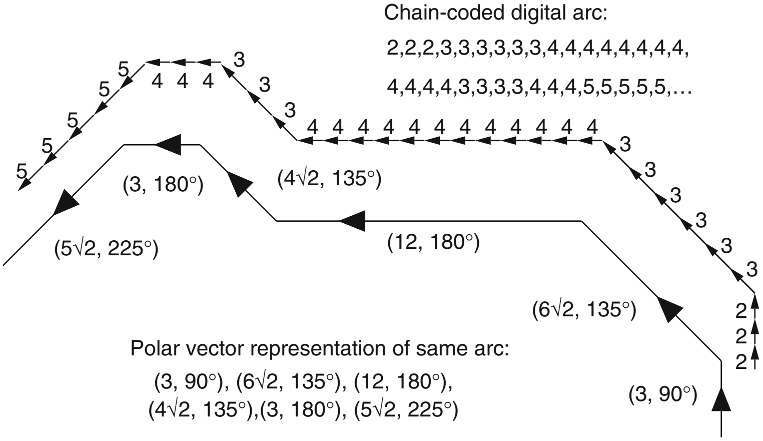


Fig. 17.11

(a) Chain-coded edge (b) Equivalent polar vector representation (PVR)

Thus, we have reduced a chain code sequence containing 37 elements to a polar vector representation continuing just six elements. It may, of course be possible to merge consecutive PVR elements to form longer vectors at angles that are non-integer multiples of 45°. This will make the PVR even more compact. However, this is venturing into edge smoothing, which will be discussed later.

17.2.1.4 Cartesian to Polar Coordinates

Many artifacts that are made in industry are circular, have circular markings, or possess a "wheel spoke" pattern. In order to simplify the processing of such images, it is possible to use a Cartesian-axis to polar-coordinate-axis transformation. If the axis transformation is centred properly, concentric circles are mapped into parallel lines. A series of radial lines is also mapped into a set of parallel lines that is orthogonal to the first set. Performing the axis transformation is not as straightforward as one might, at first, imagine. The reason is that pixels in the two coordinate systems do not have a one-to-one correspondence to one another (➋ *Fig. 17.12*). If we consider the pixels to be points, we can see how we can use interpolation in order to avoid sampling errors. Suppose that a point (i′, j′), defined using in the polar-coordinate axes, maps to the point (i″, j″) located with respect to the Cartesian-coordinate axes. Normally, (i′, j′) does not lie exactly on one of the grid points corresponding to the pixels in the Cartesian-axis image. To estimate the intensity at (i″, j″), we can use simple bilinear interpolation, just as we explained in connection with image rotation (➋ Sect. 2.6).

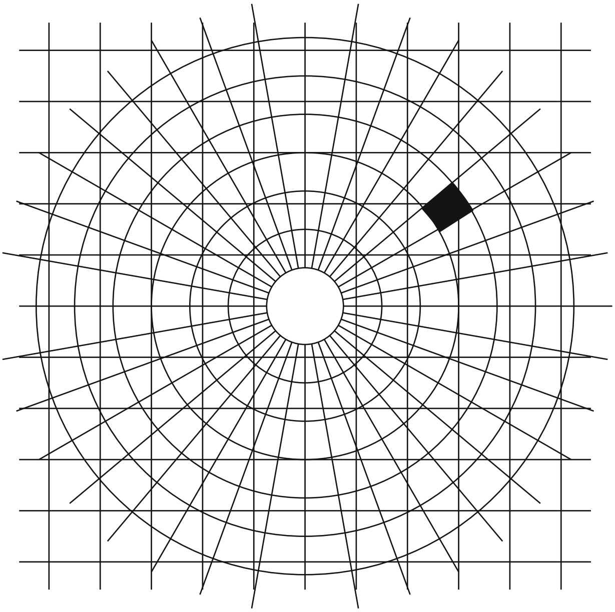


Fig. 17.12

Conversion from Cartesian to polar coordinates. Notice that there is no 1:1 correspondence between pixels in these two image formats

17.2.1.5 Colour Axis Conversion

Most colour cameras used in machine vision generate the RGB colour signals mentioned in ➲ Chap. 10. However, the HSI representation is usually regarded as being more useful for colour recognition. The transformation is defined in terms of the following equations:

$$Hue : H = cos^{-1}\left((2R - G - B)/2\left(((R - G)^2 + (R - B)(G - B)\right)\right)$$
$$Saturation : S = 1 - (3min(R, G, B))/(R + G + B)$$
$$Intensity : I = (R + G + B)/3$$

Given measured values of the signals R, G and B, these equations can be evaluated using software. However, this process must be repeated for every pixel in the input image and hence is likely to be very slow. At first sight it seems unlikely that look-up tables could be used effectively, since there are three variables. However, in most applications, we can safely restrict each of the RGB channels to 6 bits without serious loss of colour information. Even if we have to use 8 bits/channel, the situation is not too severe. The point is that, by using limited-precision colour representation, we make look-up tables quite effective. Since there are three colour channels, each with 6 (or 8) bits/channel, the total number of possible input conditions is 262,144=2^{18} (or 16,777,216=2^{24}). There are three signals (H, S and I) to be computed, each of which requires 6 (or 8 bits). Hence, we require a total of 0.75 MB (or 48 MB) of storage for the colour look-up tables. The cost of ROM/RAM is quite modest, so colour look-up tables provide a convenient option for colour axis conversion. Notice that look-up tables based on RAM/ROM can perform the colour transformation in real time on a digitised standard video signal.

17.2.2 Processing a Binary Image as a Grey-Scale Image

Two separate series of operators appropriate for processing grey and binary images were defined in ➲ Chap. 14. At first sight, it may seem unlikely that grey-scale processing algorithms could be applied to good effect on binary images but, in fact, there is a lot of benefit to be obtained by this simple ploy. For example, we can smooth the edge of a blob in a binary image by applying a low-pass blurring filter, such as the QT operator *lpf*, a few times (M) and then thresholding [QT: *thr*] at the mid-grey level. (We assume that the binary image intensity values are stored as 0 representing black and 255 denoting white.) This process will be represented by [*M•lpf, thr*]. (This notation is not part of QT but is allowed in PQT, its extended version in which QT is integrated with Prolog. ➲ Chap. 23.) It removes thin hair-like protuberances and “cracks” on the edge of a blob. The same grey-scale operator sequence can also eliminate so-called salt-and-pepper noise consisting of small well-scattered black and white spots. The effect achieved is very similar to that of the binary operator sequence [*N•lnb, N•snb, N•snb, N•lnb*], where N is instantiated to some suitable (small) integer value.

The grass-fire transform [QT: *gft*] is one obvious way in which a grey-scale image can be used to good effect to make it easier to analyse a binary image (➲ Fig. 14.10. This operator will be discussed in more detail later in this chapter.) The result of the QT operator *gft* is a grey-scale image in which intensity indicates the distance to the closest edge point. In addition, the operator *cnw* (count white neighbours in each 3 × 3 neighbourhood) generates grey-scale output image from a binary image while still preserving the shapes of blobs. (The binary image can be reconstructed by taking black as 0 and all values in the range 1–255 as white.)

The general linear convolution operator [QT: *glc*] can be used to detect specified patterns in a binary image. For example, the diagonal edge

W	*W*	*W*
W	*X*	*B*
B	*B*	*B*

X = don't care

can be detected by [*glc*(*1,1,1,1,0,−1,−1,−1,−1*), *thr(255)*].

This process can be taken further. The binary neighbourhood coding scheme introduced earlier can be implemented with the general convolution operator using the following weights:

1	2	4
8	0	16
32	64	128

(The QT command is [*glc(1,2,4,8,0,16,32,64,128)*], which generates the value 15 [= 255 × (1+2+4+8)/{1+2+4+8+16+32+64+128}] on the diagonal edge pattern above.)

The QT operator *rin* counts the number of white pixels in each row when it is applied to a binary image, while *rox* may be used to find the "shadow" of a blob.

The QT grey-scale operator *mxi* is very useful for superimposing white lettering (in a binary image) onto a grey-scale picture. On the other hand, a common ploy when processing grey-scale images containing several objects is to process the image first to produce a blob defining the edge of each object. We then select one blob and combine it with the original grey-scale image using the QT operator *mni.*

Several binary operators can be implemented directly using their grey-scale counterparts. Here are some examples:

Binary	**Grey scale**
`exw (expand white regions)`	`lnb`
`skw (shrink white regions)`	`snb`
`ior (OR)`	`mxi`
`xor (exclusive OR)`	`sub, thr(127,127), neg`
`bed (binary edge)`	`sed (approx.)`
`not (inverse)`	`neg`
`cnw (count white 8-neigbours)`	`glc(1,1,1,1,1,1,1,1,1)`
`cln (remove isolated white pixels)`	`glc(−1,−1,−1,−,1,16,−1,−1,−1,−1), thr(128, 254)`

17.2.3 Using Images as Look-Up Tables

Images can be used as look-up tables. For example, a general image warping transformation can be achieved with two images used as look-up tables. In the following equation, {u(i,j)} and {v(i,j)} are two images which, between them, define the warping function

$$c(i,j) = a(u(i,j), v(i,j)), 1 \leq u(i,j) \leq m, n \leq W.$$

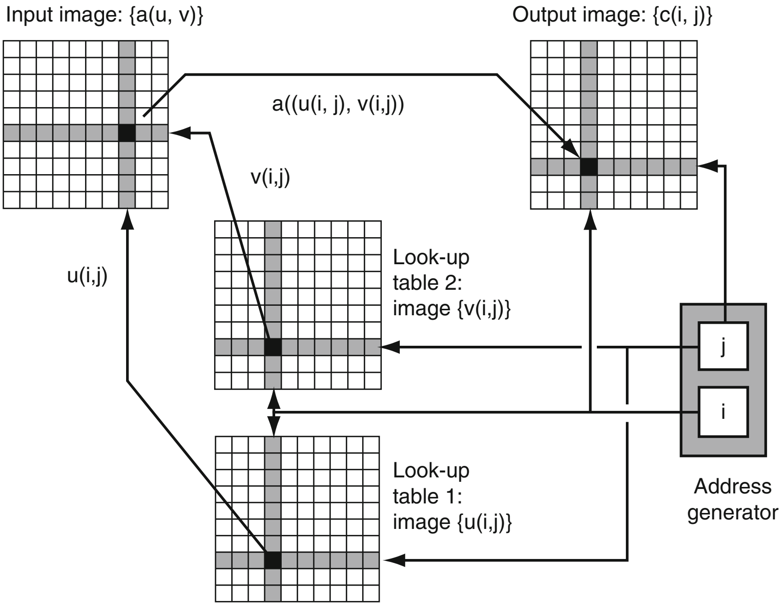


Fig. 17.13
Using images as look-up tables for image warping

The output image is {c(i,j)}; the input image is {a(u,v)}. The intensity of pixel (i,j) in the output image (i.e. c(i,j)) is derived from that pixel in the input image whose x-coordinate is u(i,j) and whose y-coordinate is v(i,j) (*Fig. 17.13*). Notice that the number of grey levels must not be smaller than the number of rows and columns in the images. If there were no warping at all, u(i,j)≡i and v(i,j)≡j. This technique will quite adequately perform geometric transformations where there is a small amount of warping; for example due to slight pin-cushion or barrel distortion, and is suitable for correcting for lens distortion, or the distortion caused by tube-camera scanning errors. It is also suitable for correcting trapezium (trapezoidal) distortion, which occurs when a camera looks obliquely onto a surface. However, it does have problems if the warping is very severe. For example, an approximate Cartesian-to-polar coordinate mapping can be achieved in this way but there are severe difficulties near the centre of the output image. *Figure 17.12* shows why this is so. Since there is no interpolation, each output pixel is derived from just one input pixel. As a result, the mapping is inaccurate close to the centre of the image. However, an annulus whose inner radius is not less than about half of its outer radius can be mapped quite accurately to a rectangle by this method.

17.3 Redefining Algorithms

In this section, we consider how an algorithm may be reformulated while its performance remains unaltered. In later parts of this chapter, we will discuss approximate and heuristic methods.

17.3.1 Convolution Operations

There are two ways to simplify the calculations required by the linear convolution operator [*glc*]:

1. Decomposition into two 1-dimensional operators, acting separately along the vertical and horizontal axes of the image.
2. Iterative application of small-kernel filters, to produce the same effect as a large-kernel filter.

We will now consider these separately in turn and then combine them later.

17.3.1.1 Decomposition

Consider the 7×1 linear convolution operator with the following weight matrix:

1	1	1	1	1	1	1

This filter blurs along the horizontal axis only; vertical intensity gradients are smoothed, while horizontal ones are not. On the other hand, this 1×7 filter

1
1
1
1
1
1
1

blurs horizontal intensity gradients but not vertical ones. Applying them in turn produces exactly the same effect as the following 7×7 filter:

1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1

Of course, this filter blurs in all directions. If the normalisation constants given in ➔ Sect. 14.2.4 are applied separately for the two 1-dimensional filters, the final result is also normalised correctly. Notice also that the simpler 1-dimensional filters can be applied in either order, without altering the final result.

Let us consider another simple example, where the weights are not all equal to unity and some are negative. Recall that the formula for the Sobel edge detector (➔ Sect. 14.2.5) consists of two parts, each combining blurring in one direction and intensity differencing in the orthogonal direction. Here is the formula for the part which performs horizontal blurring and vertical differencing:

$$E \leftarrow (A + 2\,B + C) - (G + 2\,H + I)$$

This operation can be performed as a horizontal blurring process using the 1 × 3 weight matrix

1	2	1

followed by calculating the vertical difference using this 3 × 1 weight matrix:

1
0
–1

These filters can be applied in either order.

The other component of the Sobel edge detector is $E \leftarrow (A+2D+G)-(C+2F+I)$, which can be computed by applying the 1 × 3 differencing filter with weight matrix

1	0	–1

followed by vertical blurring with this weight matrix:

1
2
1

Again, these operations can be applied in either order. Hence, the Sobel edge detector can be performed using four 1-dimensional filters, which may be easier than implementing it directly with 2-dimensional filters.

Decomposition is important because it replaces a "difficult" 2-dimensional filtering operation by two simpler ones. Filtering along the rows or columns of a raster-scanned image is very straightforward using standard filtering techniques. It is possible to hold the interim results from the first 1-D (row scan) filter in a RAM-based store. This is then rescanned in the orthogonal direction (column scan), thereby generating another digital video signal which can be filtered (➲ *Fig. 17.14*).

It is important to note that not all 2-D local operators can be decomposed into two orthogonal 1-D filters. To see that this is so, we only have to compare the number of degrees of freedom of a 2-D filter (49 for a 7 × 7 filter), with the equivalent figure for the two decomposed filters ($14 = 1 \times 7 + 7 \times 1$). However, most of the more important filters can be decomposed in this way, particularly if they possess both vertical and horizontal symmetry. (The number of degrees of freedom falls to 9 for 7 × 7 bi-symmetrical filters.)

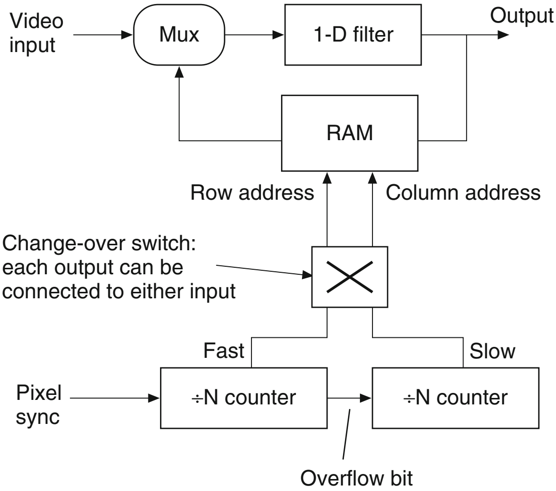


Fig. 17.14
Schematic circuit for implementing decomposable filters. In the first phase, the input image is scanned left-to-right/top-to bottom. (The signal on the line labelled "output" is ignored at this stage.) The image resulting from the row-filtering operation is stored in the RAM, filling it in the same manner. Then, in the second phase, the image is read out of the RAM, scanning the image in the order top-to bottom/left-to-right. The output signal is now taken directly from the line leading out of the 1-D filter

17.3.1.2 Iterated Filters

Consider the following filter

1	1	1
1	1	1
1	1	1

applied repeatedly to a very simple image consisting of zero (black) everywhere, except for a single central white pixel. For simplicity, we will ignore normalisation and assume that the white pixel has an intensity equal to 1. The results of applying the filter once (left) and twice (right) are shown below:

...	...	...	...	...	...	...	...	...	...	...
...	0	0	0	0	0	0	0	0	0	...
...	0	0	0	0	0	0	0	0	0	...
...	0	0	0	0	0	0	0	0	0	...
...	0	0	0	1	1	1	0	0	0	...
...	0	0	0	1	1	1	0	0	0	...
...	0	0	0	1	1	1	0	0	0	...
...	0	0	0	0	0	0	0	0	0	...
...	0	0	0	0	0	0	0	0	0	...
...	0	0	0	0	0	0	0	0	0	...
...	...	...	...	...	...	...	...	...	...	...

...	...	...	...	...	...	...	...	...	...	...
...	0	0	0	0	0	0	0	0	0	...
...	0	0	0	0	0	0	0	0	0	...
...	0	0	1	2	3	2	1	0	0	...
...	0	0	2	4	6	4	2	0	0	...
...	0	0	3	6	9	6	3	0	0	...
...	0	0	2	4	6	4	2	0	0	...
...	0	0	1	2	3	1	1	0	0	...
...	0	0	0	0	0	0	0	0	0	...
...	0	0	0	0	0	0	0	0	0	...
...	...	...	...	...	...	...	...	...	...	...

Two more applications yield the following images:

...	...	...	...	...	...	...	...	...	...	...
...	0	0	0	0	0	0	0	0	0	...
...	0	1	3	6	7	6	3	1	0	...
...	0	3	9	18	21	18	9	3	0	...
...	0	6	18	36	42	36	18	6	0	...
...	0	7	21	42	49	42	21	7	0	...
...	0	6	18	36	42	36	18	6	0	...
...	0	3	9	18	21	18	9	3	0	...
...	0	1	3	6	7	6	3	1	0	...
...	0	0	0	0	0	0	0	0	0	...
...	...	...	...	...	...	...	...	...	...	...

...	...	...	...	...	...	...	...	...	...	...
...	1	4	10	16	19	16	10	4	1	...
...	4	16	40	64	76	64	40	16	4	...
...	10	40	100	160	190	160	100	40	10	...
...	16	64	160	256	304	256	160	64	16	...
...	19	76	190	304	361	304	190	76	19	...
...	16	64	160	256	304	256	160	64	16	...
...	10	40	100	160	190	160	100	40	10	...
...	4	16	40	64	76	64	40	16	4	...
...	1	4	10	16	19	16	10	4	1	...
...	...	...	...	...	...	...	...	...	...	...

Inspection of these results shows that we have progressively blurred the spot, which began as a 1-pixel entity and was transformed into an indistinct "fuzz" occupying 9 × 9 pixels. The same result could be obtained applying the following 9 × 9 filter once:

1	4	10	16	19	16	10	4	1
4	16	40	64	76	64	40	16	4
10	40	100	160	190	160	100	40	10
16	64	160	256	304	256	160	64	16
19	76	190	304	361	304	190	76	19
16	64	160	256	304	256	160	64	16
10	40	100	160	190	160	100	40	10
4	16	40	64	76	64	40	16	4
1	4	10	16	19	16	10	4	1

It is, of course, no accident that the weights in this matrix are exactly the same as the values in the fourth image matrix above.

By combining the decomposition and iterated filter techniques, we can simplify this even further. We can obtain the same result as the 9 × 9 filter by applying the 1 × 3 1-D *row* filter with weight matrix [1, 1, 1] four times, and then applying the 3 × 1 *column* filter with weights [1, 1, 1] four times.

17.3.2 More on Decomposition and Iterated Operations

The decomposition technique just described can be applied to a number of other operators. For example, the QT operator *lnb* (largest 8-neighbour intensity value) can also be separated into row and column operations, thus:

$$E \Leftarrow \mathrm{MAX}(D, E, F)$$

followed by

$$E \Leftarrow \mathrm{MAX}(B, E, H).$$

The operator *lnb* computes its output using all of the 8-neighbours. However, if we had defined *lnb* in a different way, using only the 4-neighbours, decomposition in this sense would not be possible.

So far, we have used the results of two filters in series, so that the result of one image processing (row) operator is applied to another (column) operator. However, we could apply two 1-D filters in parallel and then combine the results. Suppose we perform the operations E ⇐ MAX(D, E, F) and E ⇐ MAX(B, E, H) in *parallel* and then combine the results by calculating the point-by-point maximum of intensities in the processed and original images [QT: *mxi*]. The result is exactly the same as the following operator, which combines only the 4-neighbours:

$$\mathrm{E} \Leftarrow \mathrm{MAX}(\mathrm{B}, \mathrm{D}, \mathrm{E}, \mathrm{F}, \mathrm{H})$$

We will refer to this process later, so we will use the mnemonic *lnb4*, to emphasis that it uses only the 4-neighbours. The equivalent decomposition process can be defined for the related operator *snb* (smallest 8-neighbour intensity value).

By repeating the operator *lnb* or *snb*, we can extend the size of the processing window (➲ *Fig. 17.15*). For example, the QT sequence *N•lnb* finds the maximum intensity within each window of size (2 · N+1) pixels. By repeating *lnb4* several times, we find the maximum intensity within a larger diamond-shaped region.

By alternating *lnb* and *lnb4*, we can find the maximum intensity within a regular octagonal region surrounding each pixel. Since this octagon is a better approximation to a circle than either *lnb* or *lb4* alone produces, we may find this combination attractive in certain applications. In the figure, [*5•lnb*] finds the largest intensity within an 11 × 11 window, while [*5•exw*] expands a single white pixel to cover an 11 × 11 square. Alternating *lnb4* and *lnb8* (or *exw4* and *exw8*) generates an octagonal covering. The erosion operators *snb* and *skw* operate similarly on black pixels.

Dilation [QT: *dil*] and *erosion* [QT: *ero*] are two important operators for processing binary images. Since they are so closely related to *lnb* and *snb* respectively, it will hardly come as any surprise that they can be decomposed in a similar way. Indeed all of the remarks made about *lnb* also relate directly to *dil* and those about *snb* to *ero* (binary erosion). This includes the comments made about the 4-neighbour versions.

17.3.3 Separated Kernel Image Processing Using Finite-State Machines

In a series of 23 papers beginning in 1994, Waltz (WAL94a... WAL99b, HUJ95a, HUJ95b, HAC97a, MIL97) described an extremely powerful and radically new paradigm for the implementation of many important neighbourhood image processing operations. These techniques, taken together, have been given the acronym SKIPSM (*Separated Kernel Image Processing using finite-State Machines.* See ➲ Chap. 20). The improvements in performance (speed or neighbourhood size) provided by SKIPSM are so great in some cases as to demand a radical rethinking of what is or is not practical in industrial applications. The SKIPSM approach is mentioned briefly in this section on operator decomposition because it allows the decomposition of a broad range 2-D (and even 3-D or higher) linear and nonlinear operators into a sequence of 1-D operations. This includes many operators which

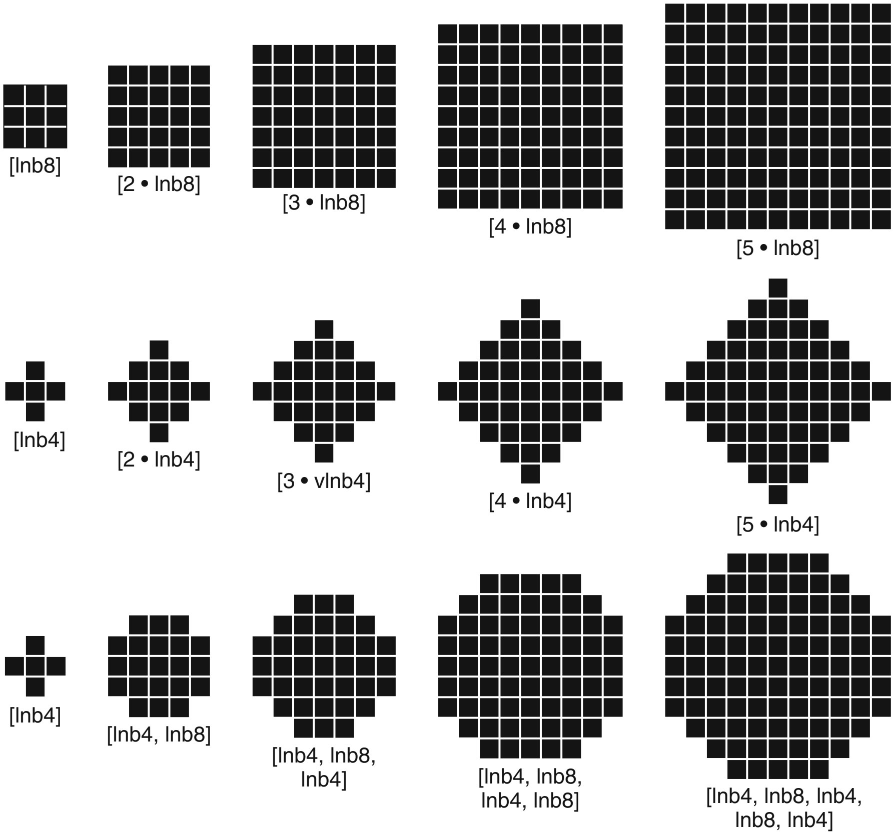


Fig. 17.15
Applying the dilation operators *lnb* and *exw* iteratively (*lnb* operates on grey-scale pictures, while *exw* operates on binary images.)

were previously thought to be non-separable. An example of intermediate difficulty will be presented here.

17.3.3.1 SKIPSM Example

We wish to apply *six* 7 × 9 binary erosion structuring elements (SEs) *on a single raster-scan pass* through the input image. The six SEs are shown in *Fig. 17.16*.

Using the automated SKIPSM code-generation programs, these six operations are individually separated into row operations and column operations. The row operations are combined into a single recursive FSM (finite-state machine), the *row machine*. The six column operations are combined into a second FSM, the *column machine*. These two FSMs are then typically (but not necessarily) implemented as look-up tables (LUTs), as shown in *Fig. 17.17*. Each of the six separate binary outputs is assigned to a single bit plane of the 8-bit output image.

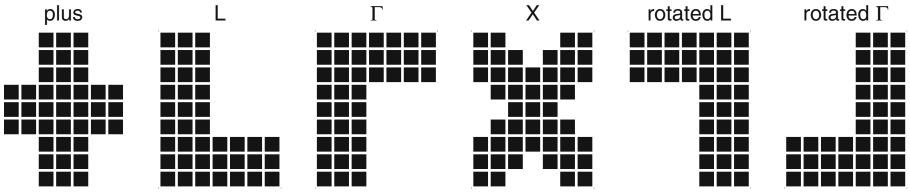


Fig. 17.16
The six 7 × 9 binary erosion structuring elements applied simultaneously in Example 1. Among these, there are only seven distinct row patterns to be encoded by the row machine: 0011100, 1111111, 1110000, 1100011, 1110111, 0111110 and 0000111

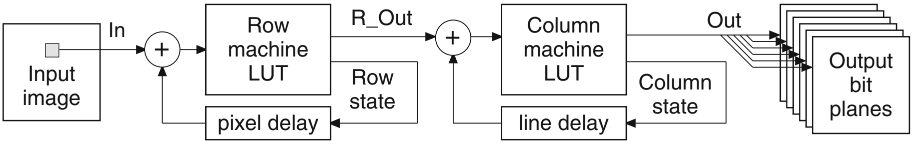


Fig. 17.17
SKIPSM flow chart (applicable to both hardware and software implementations) showing the row and column machines being implemented as look-up tables

The pixel values from the input image are fed to this system in the usual raster-scan order. Each pixel is fetched *once and only once*, because all of the necessary "history" of the image neighbourhood is contained in the *states* of the two FSMs.

The combined row machine has 216 states and, because the input has two levels, its LUT therefore has only 432 addresses. The combined column machine has 1,748 states and 16 input levels. Thus, its lookup table has 10,488 addresses. This is much larger than the row-machine LUT, but is still not excessively large, by today's standards. These LUTs are generated automatically by the SKIPSM software.

Please note a *very* important point here: The execution time for SKIPSM lookup-table-based implementations is *very* fast (requiring only four LUT accesses and two additions or bitwise-ORs), *independent of the size of the neighbourhood and independent of the number of simultaneous operations performed.* Large operations and/or multiple operations require larger LUTs, but execute in the same very fast time as small ones. This fact has been demonstrated repeatedly on actual systems. The SKIPSM approach is described in more detail in ➋ Chap. 20.

17.3.4 Binary Image Coding

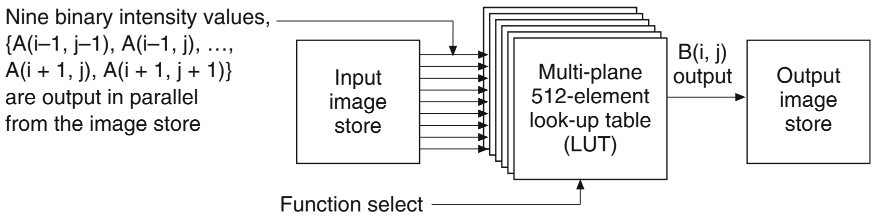
In ➋ Sect. 17.2.1 (➋ *Figs. 17.8–17.10*), we introduced the idea of the binary image coding scheme and in ➋ Sect. 17.2.2 described how the linear convolution operator [*glc*] can be used to generate it. A wide range of binary image processing algorithms can make good use of this coding technique, which is incorporated into the implementation scheme outlined in ➋ *Fig. 17.18*. The important point to note is that any logic function of the nine pixel intensities found within a 3 × 3 window can be "computed" by a look-up table. Here is a list of some of the binary image processing functions which can be implemented in this way.

Operation	QT mnemonic	Comments
Dilate	dil	
Erode	ero	
Binary edge detection	bed	
Remove isolated white points	wpr	
Count white neighbours	cnw	
Critical points for connectivity	cny	
Skeleton joints	jnt	
Skeleton limb ends	lme	
Extend ends of arcs (skeleton limbs)	eel	
Euler number	eul	Extra processing needed
Chain code - preprocessing	fcc	See ❯ Sect. 17.2.2
Grass-fire transform (1 stage)	gft	Included in iterative loop
Skeletonisation	ske	Included in iterative loop

Notice that the binary image coding and the implementation scheme of ❯ *Fig. 17.18* can be used to good effect in the grass-fire transform [QT: *gft*] and the skeletonisation operator [QT: *ske*]. In both cases, a complex logical function is applied within an iterative loop, so high-speed implementation is of increased importance. Implementations using this configuration will be considerably slower than those for the corresponding SKIPSM implementations, as described in ❯ Sect. 17.3.4, but the required look-up tables would be smaller. Therefore, these might be preferred where large look-up tables are not feasible.

17.3.5 Blob Connectivity Analysis

The task to which we will give our attention in this section is that of labelling blobs in a binary image such as shown in ❯ *Fig. 17.19*. All pixels in each 8- or 4-connected blob are assigned an intensity value to indicate the blob identity. Expressed in this way, the requirement seems very straightforward. However, when we consider general shapes about which nothing is known in advance, the real complexity of the algorithm becomes apparent. In particular, intertwined spirals or interleaved comb-like structures make the task far from straightforward.


Nine binary intensity values,
{A(i–1, j–1), A(i–1, j), …,
A(i + 1, j), A(i + 1, j + 1)}
are output in parallel
from the image store
Input image store
Multi-plane 512-element look-up table (LUT)
B(i, j) output
Output image store
Function select

■ **Fig. 17.18**
Using a look-up table to perform a binary image processing operation. The function implemented by this circuit can be altered very quickly by changing the "Function Select" input

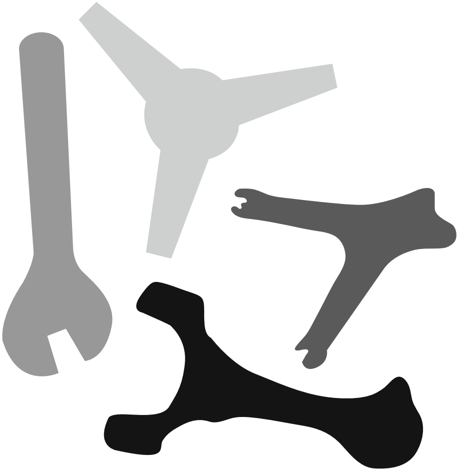


Fig. 17.19
All pixels within a given blob are given the same intensity value. Each blob is assigned a different intensity

Several important image analysis tasks are made possible by blob labelling, including these:

- Identifying the blob with the largest area.
- Eliminating from an image all blobs that are smaller than a predetermined size.
- Associating the holes (or lakes) within a blob with the outer edge of that blob.

17.3.5.1 First Blob-Labelling Scheme

We will describe three quite different algorithms for blob labelling. The first algorithm is explained in *Fig. 17.20* and is based on the array representation of a binary image. This algorithm requires two passes through the image. Thus, each pixel is visited twice. During the first pass, each pixel is assigned a provisional numeric value (label). A blob with a complicated shape may receive two or more different labels. On this first pass, each new "finger" that points upwards is assumed to be a new blob and hence is assigned a new label. During the second pass, these sub-regions are renumbered as required, so that all pixels within a given blob are given the same label.

17.3.5.2 Second Blob-Labelling Scheme

The second blob-labelling scheme represents a fairly minor extension of the algorithm for generating the chain code (Sect. 17.2.1 and *Fig. 17.9a*), and is shown in *Fig. 17.21*. The modified algorithm requires that only the blob edge pixels in the input image be white. All other pixels, including the background and pixels "inside" the blobs, should all be black. This can be achieved by applying a simple "binary edge" processing sequence before the following edge tracing process is started:

1. Set a counter (denoted here as *Counter*) equal to 1.
2. Initialise the output image to black (level 0).
3. Initialise parameters *U* and V to 1.

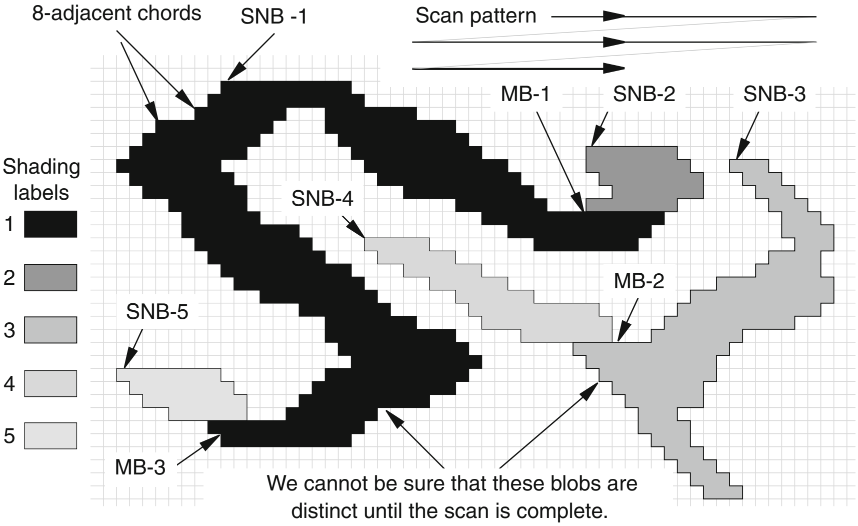


Fig. 17.20
First blob labelling algorithm. This takes place in two stages, each comprising a raster scan of the image. During the first stage, each pixel is assigned an initial label. SNB denotes "starting new blob". MB indicates where parts of blobs are merged. We suspect that a new blob has been found each time we encounter the binary pattern

W	W	W
W	B	X
X	X	X

where W=white (background), B=black (object), and X="Don't care." Each pixel identified thus is given a new initial label. (Pixels SNB-1 to SNB-5) Other white pixels that are discovered to be 8-connected to SNB-i are given the same initial label (i.e., value i). The pixels marked MB-1 to MB-3 indicate where two regions that were originally thought to be separate are discovered to be 8-connected and hence form parts of the same blob. For example, at MB-1, we find that initial labels 1 and 2 represent regions within the same blob and should therefore be merged. Later in the first raster scan, at MB-3, we discover that the same blob also includes region 5. During the second raster-scan, regions 2 and 5 and reassigned with label 1. Of course, regions 4 and 3 are also merged, by mapping all pixels

4. Start a raster scan of the input image at point $[U,V]$. When the first white pixel is encountered, record its position coordinates, $[X_{Counter}, Y_{Counter}]$, then proceed to the next step.
5. Trace the edge contour just found. At each edge pixel $[X_{edge}, Y_{edge}]$ perform the following operations:
   (a) Set pixel $[X_{edge}, Y_{edge}]$ in the input image to level 0 (black).
   (b) Set pixel $[X_{edge}, Y_{edge}]$ in the output image to level *Counter*.
   (c) Stop edge tracing when we revisit pixel $[X_{Counter}, Y_{Counter}]$ in the input image and proceed to the next step.
6. Set $[U,V]=[X_{Counter}, Y_{Counter}]$.

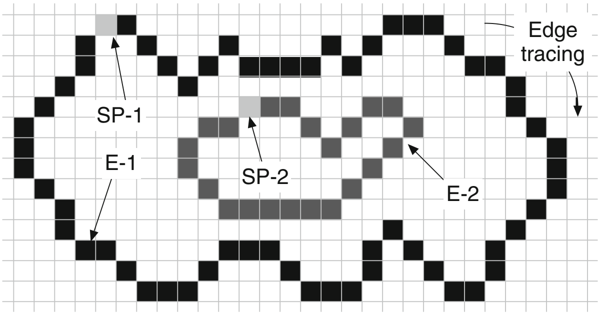


Fig. 17.21
Second blob labelling algorithm. Here there are two blobs, with edges, E-1 and E-2. Following the procedure (step 4), the first starting point (SP-1) is found, and edge E-1 is traced back around to SP-1 (step 5). Then the second starting point (SP-2) is found. And so on

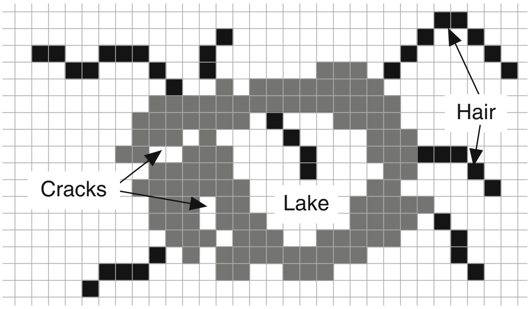


Fig. 17.22
"Hair", "cracks" and a "lake" in a binary object

7. Increment *Counter*.
8. Repeat steps (4) to (7), using the new values for *Counter, U* and *V*.

An important feature of this algorithm is that it cannot tolerate "hairy" edges, since they cause the edge tracer to travel along "blind alleys." ("Hair" is an open-ended digital arc that is only one pixel wide and protrudes outwards from the outer edge contour into the background or inwards from the inner edges of lakes (see Fig. 17.22)). "Hair" can be removed and a closed edge contour created by simple binary image filtering:

skw, exw, exw, xor % Creates an edge just outside object boundary

or

skw, exw, skw, xor % Creates an edge just inside object boundary

One small task remains: differentiating between outer and inner edges. We can use the following masks to identify outer or inner edges (edges surrounding lakes):

Mask for outer edges

B	B	X
B	W	X
X	X	X

Mask for inner edges

W	W	X
W	W	X
X	X	X

(X indicates don't care.) These masks are applied at step 4 to test the pixel located at [$X_{Counter}$, $Y_{Counter}$] in the preprocessed image. The masks are not applied to the edge contour image, just this single point.

17.3.5.3 Third Blob-Labelling Scheme

The third method for blob labelling uses a technique borrowed from computer graphics, where it is used for object filling. We must first define our notation.

1. Let [X_1, Y_1] be a point that is known to lie on the edge of or within the blob that we wish to isolate.
2. Let *L* denote a list of points to be analysed.
3. Let *S* be a list. This will eventually hold the addresses of all pixels that are 8-connected to [X_1, Y_1].
4. Let pixel intensities be represented by 0 (black), 1 (white and not yet analysed), and 2 (white and already analysed).
5. Let N(*X*,*Y*) denote the list of addresses of pixels that are 8-connected to [*X*,*Y*] and have intensity equal to 1. (Pixels of intensity 0 and 2 are ignored when computing N(*X*,*Y*)).

Here is the algorithm, described as a set of simple rules:

1. (a) Initialise *L* to [[X_1, Y_1]].
   (b) Set *S* to [], i.e. the empty list.
   (c) Set pixel [X_1, Y_1] to intensity value 2.
2. (a) If *L* is empty, go to Step 4.
   (b) Otherwise, find the first element of *L* and denote this by [*X*,*Y*].
3. (a) Derive the list N(*X*,*Y*).
   (b) Set the intensity value of pixel [*X*,*Y*] to 2.
   (c) Append N(*X*,*Y*) to *L*.
   (d) Append [*X*,*Y*] to the list *S*.
4. Finish.

The result is a list (*S*) which contains the addresses of all of the pixels that are eight-connected to the starting point [X_1, Y_1]. However, the result (*S*) in a rather inconvenient format, since the image scanning process is performed as we generate N(*X*,*Y*).

In this section, we have seen that three completely different computational techniques can be used to perform blob labelling. We have also seen that sometimes it can be beneficial to store and employ more than one image representation.

17.3.6 Convex Hull

The convex hull of a single connected object (*S*) in a binary image is easily imagined as the region contained within a rubber band stretched around *S*. The convex hull of a set of disconnected objects can also be envisioned in this way. There are several known algorithms for computing the convex hull of a single blob and rather fewer for the more general multiple-blob case. Algorithms for calculating the convex hull of a single blob can be based on the array, run-code, chain-code or polar-vector representations. For the sake of simplicity, we will concentrate in this section on algorithms for computing the convex hull of a single connected blob. This situation is also of greatest interest for industrial applications. However, we must make it clear that, strictly speaking, the convex hull is defined for *any* set of points in a plane, whether or not they are connected to each other.

A very useful point to note is that for any given shape (*S*) there are four very easily identifiable points on its convex hull [WIL89] (see *Fig. 17.23*):

(a) A vertical line through the left-most point of *S*
(b) A vertical line through the right-most point of *S*
(c) A horizontal line through the top point of *S*, and
(d) A horizontal line through the bottom point of *S*

If we were to rotate *S*, we could obtain another four points on its convex hull with equal ease. Of course, to find all of the points on the convex hull requires that we rotate *S* many times and find these four extreme points each time.

This provides the basis for most of the known algorithms for computing the convex hull. Let us now discuss the details of some of these.

17.3.6.1 Array Representation

Consider the set of parallel lines defined by the equation

$$y = x \tan(\theta) + c$$

where c is allowed to vary but θ is a constant. Some of these lines will intersect the given shape *S*, while others will not. The two lines that are tangential to *S* are of particular interest, since they

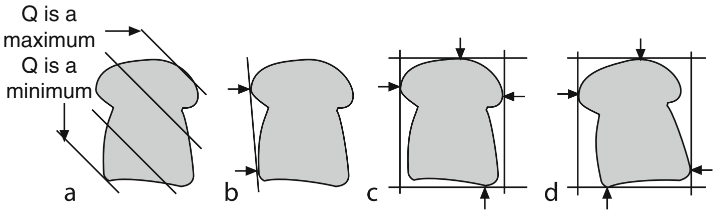


Fig. 17.23

Heuristics for finding a subset of the nodes of the convex hull of a binary object. (a) Finding the values for [*X,Y*] which maximise/minimise Q = (*Y* − mX − c) yields at least two nodes for each pair of values (m, c). (b) This heuristic may find more than two nodes. (c) The top-most, right-most, bottom-most and left-most points of an object indicate (at least) four nodes. (d) Four more nodes can be found by rotating the object and repeating (c)

define (at least) two points on the convex hull of *S* (➲ *Fig. 17.23*). We can find their coordinates in the following way:

1. Evaluate the expression $(Y - X\tan(\theta) - \mathrm{c})$ for all points $[X,Y]$ in the set of points which define *S*. (For convenience, we will also refer to this set as *S*.) Hence, we compute Q, where $\mathrm{Q} = (Y - \mathrm{m}X - \mathrm{c})$, for all $(X,Y) \in \mathrm{S}$.
2. We record those values of (X,Y) which make Q a minimum and a maximum. Let us denote these values by $(X_{min,\theta}, Y_{min,\theta})$ and $(X_{max,\theta}, Y_{max,\theta})$.
3. Then, $(X_{min,\theta}, Y_{min,\theta})$ and $(X_{max,\theta}, Y_{max,\theta})$ are two points on the convex hull of *S*.
4. Additional points on the convex hull of *S* can be found by varying θ and repeating steps 1 through 4.

Notice that for a given value of θ there may be more than one convex hull point on each line (➲ *Fig. 17.23b*). This is a bonus and gives us more convex hull points for less computational effort. However, we cannot be sure that we will obtain more than one point per tangent line.

The above procedure can be used as the basis for a number of algorithms for deriving the convex hull. These will now be described.

17.3.6.2 Scanning the Image Array

The most obvious way to compute the convex hull of a shape *S* is to repeat steps 1 through 4 a large number of times for small increments of θ. Each pair of convex hull points $(X_{min,\theta}, Y_{min,\theta})$ and $(X_{max,\theta}, Y_{max,\theta})$ is computed by scanning the whole image array to find values which are members of *S*.

A second and slightly more efficient algorithm can be obtained by performing the binary edge detector function [QT: *bed*] on the image array before we apply the procedure just described. The savings in computational effort arise because we calculate only the value of Q for edge points – "internal" points are ignored. Notice, however, that each time we add two more convex hull points to the list, each pixel in the *whole image array* must be "visited" once.

17.3.6.3 Chain Code

A third and even more efficient procedure for deriving the convex hull can be based on the chain code and is exactly as defined above, except that we "visit" only the edge pixels. (Background and "internal" points are simply ignored.) This algorithm requires that we compute $(X_{min,\theta}, Y_{min,\theta})$ and $(X_{max,\theta}, Y_{max,\theta})$ from the (X,Y)-coordinates of all of the edge points. This is a trivial calculation based on the following recursive procedure:

$$(X_{i+1}, Y_{i+1}) = ((X_i + H(C_i)), (Y_i + V(C_i)).$$

Here, C_i is the chain code value which corresponds to a movement from the i$^{\mathrm{th}}$ edge pixel (X_i, Y_i) to the adjacent (i+1)$^{\mathrm{st}}$ edge pixel, (X_{i+1}, Y_{i+1}). H(*C*) and V(*C*) are derived from the following table:

C	0	1	2	3	4	5	6	7
H	1	1	0	−1	−1	−1	0	1
V	0	1	1	1	0	−1	−1	−1

Of course, we can modify the chain code generator instead, so that it also records the (X, Y) coordinate values of each of the edge pixels as the chain code is being computed. This will save some further computational effort.

17.3.6.4 Polar Vector Representation (PVR)

A fourth, radically different, algorithm for computing the convex hull can be defined in terms of the polar vector representation (PVR). Another similar procedure based on the chain code also exists but is less efficient [2]. In the following description, we will make use of the fact that the PVR defines a polygonal representation of a shape. We will use N_v to indicate the number of vertices of this polygon, which we will represent by P. The algorithm finds a set of points, which we will call *nodes* and which are the vertices of that polygon which defines the convex hull of P. We will use N_n to indicate the number of nodes. The set of nodes is a subset of the set of vertices. Hence, $N_n \leq N_v$. Also, for many shapes, particularly after edge smoothing has been applied, we find that N_v is *very much less* than the number of edge pixels (N_{edge}).

The first step in the algorithm is to find any one node and a reference axis which passes through it. A simple way to do this is to find the right-most vertex and then draw a vertical line through it (see ➲ *Fig. 17.24a*). Any other line that is tangential to the polygon is acceptable as the initial reference axis. On subsequent steps in the algorithm, the reference axis is redefined by the line joining the two most recently discovered nodes.

Consider ➲ *Fig. 17.24b*, which shows the angle θ formed between the reference axis and the line which joins the most recently found node (labelled ‘2’) to another vertex. The next node to be identified (3) is therefore identified as that vertex where the angle θ is a minimum. To minimise θ, we must analyse at most $(N_v - 2)$ vertices. Vertices labelled ‘1’ and ‘2’ are already known to be nodes of the convex hull and therefore need not be considered again. However, we can improve the algorithm in two ways:

(a) We do not need to calculate the angle θ for those vertices that lie between the node that was identified initially and the one discovered most recently.

(b) We need analyse only vertices which lie in the same “quadrant” of the polygon P. See ➲ *Fig. 17.24c*. The justification for this follows directly from ➲ *Fig. 17.23*.

As a crude approximation, rule (a) halves the amount of computing effort needed, while rule (b) reduces it to 25%. These two savings are cumulative. Hence, the total computational effort in finding all nodes involves roughly $(N_n\ N_v/8)$ angle calculations. An adaptation of this algorithm but based on the chain code has also been devised [7].

17.3.6.5 Parallel Implementation of the PVR/Chain Code Algorithm

It is possible to break the calculation into an arbitrary number of smaller tasks, which can be performed in parallel by a set of processors operating independently of each other. The number of parallel sub-tasks and hence processors is unlimited. The parallel version of the PVR algorithm relies on three phases: finding a sub-set of the nodes of the convex hull; segmenting the edge, and calculating the intermediate nodes for each segment. Here is the algorithm explained in more detail.

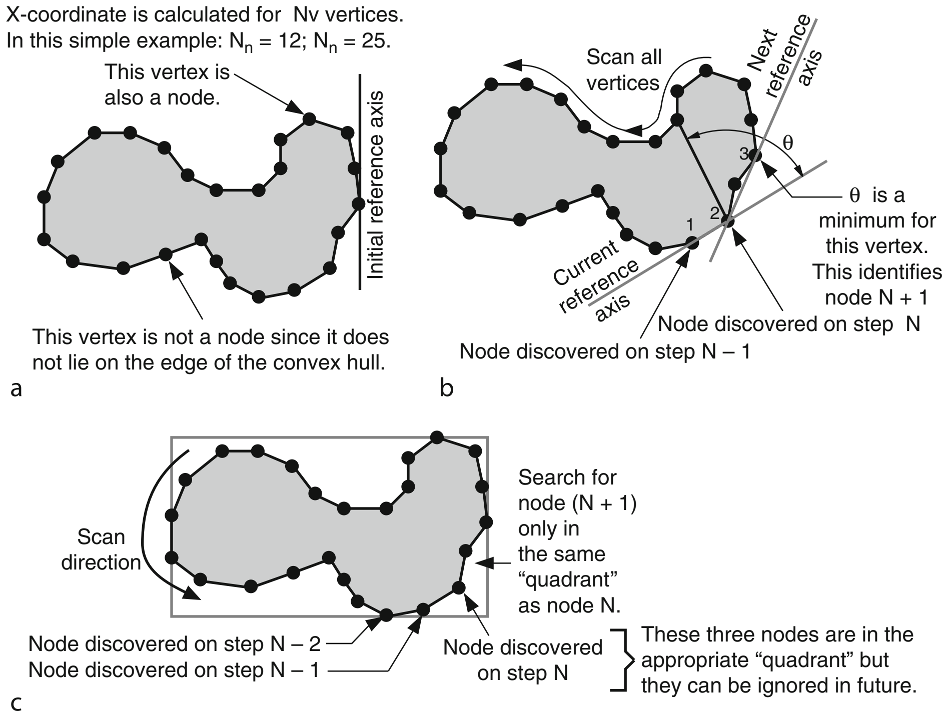

X-coordinate is calculated for Nv vertices.
In this simple example: N_n = 12; N_n = 25.
This vertex is also a node.
Initial reference axis
This vertex is not a node since it does not lie on the edge of the convex hull.
a
Scan all vertices
Next reference axis
θ
3
2
1
θ is a minimum for this vertex. This identifies node N + 1
Current reference axis
Node discovered on step N
Node discovered on step N – 1
b
Scan direction
Search for node (N + 1) only in the same "quadrant" as node N.
Node discovered on step N – 2
Node discovered on step N – 1
Node discovered on step N
These three nodes are in the appropriate "quadrant" but they can be ignored in future.
c

Fig. 17.24

Calculating the convex hull from the polar vector representation. (a) Initial reference line is the vertical line through the right-most vertex. (b) The (N+1)st node is derived from the (N – 1)st and Nth nodes. (c) Segmenting the PVR at the top-most, right-most, bottom-most and left-most vertices, creates four sub-tasks which can be performed in parallel

(a) Find a sub-set of nodes of the convex hull. To do this, we calculate the coordinates [X, Y] of each point which maximises the quantity

$$Q = (Y - X\tan(\theta) - c) \text{ for } \theta = 0, 360/\mathrm{N}, 720/\mathrm{N}, 1080/\mathrm{N}, \ldots, 360(\mathrm{N}-1)/\mathrm{N},$$

where N is the number of processors used. Of course, all points must lie on the edge of the input shape. This is an N-way parallel process; each process can be performed independently by a (serial) processor.

(b) Each processor passes the coordinates of the nodal point it has just identified to one of its neighbours. We are effectively sub-dividing the edge of the input blob into N segments. In *Figs. 17.23* and *17.24c*, we suggested that we break the edge of the input blob into four sections. This is simply a generalisation of that process. See *Fig. 17.25*.

(c) Find the remaining nodal points, as we did in the fourth PVR method. Each processor has the coordinates of two nodes and therefore finds the other nodes which lie between them.

This algorithm is important because it has deliberately been designed to fit onto an N-way parallel processor. This means, of course, that there is the potential to achieve high operating speed.

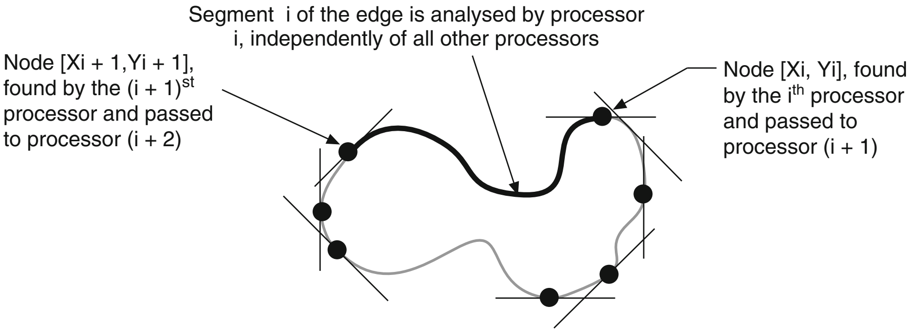

Segment i of the edge is analysed by processor i, independently of all other processors
Node [Xi + 1,Yi + 1], found by the (i + 1)st processor and passed to processor (i + 2)
Node [Xi, Yi], found by the ith processor and passed to processor (i + 1)

Fig. 17.25
The convex hull calculation lends itself to implementation in parallel processors

17.3.6.6 Run Code

Another ingenious algorithm for finding the convex hull uses the run code [10]. However, this will not be explained in detail here, since it is not possible to do so quickly and easily. This algorithm is suitable only for a serial processor and hence is most appropriate for implementation in software.

17.3.7 Edge Smoothing

As an abstract concept, edge smoothing needs very little explanation, even to the reader with little prior experience of machine vision. The algorithmic procedures which nominally perform edge smoothing are highly varied and are based on several different image representations. There are subtle differences among these competing techniques, which may be important in some applications but for most are insignificant. A procedure which embodies edge smoothing and relies upon these minor differences is likely to be over-sensitive for most industrial applications. In such situations, the system design engineer is more likely to find other more profitable ways of improving the machine. Adjusting the lighting/viewing subsystem is often far better!

It should be understood that the objective for edge smoothing is not specified in absolute terms. Hence, we take it that we may make a free choice of whichever edge smoothing technique happens to fit in best with our broad objectives. We might, for example, find that for a particular system the chain code provides a suitable data structure for implementing a multi-step algorithm. In this event, we would naturally use one of the edge smoothing operators that is based on the chain code. In certain other situations, however, the hardware might employ a different image representation. In this case, we find ourselves using an edge-smoothing procedure which behaves in a slightly different way. The point is that we simply should not put ourselves in a position where it matters that we select one edge smoothing algorithm rather than another. This is yet another illustration of a very important general principle: *Ease of implementation is as important as performance.*

17.3.7.1 Edge Smoothing in QT

The morphological operator *closing* performs one type of edge smoothing, in which thin "cracks" are eliminated (see *Fig. 17.26*). Let closing(N) be defined as N • exw, N • skw.

Remember that *exw* denotes expand white regions and *skw* signifies shrink white regions. Similarly, closing which is defined as follows N • skw, N • exw eliminates "hair" (*Fig. 17.22*). Both "hair" and "cracks" can be eliminated by concatenating these two operators as follows:

$$\text{closing}(N),\ \text{opening}(N).$$

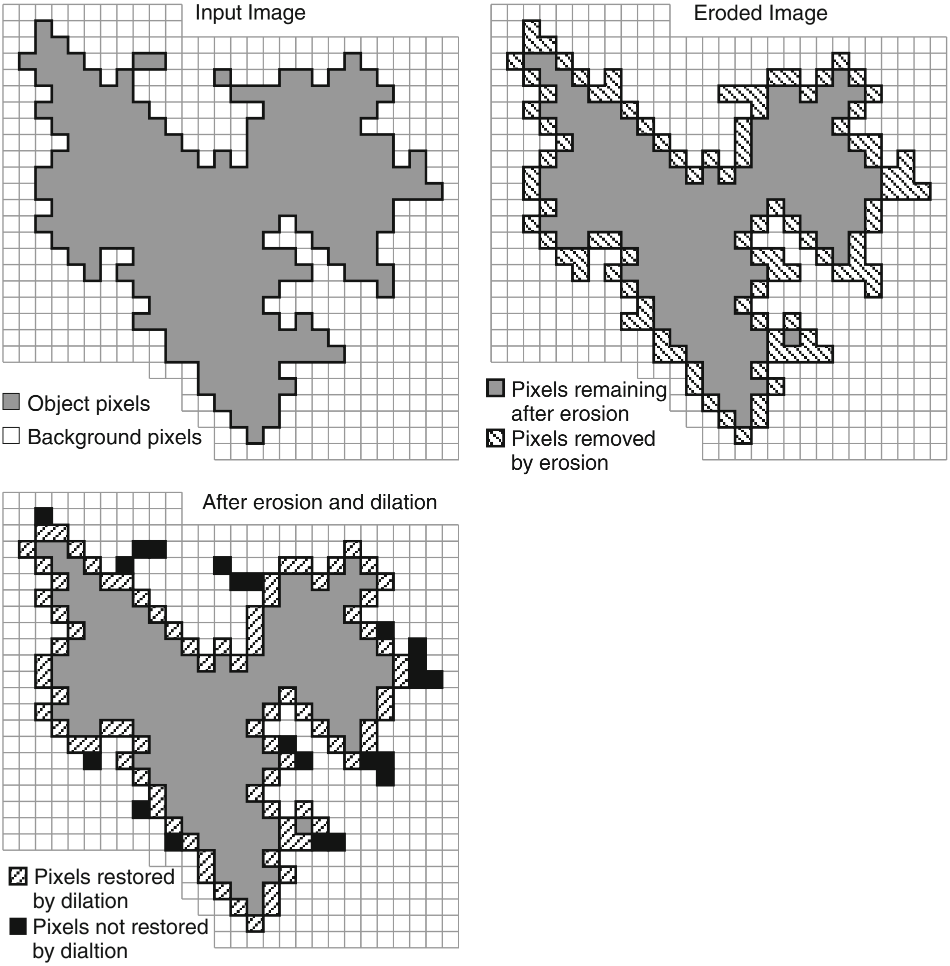

Input Image
Object pixels
Background pixels
Eroded Image
Pixels remaining after erosion
Pixels removed by erosion
After erosion and dilation
Pixels restored by dilation
Pixels not restored by dialtion

Fig. 17.26
Edge smoothing by erosion and dilation (*upper left*). Input image (*upper right*). After erosion [*skw*] (*lower left*). After erosion and dilation [*skw, exw*]

In some circumstances, it may be advantageous to reverse the order of execution:

$$\text{opening}(N), \ \text{closing}(N).$$

17.3.7.2 Smoothing the Chain Code

Consider the chain code sequence "..., 0, 0, 0, 1, 7, 0, 0, 0,...." It is immediately obvious that the sub-sequence "1,7" is a small glitch. This might arise from camera noise and digitisation effects, and should be replaced by "0, 0". This makes the edge represented by the sequence (...0, 0 0, 0, 0, 0, 0, 0,....) smoother (*Fig. 17.27a*). In a similar way, the sharp corner represented by the chain code sequence "..., 0, 0, 2, 2, ..." could be replaced by the smoother curve represented by "..., 0, 1, 2, ..." (*Fig. 17.27b*). We can, of course, take this further and look for more complex sub-sequences which represent sharp corners and glitches. For example, the chain code sequence "..., 0, 0, 0, 0, 2, 2, 2, 2, ..." might be replaced by "..., 0, 1, 1, 1, 2, ..." (*Fig. 17.27c*).

A fast low-cost edge-smoothing device based upon the chain code can be built using a shift register and look-up table (*Fig. 17.28*). The same general principle could be used to good effect in software.

17.3.7.3 Smoothing the Polar Vector Representation

In one of the simplest edge-smoothing schemes based on the PVR, two adjacent elements are merged if they are nearly aligned. Thus, two consecutive PVR elements, $[r_1, \theta_1]$, $[r_2, \theta_2]$ are replaced by a single vector $[r_{1,2}, \theta_{1,2}]$ if θ_1 and θ_2 are nearly equal (see *Fig. 17.29*). The resultant vector is obtained using the standard vector addition formulae:

$$r_{1,2} = \text{sqrt}\left\{[r_1 \ \cos(\theta_1) + r_2 \ \cos(\theta_2)]^2 + [r_1 \ \sin(\theta_1) + r_2 \ \sin(\theta_2)]^2\right\}$$
$$\theta_{1,2} = \tan^{-1}([r_1 \ \sin(\theta_1) + r_2 \ \sin(\theta_2)]/r_1 \ \cos(\theta_1) + r_2 \ \cos(\theta_2)])$$

The rule for deciding whether or not to merge two PVR elements may be expressed in a number of ways:

1. Merge all pairs of adjacent elements $[r_i, \theta_i]$ and $[r_{i+1}, \theta_{i+1}]$, if $|\theta_i - \theta_{i+1}| \leq d_1$, where d_1 is some parameter we choose to suit the application requirements.
2. Merge all pairs of adjacent elements $[r_i,\theta_i]$ and $[r_{i+1},\theta_{i+1}]$, if $|r_{ij} - r_i - r_{i+1}| \leq d_2$, where d_2 is some parameter we choose to suit the application requirements.

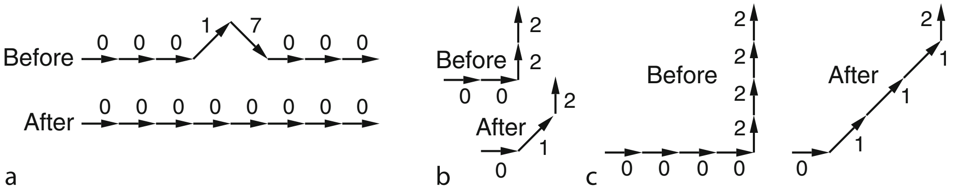
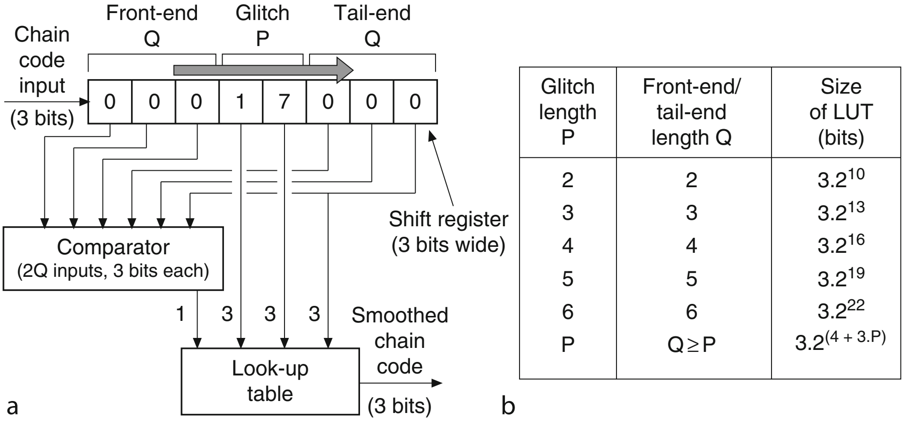


Fig. 17.27
Edge smoothing: filtering the chain code. (a) Replacing glitches (b) Smoothing corners (mild) (c) Smoothing corners (more pronounced)



Glitch length P	Front-end/ tail-end length Q	Size of LUT (bits)
2	2	3.2^{10}
3	3	3.2^{13}
4	4	3.2^{16}
5	5	3.2^{19}
6	6	3.2^{22}
P	Q≥P	$3.2^{(4 + 3.P)}$

Fig. 17.28

Filtering the chain code. (a) This circuit can detect and remove glitches of the form A^Q,S(P),A^Q, where P and Q are integers and S(P) is a chain-code sequence of length P. The front-end and tail-end sub-sequences are both of length Q. (b) Size of the look-up table as a function of glitch length (P) and front-/tail-end length (Q)

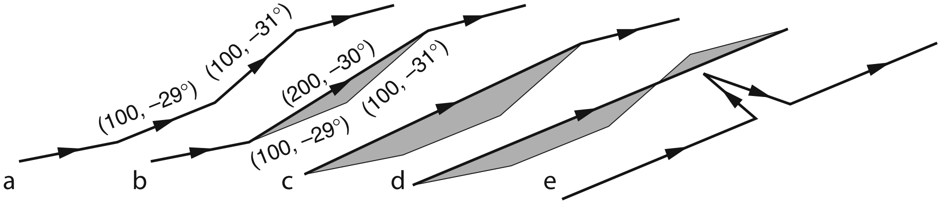


Fig. 17.29

Edge smoothing: processing the PVR. (a) Original edge, consisting of 4 PVR elements. (b) Replacing two nearly collinear vectors by vector addition. (c) Replacing three vectors. Notice that the error, measured by the shaded area is small, whereas the length of the resultant vector is quite long. (d) Replacing four vectors. Notice that there are both small positive and small negative areas (i.e., shaded areas above and below the resultant vector), which tend to "cancel" one another. (e) Glitch can be detected quite easily since there are two long vectors that are nearly in line and two short ones between them

3. Merge all pairs of adjacent elements [r_i,θ_i] and [r_{i+1},θ_{i+1}], if the triangle enclosed by the three vectors [r_1,θ_1], [r_2,θ_2] and [$r_{1,2}$,$\theta_{1,2}$] has an area that is less than d_3, where d_3 is some parameter we choose to suit the application requirements.
4. Investigate all pairs of elements in the PVR coding of an edge but merge only one pair at a time. Obviously we could choose to merge a pair of PVR vectors if they minimise any of the criteria used in 1–3.

Defining suitable rules for detecting and removing glitches, such as

$$\ldots, [100,\ 37^\circ],\ [1,\ 127^\circ],\ [1, -52^\circ],\ [100,\ 37^\circ], \ldots$$

will be left as an exercise for the reader. (The rules become fairly obvious if we represent the PVR graphically.) Notice that a glitch can always be identified from the PVR as being either "hair" or "crack" (➋ *Fig. 17.22*), since the inside of the blob is always on the left (or right) of the vectors.

➋ *Figure 17.30* shows the effects of applying PVR edge smoothing to the outline of a model car. By applying the smoothing operator iteratively, long PVR elements are created, which in this case can be identified easily with the roof, windscreen and hatchback of the vehicle. The edge smoothing process is, in effect, performing a feature-identification operation. This very useful side effect is not shared by any other edge-smoothing procedure. This illustrates an important general principle: We must always be alert to such "side effects", since they can significantly alter our judgement of a procedure.

17.3.7.4 Dog-on-a-Leash Algorithm

Another edge smoothing operator can be defined in terms of the chain code, using a model based upon the idea of a person taking a lazy dog for a walk (see ➋ *Fig. 17.31*). The point C represents the person and A the dog, which is always a given distance behind. On this occasion, distance is measured in terms of the number of chain code steps between dog and the person. The dog is on a leash whose centre point is D. (The model is somewhat unrealistic because the length of the lead varies, while the dog is a set number (N) of paces behind the man.) The locus of D as the person and dog traverse the edge of the blob forms the desired smoothed edge.

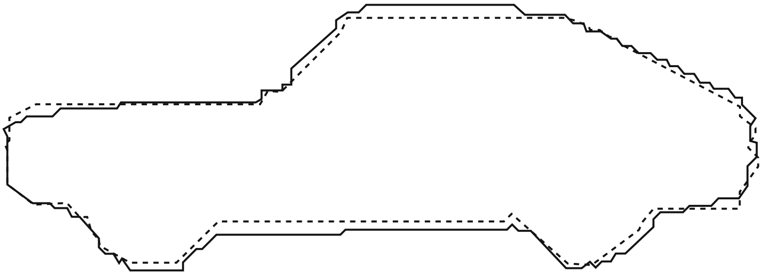


Fig. 17.30
PVR edge smoothing example

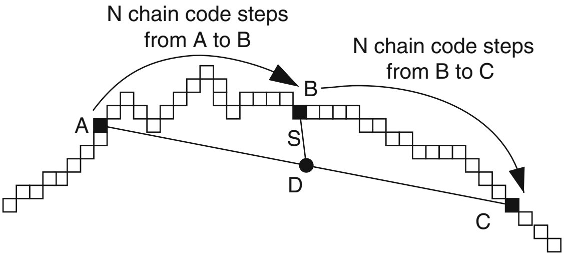


Fig. 17.31
Dog-on-a-leash algorithm for edge smoothing. The locus of D forms the smoothed edge. The distance S is a measure of the local curvature and provides a useful method for detecting corners

This operation can be expressed conveniently in terms of the chain code and polar vector representations.

An added benefit of this edge-smoothing algorithm is that it is very closely related to another which highlights corners and therefore acts as feature detector. Consider ➋ *Fig. 17.31* again. The point B is mid-way between A and C, again measured in terms of the number of chain-code steps. The Euclidean distance between B and D is measured for each edge point. (B is assigned an intensity value proportional to S.) The larger S is, for a given arc length N, the smaller the radius is of that circle that intersects points A, B and C. Hence, bright points indicate corners (i.e., points of high curvature). Clearly, it is possible to combine both edge smoothing and corner detection into a single routine. As before, this "free benefit" can make one technique preferable over another.

17.3.7.5 Processing the Binary Image as a Grey-Scale Array

Consider the following QT command sequence applied to a binary image (➋ *Fig. 17.32*) smooth_edge(N) defined as N•lpf, thr, for some suitable value of N. (Typically, $1 \leq N \leq 10$.) Using the blurring operator (i.e. low-pass filter) *lpf* on binary images is perfectly acceptable, since they are simply special cases of grey-scale images. (We use the convention that white is represented by level 255 and black by zero.) By applying *N•lpf* to a binary image, we generate a multi-level ("grey scale") image in which the intensity at any given point indicates how many white pixels lie nearby. If we then threshold the resulting image, we create a new binary image. The overall effect is to smooth edges in a way that is at least intuitively acceptable. The parameter N can, of course, be adjusted to taste: larger values of N produce a greater degree of smoothing.

There is an additional point to be made here. We may wish to smooth only convex corners and not concave ones, or vice versa. This can be achieved by smoothing edges as above and then combining the processed image with the original (binary) image. The QT sequence

smooth_convex_corners(N) defined as wri, smooth_edge(N), rei, mxi

(achieves this Note, *wri* saves an image on disc and *rei* reads an image from disc.)

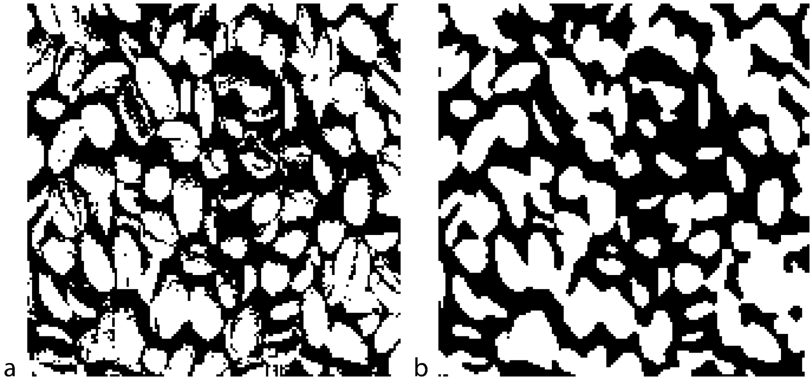


Fig. 17.32
(a) Original (binary) image, derived from ➋ *Fig. 17.7* by applying [*heq, thr(100)*] (b) Result of applying [*raf, thr*] to (a)

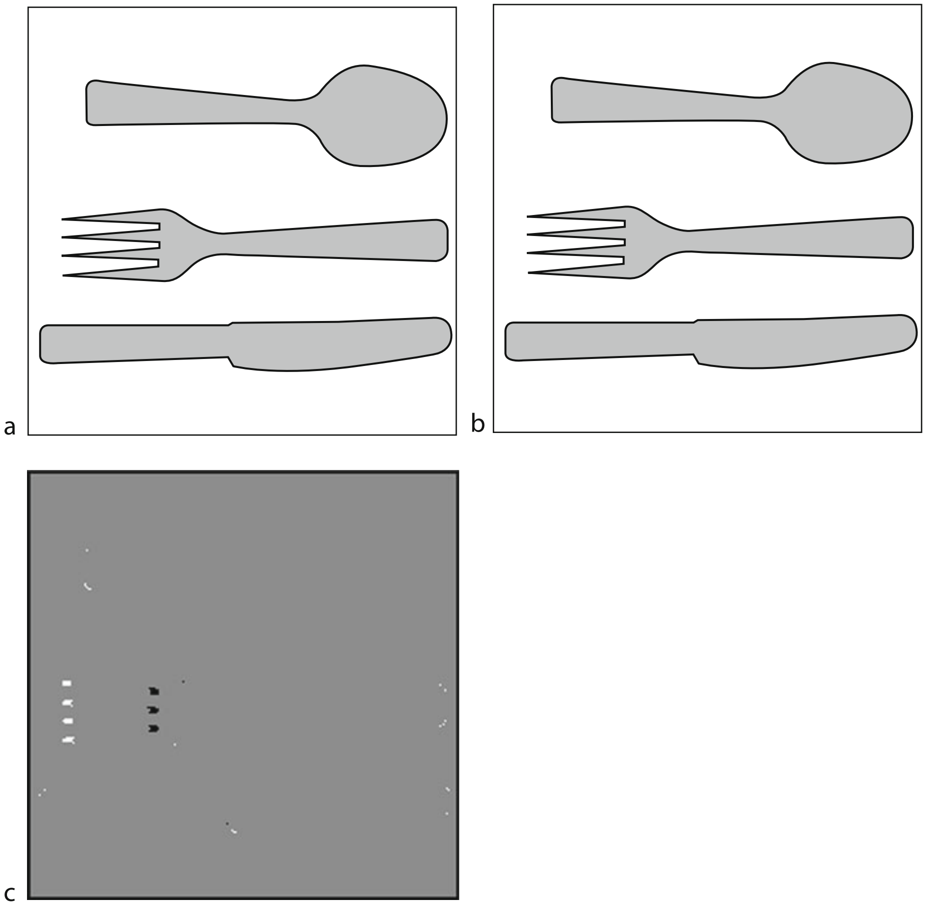


Fig. 17.33
(a) Original (binary) image (b) After applying [*3 • (raf, thr)*]. The tips of the tines and the bottoms of the spaces between the tines have been smoothed most effectively. (c) Result of subtracting (b) from (a)

A typical result obtained by applying *smooth_convex_corners (3)* to a binary image is shown in *Fig. 17.33*. Notice that the tips of the tines and the bottoms of the spaces between the tines can be detected and hence smoothed separately.

17.3.7.6 Edge Coding

Several more algorithms for edge smoothing can be derived from a type of edge-tracing algorithm which has not been mentioned previously. This starts at some convenient edge point P which might, for example, be the right-most point lying along the horizontal line passing through the centroid (CG in *Fig. 17.34*). Let D_i denote the distance around the edge from P to a given edge point Q_i. Furthermore, let R_i be the distance from the centroid of the blob to Q_i. Notice that D_i may be estimated from the chain code simply by counting

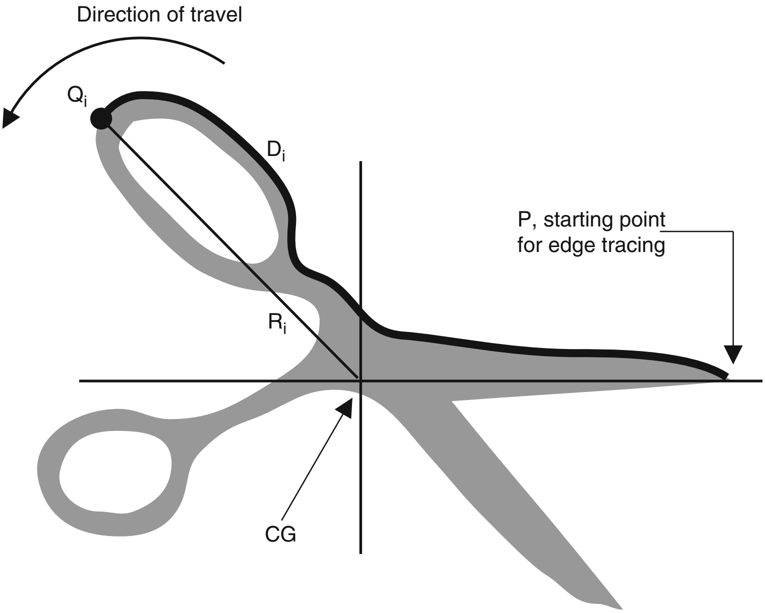


Fig. 17.34
Coding the edge of a blob by expressing the radius R_i (measured from the centroid CG) as a function of the distance D_i around the edge. D_i is measured by ($\surd 2 \bullet N_o + N_e$)

odd-numbered values, each of which contributes an increment of √2, and even numbered values, which contribute an amount 1.0. Clearly, R_i is a single-valued periodic function of D_i. The period is equal to the crude perimeter estimate

$$\sqrt{}2N_o + N_e$$

where N_o is the number of odd-numbered chain-code elements and N_e is the number of even-numbered elements.

There are several ways to devise an edge-coding scheme based on this coding technique. All of the following suggestions are based on conventional low-pass filtering methods typically used by electronics engineers to attenuate the high-frequency components in a time series:

- Simple N-element averaging (typically N=3, 5, 7, ...).
- Weighted-value smoothing, an extension of the previous idea which reduces the effects of distant events on the output signal.
- Fourier transform. This uses the fact that R_i is a periodic function of D_i.
- Walsh/Haar transform. Again, this makes use of the periodicity of R_i.

17.3.8 Techniques Based on Histograms

The importance of the histogram for machine vision can hardly be overstated, since parameters for many useful functions can be derived from it. Hence, the ability to find novel ways to compute the histogram opens up new opportunities for improving processing speed. Sometimes, as we will see, it is not necessary to calculate the complete histogram, since only one or two numeric values derived from it may be important for a specific application. We will touch on this point several times subsequently in this section.

The direct implementation of the histogram is straightforward. We scan the whole image and, at each pixel visited (i,j), we find the intensity A(i,j) and then increment the A(i,j)-th element in a (1+W)-element vector. (The minimum intensity in the image is zero, and W denotes the maximum intensity.) Initially, every element in this vector is set to zero. The U$^{\text{th}}$ element in the histogram indicates how many times the intensity value U occurs in the image.

Several approximate methods for computing the intensity histogram are known. First, we may sample the image to obtain an approximation to the histogram that is less accurate but can be calculated more quickly. The simplest way is to sub-sample the columns and/or rows and then calculate the intensity histogram on the lower resolution image. It is often found that reducing the spatial resolution of an image makes little difference to the result, since calculating the histogram is an integrative process. Of course, pictures which contain a few large nearly-homogenous regions produce the most stable histograms when calculated in this way. The effects of changing the resolution can be investigated experimentally by running the following QT program with different values of N:

```
N•psq,      % Halve spatial resolution when we increment N by 1
N•pex,      % Return image to the original size
hpi.        % Calculate the histogram
```

Another reduced-precision method relies on the use of a Hilbert-curve sampling pattern (➔ *Fig. 17.35*). Notice that the Hilbert curve meanders through the image, providing a uniform sampling density everywhere. The sampling density can be controlled by a single parameter.

An option that we may use very effectively in some situations derives the histogram in the normal way but only on some important sub-region of the image (known as a *region of interest*,

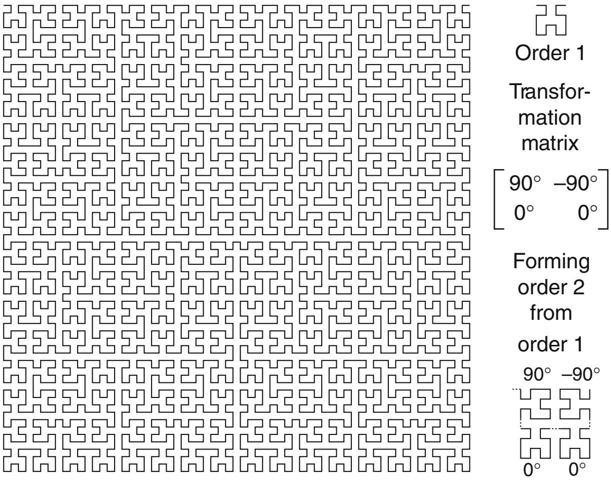


Fig. 17.35

Hilbert curve of order 5. Notice that the sampling density is uniform and can be varied at will by using Hilbert curves of different orders to sample the image. Higher-order Hilbert curves are derived by applying the transformation matrix shown above recursively

or *ROI*), ignoring other irrelevant parts. (This may achieve more useful results than the direct method, since non-relevant parts of the picture are ignored.)

It is possible to invent lots of other sampling schemes, which are combinations of or derivations of these basic ideas. One such hybrid scheme uses a low-precision scan (e.g., Hilbert-curve or raster) to locate a region of special interest and then employs higher-resolution sampling (perhaps of a different kind) to derive a high-precision histogram for a small part of the image.

17.3.8.1 Cumulative Histogram

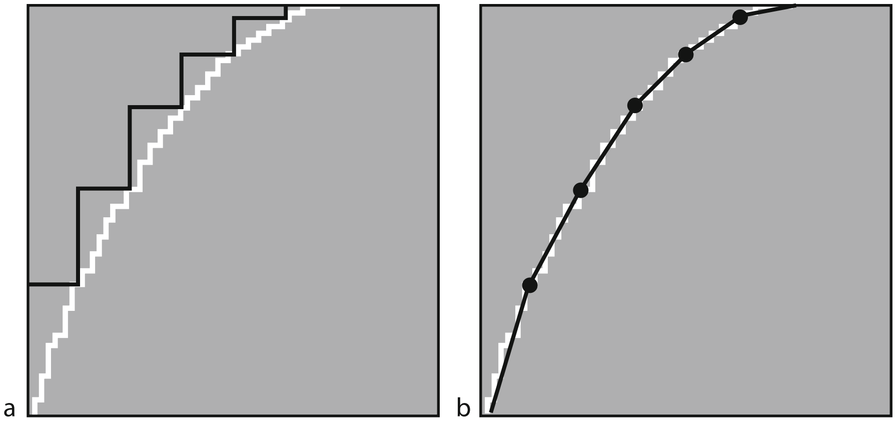
If the histogram is defined as the elements of the vector {H(i), i = 0,1,2, ..., W}, then the cumulative histogram is given by the recursive relationship

$$C(i) = C(i-1) + H(i), \text{with } C(0) = 0.$$

Clearly, the cumulative histogram can be derived simply, by integrating the intensity histogram. However, there is another possibility which might, in some situations, be superior. Assume that we possess some very fast (hardware) device for performing thresholding and pixel counting. (A very simple trigger circuit and pixel counter will derive one new element for the cumulative histogram every frame period.) We can easily compute the cumulative histogram by progressively adjusting the threshold parameter. This technique lends itself very well to sampling the cumulative histogram. As before, we can exchange processing speed for precision. We have to realise that the cumulative histogram finds its main application in deriving the look-up table used for histogram equalisation [*heq*] and that the cumulative histogram is monotonically increasing. With this in mind, we can see the advantage of sampling the cumulative histogram; we can use simple (e.g., linear) interpolation and just a few sample points on the cumulative histogram to obtain an approximation to the exact histogram-equalisation intensity-mapping function (➲ *Fig. 17.36*).

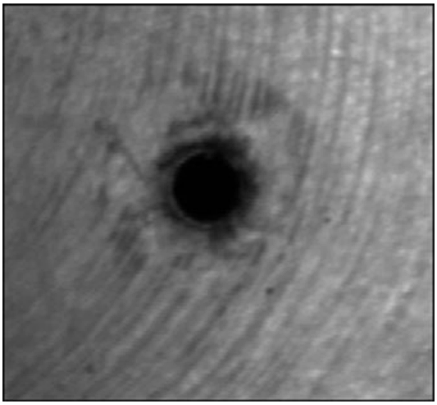
A completely different approach to approximating histogram equalisation relies on the fact that, in many applications, the images obtained at nearby moments in time are statistically very similar. In this situation, successive frames in the signal obtained from a video camera will generate similar histograms. A small object moving in front of a featureless background will generate an almost identical histogram in successive video frames (➲ *Fig. 17.37*). A web-like product will similarly generate very similar histograms from images derived from areas that are close together. (Many changes in the web are due to slowly changing factors, such as roller wear, coating material viscosity, etc. Of course, defects in the web may cause significant and very sudden changes in the histogram.) When the histogram is changing slowly, the histogram equalisation mapping function derived on one image can be applied to another image, digitised soon afterwards. This provides the basis for an approximate method of performing histogram equalisation in real time (i.e., full video frame rate) but with a 2-frame delay [8].

1. Frame 1: Calculate the intensity histogram of image 1.
2. Frame 2:
   (a) Derive the cumulative histogram. (Integrate the histogram.)
   (b) Rescale the cumulative histogram to form the mapping function.
   (c) Load this function into the look-up table.
3. Frame 3: Apply the mapping function to video frame 3.



■ **Fig. 17.36**

(a) Approximating the cumulative histogram. *White*: cumulative histogram of the beans image (See ➔ Table G.4). *Black*: cumulative histogram generated by applying *sca(3)* to the beans image. Notice that the staircase touches the true cumulative histogram exactly at seven places.
(b) Applying linear interpolation between these points produces a close fit to the true cumulative histogram



■ **Fig. 17.37**

The intensity histogram is very nearly constant, even if the major feature of the image wanders over a large area. Since the histogram of the machined background is nearly constant everywhere, it does not matter exactly where the camera is located in relation to the hole

17.3.8.2 Local-Area Contrast Enhancement

This is a non-linear filtering technique that is particularly useful for processing images derived from textured surfaces [5]. Histogram equalisation is performed on a relatively small region of the input image (e.g., 25 × 25 pixels) but only one pixel value is retained from this calculation. The processing window is scanned across the picture in a raster-like manner. At each step, the histogram-equalisation process is performed. Let us assume that the image contains N^2 pixels and that the processing window contains P^2 pixels. Then, the direct implementation requires that we performing N^2 histogram equalisation operations on P^2 pixels, keeping only N^2 pixels. We can do better than this! Let us consider what histogram equalisation actually means. Suppose that after histogram equalisation has been performed within a certain processing window $V_{i,j}$, we find that the new intensity value of a certain pixel (i,j) is $Q_{i,j}$. (As usual, the maximum value is W and the initial value of pixel [i,j] is $A_{i,j}$.) Then, a proportion $Q_{i,j}/W$ of the pixels in the window $V_{i,j}$ were darker than $A_{i,j}$. This is useful because it allows us to perform histogram equalisation for a single point, simply by counting how many pixels in the window $V_{i,j}$ are darker than $A_{i,j}$. This little trick has no real value for (full image) histogram equalisation but it can be useful for local-area histogram-equalisation, where we have to perform the same operation many times. The revised algorithm for local area contrast enhancement is therefore as follows. (The processing window $V_{i,j}$ is centred on the point [i,j]. To ensure that $Q_{i,j}$ is a true count of the number of pixels in $V_{i,j}$ that are brighter than $A_{i,j}$, we assume that $V_{i,j}$ contains $P^2 \leq W$ pixels.)

1. Set the raster-scan counters [i,j] to their initial values.
2. Count how many pixels in $V_{i,j}$ are darker than $A_{i,j}$. Denote this value by $Q_{i,j}$. (Notice that $0 \leq Q_{i,j} \leq P^2 \leq W$.) Set the intensity at point [i,j] in the output image to $Q_{i,j}$.
3. Increment the raster-scan counters and repeat step 2 for all pixels in the input image.

Notice that we have managed to reduce the calculation to one in which we merely perform $(P^2 - 1)$ comparisons for each pixel. We have eliminated the need to calculate the histogram directly.

17.3.8.3 Percentage Threshold

Let us turn our attention now to another histogram-based function, that of finding a threshold parameter which will divide the image in a given ratio. That is, given a parameter R (where $0 \leq R \leq 100$), what threshold parameter will segment the image so that R% of the pixels becomes black and $(100 - R)$% white? The cumulative histogram allows this parameter to be calculated directly. A particularly useful trick when using PQT is to perform histogram equalisation followed by fixed-value thresholding. For example, [*heq, thr*] segments the image so that 50% of the image is black, while [*heq, thr(200)*] makes 78% black [= (200/255) × 100%]. In more general terms, the following program makes R% of the thresholded image black.

```
heq,                 % Histogram equalisation
T = 255 ∗ R/100,     % Rescale percentage R to intensity scale, [0, W]
thr(T).              % Threshold so that R% is black and (100−R)% is white
```

(This is a possible way to implement the QT operator *pth.*) If we compute the intensity histogram first, we can terminate the integration process early, when R% of the total number of pixels in the image has been taken into account.

17.3.8.4 Contrast Enhancement

The PQT operator *enc* calculates the minimum and maximum intensities in an image and then rescales all of the intensity values, so that the darkest pixel becomes black and the brightest becomes white. This simple procedure is very effective in many applications but it is sensitive to noise. A more robust scheme for contrast enhancement is as follows: Let T(R) be that threshold parameter which ensures that R% of the threshold image is black. Let R_1 and R_2 be two parameters defined by the programmer. (Typically $R_1 = 5\%$ and $R_1 = 95\%$.) Then, we modify all of the intensities in the image, so that pixels of intensity less than or equal to $T(R_1)$ become black, pixels of intensity greater than or equal to $T(R_2)$ become white, and those in the intermediate range $[T(R_1), T(R_2)]$ are rescaled in a linear manner. This calculation is easily achieved using the cumulative histogram. However, note that we do not need all of the information in either the histogram or the cumulative histogram. We need only two values: R_1 and R_2. When using certain types of hardware (for example, the fast threshold and pixel-counting device mentioned above), we may not need to compute the histogram at all. Another point: once the values of R_1 and R_2 have been found, we can use a look-up table, rather than an arithmetic unit, to perform the intensity-scale mapping.

17.4 Approximate and Heuristic Methods

So far in this chapter, we have concentrated almost exclusively on algorithmic methods. Now, we turn our attention to approximate and heuristic techniques. There are several ways in which heuristics can be used in Machine Vision:

- To replace an algorithm
- To extend the range of application of an algorithm
- To make an algorithm run faster
- To do something useful when there is no known algorithm
- To do something useful when using an algorithm would be impractical

Machine Vision is a pragmatic subject. If we were to insist on using only algorithmic methods, our scope for building machines that are useful in practice would be severely curtailed. In this section, this point will be illustrated many times over.

17.4.1 Measuring Distance

Distance is an important parameter that is required in a very wide range of inspection applications. If we are asked to monitor the size distribution of granular material, we do not require a precise measurement of each particle. We often need to know the general trend, so that we can determine whether the particles are getting bigger or smaller. When grading carrots, parsnips, etc., we do not need to know the length precisely, since the concept of length is itself ill-defined for such an object. Similarly, when measuring the eccentricity in the placement of a piece of cherry on top of a cake, we do not need to know the result precisely. Again, there are fundamental limits, since the position of an object of indeterminate shape such as a piece of cherry cannot be defined precisely. (The centroid is often used to define position but there is nothing absolute about this.)

As we will see in this section, there are many ways to measure distance and hence linear dimensions. These alternative measurements do not always coincide exactly with what we normally call "distance", which we can measure using a tape-measure, micrometer or caliper gauge. However, they are often easier to calculate and will suffice for many applications. We use the term *Euclidean distance* to refer to the measurement that these instruments yield. In everyday life, we often use the phrase *as the crow flies* to emphasis that we are using the Euclidean distance. Astronomers and radar systems engineers relate distance to the time it takes for light or radio waves to travel between two points. These are simply ways of measuring Euclidean distance. However, vehicle drivers are well aware that a meaningful concept of "distance" must be related to the available roads. In this situation, the Euclidean distance metric is not the most meaningful.

The grass-fire transform was defined in ➋ Sect. 17.3 in terms of the Euclidean distance. As we will see later, it is often implemented using iterative techniques, which do not measure distance explicitly. Certain binary morphological operators can be regarded as performing a similar function to a *go/no-go* gauge and hence are size-sensitive. We will also describe a grey-scale morphological operator that implements the grass-fire transform and hence implicitly measures distances.

Let us now define the concept of Euclidean distance in formal terms. We will do so for a 2-dimensional space since we are interested in pictures. However, we should point out that the concepts we are about to discuss can be extended easily to 3- and higher dimensional spaces. Let $[X_1, Y_1]$ and $[X_2, Y_2]$ be two points within an image. Then, the Euclidean distance between them is

$$D_e([X_1, Y_1], [X_2, Y_2]) = \surd\{(X_2 - X_1)^2 + (Y_2 - Y_1)^2\}$$

As we hinted earlier, there are numerous alternatives to the Euclidean distance, which may be rather easier/faster to calculate in practice and which are intellectually just as valid. Here are the definitions of just two of them:

$$\text{Manhattan (city block) distance :} \quad D_m([X_1, Y_1], [X_2, Y_2]) = |X_2 - X_1| + |Y_2 - Y_1|$$
$$\text{Square distance :} \quad D_s([X_1, Y_1], [X_2, Y_2]) = \text{MAX}(|X_2 - X_1|, |Y_2 - Y_1|)$$

The Manhattan distance indicates how far a person would walk or drive in some North American cities, where the roads form a neat orthogonal grid. When travelling in such a city, it is perfectly natural to relate journey times to the Manhattan distance (i.e., number of city blocks to be traversed), rather than the Euclidean distance. The Square Distance is also more natural in some circumstances. For example, a *bang-bang* servomechanism operating a multi-axis robot will respond in a time that is proportional to the *Square Distance* to be travelled.

All three of these metrics have their uses for machine vision. In some situations, we may be able to use the squared Euclidean distance, D_{e2}, ($= D_e^2$) and thereby avoid the square-root operation.

$$D_{e2}([X_1, Y_1], [X_2, Y_2]) = (X_2 - X_1)^2 + (Y_2 - Y_1)^2$$

In mathematical terms, D_{e2} is not, strictly speaking, a distance metric, since it violates one of the essential conditions, namely that

$$D(A, C) \leq D(A, C) + D(B, C).$$

In mathematics, two other conditions are imposed on a distance metric:

$$D(A, A) = 0$$

and

$$D(A,B) = D(B,A)$$

Note that D_e, D_m and D_s are all particular cases of the *Minkowski r-distance*, which is defined thus:

$$L_r([X_1,\ Y_1],\ [X_2,\ Y_2]) = \sqrt[r]{(X_2 - Y_1)^r + (Y_2 - Y_1)^r}$$

In the limit as r tends to infinity, L_r approaches the Square distance. Hence, we refer to D_s as the L_∞ metric. Clearly, L_1 is the Manhattan distance and L_2 is the Euclidean distance. The equation

$$L_r([X_1,\ Y_1],\ [X_2,\ Y_2]) = K,$$

where K and r are constants, defines a symmetrical closed figure. Just as a circle is defined by the Euclidean distance, D_m produces a diamond-shaped figure, while D_s generates a square. The linear weighted sum ($D_m + \surd 2\ D_s$) generates an octagon, which is a more accurate approximation to a circle than either D_m or D_s alone.

The relevance of this discussion becomes clearer when we consider the effects of various operations in digital images. We have already seen in ➋ *Fig. 17.15* that applying the QT operator *exw* (expand white regions) more than once generates a 3 × 3 square from a single isolated white pixel. In other words, *exw* effectively draws contours of constant Square distance. On the other hand, *exw4*, the 4-neighbour version of *exw*, generates a diamond and hence is linked to the Manhattan distance.

Note that *exw4* may be implemented using the QT sequence

$$\mathrm{glc}(0,1,0,1,1,1,0,1,0),\ \mathrm{thr}(1).$$

The sequence

$$N \bullet (\mathrm{glc}(0,1,0,1,1,1,1,0,1,0),\ \mathrm{thr}(1),\ \mathrm{glc}(1,1,1,1,1,1,1,1,1),\ \mathrm{thr}(1))$$

generates an octagon.

Suppose now that we wish to find the farthest point in a blob, measured from some convenient reference point, such as its centroid. (This forms a useful basis for measuring object orientation.) An obvious way to do this is to scan the image computing D_e, D_m or D_s, as appropriate, for each pixel. However, this is wasteful of computing effort, if we need to do this calculation often. We can, for example, employ the QT operator *hic* to generate a reference image, which we then store for future use. *hic* generates an image in which the intensity at each point (i,j) is proportional to the Euclidean distance between (i,j) and the centre of the image. The image to be analysed is then shifted so that the reference point is coincident with the centre of the stored image (i.e., the brightest point in the output of *hic*), and these two images are combined, using *mni* (point-by-point minimum of intensities). The pixel in the resulting image at which the intensity is a maximum indicates the pixel within the blob that is farthest from the reference point.

The grass-fire transform is based on the concept of distance from the nearest edge pixel (➋ *Fig. 17.38*). While we describe the grass-fire transform informally using the concept of (Euclidean) distance, a practical implementation is most unlikely to calculate this or any of the aforementioned distance metrics directly. One possible implementation of the grass-fire transform employs a 3 × 3 local operator, applied recursively until the image does not change.

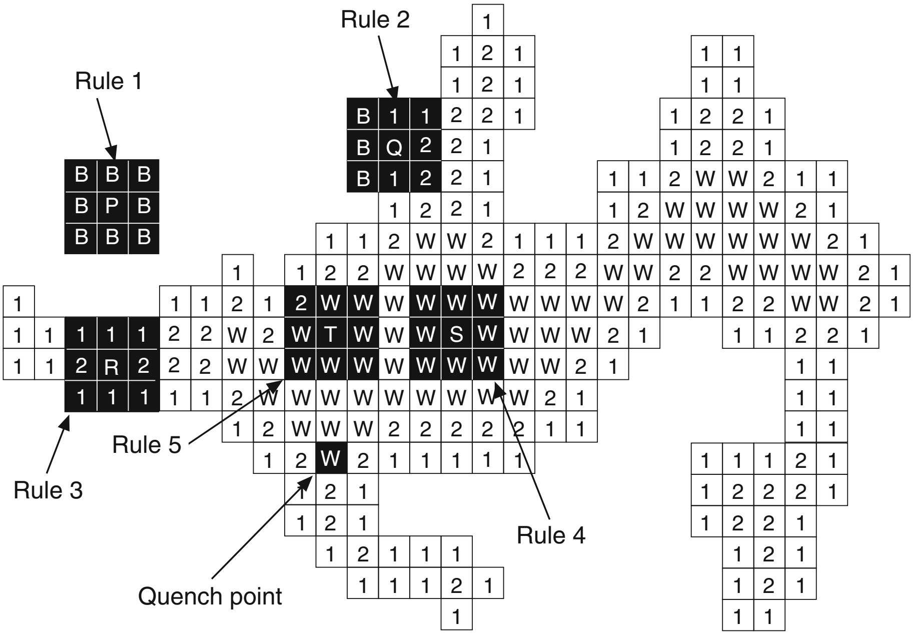


Fig. 17.38

Algorithm for the grass-fire transform. The black 3 × 3-pixel windows relate to cycle 1 (Rules 1 and 2), cycle 2 (Rule 3) and cycle 3 (Rules 4 and 5)

First, we have to decide whether to base the calculation on the 4- or 8-neighbours. (The latter effectively measures the Square distance to the nearest edge pixel. The former measures the Manhattan distance.) One possible implementation of the grass-fire transform is defined below. Initially, all pixels within an image are either white (level 255, inside a blob-like figure), or black (background). Consider pixel [i,j], whose intensity will be denoted by I(i,j). Those pixels in its (4- or 8-) neighbourhood will be denoted by N(i,j).

1. If I(i,j) = 0, [i,j] is a background pixel, in which case, we do not alter it. (Point P in *Fig. 17.38*)
2. If I(i,j) = 255 and at least one pixel in N(i,j) is black (level 0), we set I(i,j) to 1. These are the pixels just inside the edge of the blob (Point Q in *Fig. 17.38*).
3. If a pixel [i,j] already has a value other than black (0) or white (255), we do not alter it (Point R in *Fig. 17.38*).
4. If a pixel is white and all points in N(i,j) are white, the fire has not reached point [i,j] yet and we do not alter I(i,j) (Point S in *Fig. 17.38*).
5. If at least one of the points in N(i,j) has a value other than 0 or 255, we set I(i,j) to (Z+1), where Z is the minimum value for pixels in N(i,j). We ignore level 0, i.e., black, when computing I(i,j) (Point T in *Fig. 17.38*).

Rules 1–5 are applied to all pixels in the image. The whole process is then repeated recursively, until no pixels are altered (i.e., rule 5 is not invoked during a complete cycle through the image). Notice that we have not specified the order in which the pixels are processed, which gives us the chance to employ either a raster scan or an "onion peeling" scan. (The latter is faster and is discussed in the following section.)

If N(i,j) contains only the 4-neighbours of [i,j], the grass-fire transform generates a shading which records the Manhattan distance to the nearest edge point. On the other hand, if N(i,j) defines the 8-neighbourhood, the procedure records the Square distance. By alternating the procedure using 4- and 8-neighbours, we effectively combine the Manhattan and Square distances, so that the grass-fire transform records the octagon distance mentioned above (➲ *Fig. 17.15*).

While other procedures for implementing the grass-fire transform exist, we will not describe them here, other than to say that the SKIPSM approach (➲ Sect. 17.3.3 and ➲ Chap. 20) allows other distance measures to be defined by making use of arbitrarily-shaped structuring elements. The point to note is that there is no single "right way" to perform this calculation. We are used to measuring the shortest route between two points in terms of the Euclidean distance between them. However, we cannot say that the Euclidean distance is "right" and all other distance measures are "wrong." The implication for the grass-fire transform is that there is not just one procedure. Indeed, there are many and we must choose the one we want to use after considering its performance and its implementation.

Another point worth mentioning here is that the grass-fire transform can be regarded as providing a convenient alternative to an important subset of the binary morphological operators. The latter can be used to detect features that are within prescribed size limits (effectively measuring distance again). Notice, however, that the binary morphological operators do not measure distance or size explicitly; they are more akin to a *go/no-go* gauge, since they determine where an object that is the same size and shape as the structuring element will fit. The shape of the structuring element can, of course, be selected at will to be a circle, square, diamond or octagon, or any other figure of our choosing. Grey-scale morphological operators can be used to measure distance directly, using any distance metric. ➲ *Figure 17.39* explains

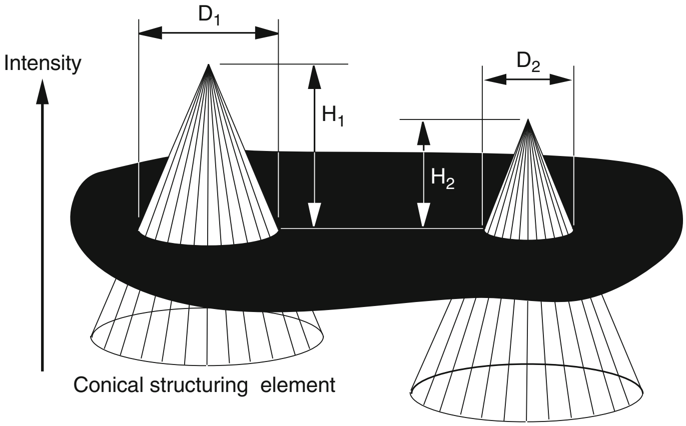


Fig. 17.39

Using grey-scale morphology to measure Euclidean distance. The structuring element is a right circular cone. The intensity surface (black region) contains two holes. The height of the tip of the cone (H_1, H_2) is directly proportional to the hole diameter (D_1, D_2)

how a grey-scale morphological operator can be used to implement the grass-fire transform based on the Euclidean distance.

17.4.1.1 Similarity and Distance

Suppose that we have derived an ordered set (vector) of numeric measurements from an image. These measurements might, for example, define the components of the colour vector at a given point, in which case, they would simply be the RGB (or HSI) signals, derived by a video camera. Another possibility is that these measurements summarise the overall texture within an image. Alternatively, the measurement vector might describe some feature such as a spot-like defect on a web. (Our inspection task then would be to identify the type of defect.) Whatever it represents, let us denote this set of measurements by $X = (X_1, X_2, ..., X_q)$, where q is a positive integer, typically in the range 2–20. In order use X to identify a given image feature it would be reasonable to compare it with a number of stored reference vectors, which we will represent by $Y_i = (Y_{i,1}, Y_{i,2}, ..., Y_{i,q})$, where i is a positive integer in the range [1,N]. A possible measure of *dissimilarity* between the features described by X and Y_i is given by the Euclidean distance:

$$D(X, Y_i) = \sqrt{\left(\left(X_1 - Y_{i,1}\right)^2 + \left(X_2 - Y_{i,2}\right)^2 + \left(X_3 - Y_{i,3}\right)^2 + \cdots + \left(X_q - Y_{i,q}\right)^2\right)}$$

There are many other ways to perform classification using the concept of distance as a measure of dissimilarity [1]. It is not our intention to review all of these in detail here. Instead, we will merely mention one important technique, nearest neighbour classification. (See ❯ Chaps. 16 and ❯ 24). In this scheme, we find the Y_i which minimises $D(X, Y_i)$ in order to recognise the feature or pattern associated with the vector X. Each of the Y_i is carefully chosen to represent or typify some cluster or group within the data set and is therefore called an *exemplar.* Y_i is associated with an appropriate label, C_i, which indicates the type of pattern, feature or event which gave rise to it. More than one Y_i may be given the same label, indicating that they correspond to the same pattern class. Several learning rules are known for designing the nearest neighbour classifier, some of which are liable to store a large number of exemplars. This can cause a problem not only for storage but also for the speed of computation. It is possible to improve the speed of the nearest neighbour calculation by subdividing the Y_i into groups, each of which is assigned a *super_exemplar.* The set of super-exemplars will be represented by $\{Z_j\}$. Notice that $\{Z_j\}$ is a proper sub-set of $\{Y_i\}$. (i.e., $\{Z_j\} \subset \{Y_i\}$) During a pre-processing phase, each of the super-exemplars, Z_j, is linked to another sub-set of the exemplars (Q_j), which are all known to be close to Z_j. (Again, $Q_j \subset \{Y_i\}$) Then, classification of X proceeds as follows:

1. Find a near neighbour to X among the set $\{Z_j\}$. This will be denoted by Z_j.
2. Find the nearest neighbour to X among the set Q_j. Denote this by W_k. (Assuming that the preprocessing phase has been properly designed, step 2 always finds the *nearest* neighbour to X among the set $\{Y_i\}$ and the number of computational steps is very much reduced. The details are tedious to explain and would distract us from our main theme.)
3. Associate the pattern which gave rise to the measurement vector X with class C_k. (Recall that W_k is the nearest neighbour to X among the set $\{Y_i\}$.)

This is an example of an heuristic being used to improve processing speed. The procedure is not algorithmic because we cannot guarantee that it will achieve a speed increase, although we can be certain that it will always find the nearest neighbour. The improvement in processing speed is achieved at a certain cost, by increasing the computational load in the pre-processing phase. This is akin to using an optimising compiler to reduce the computational effort needed at run-time.

17.4.1.2 Skeleton and the Medial Axis Transform

The skeleton of a blob-like object can be defined in terms of the grass-fire transform as the set of *quench points*. A quench point is one where two or more advancing fire lines converge at the same time and derives its name from the fact that there is no more combustible material there left to burn (➲ *Fig. 17.38*). Another definition of the skeleton relies on the fact that every point on it lies equidistant between two or more edge points. Neither of these definitions is very convenient as the basis for use in a practical application where high speed is needed. A more pragmatic approach is represented by the so-called *onion-peeling* procedure. For the sake of simplicity, in the following explanation, we ignore the possibility of there being lakes present, since they cause complications and obscure our understanding of the heuristic. The basic onion-peeling procedure may be described as follows (➲ *Fig. 17.40*): We trace the edge of the blob, as if we were chain coding it. As we do so, we delete (set to black) each pixel visited *unless it meets any one of these three conditions*:

1. It has only one white neighbour. Such a point is part of the skeleton and is therefore retained.
2. If it is critical for the connectivity among its remaining 8-neighbours. (c.f. the QT operator *cny*, (➲ Chaps. 19 and ➲ 41)) Again, such a point is part of the skeleton and is retained.
3. It is a skeleton point. (No skeleton point is ever erased.)

We repeat the process until there are no more changes possible. The resulting figure is the skeleton. However, the match-stick figure generated thus is not quite the same as that formed by the quench points generated by the grass-fire transform. (The latter is called the *medial axis transform*.) The onion peeling procedure has been found to be perfectly acceptable in all practical applications that the author has studied. Moreover, it somewhat faster and generates a "clean" skeleton, with very fewer short side branches than the quench-point method.

The onion-peeling procedure could be modified to generate the grass-fire transform, also. How this could be achieved is fairly obvious from ➲ *Fig. 17.40*.

The problem mentioned earlier relating to lakes is illustrated in ➲ *Fig. 17.41*. Notice how the skeleton is "tightened like a noose" around the lakes. To avoid this, we have to modify the procedure just described, so that we alternately trace once around each outer edge, then once around each inner edge. This involves additional rules, which inevitably slow down the procedure. For many purposes, however, the simpler heuristic will suffice, despite its known limitations on objects containing lakes.

17.4.1.3 Remarks

In this section, we have discussed several varied procedures, all of which are based on the concept of distance. First, we showed that it is possible, in many situations, to avoid the rather complicated calculation needed for the Euclidean distance. This can be achieved by using the

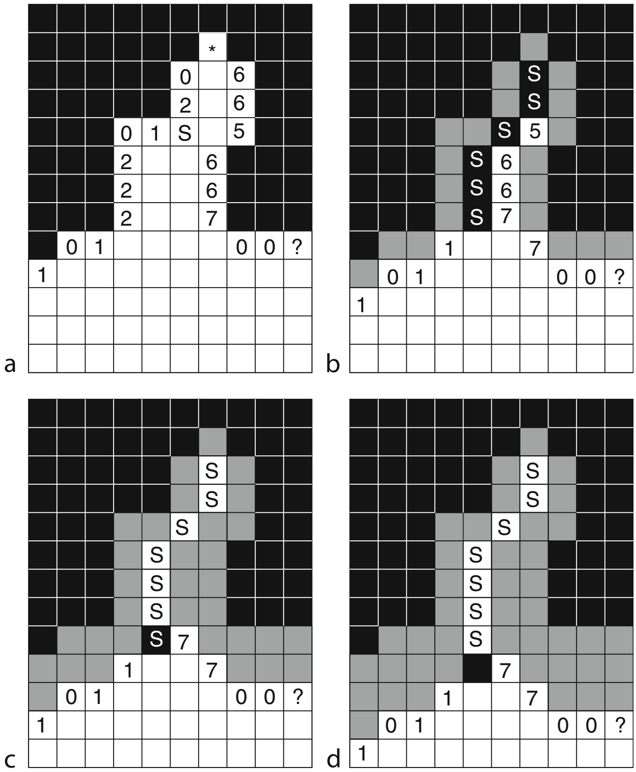


Fig. 17.40

Generating the skeleton of a blob by the onion peeling procedure. For clarity, only part of the blob is shown. When the edge follower reaches the right-hand-side of the sub-image shown here, it continues tracing the outer edge of the blob in a counter-clockwise direction until it eventually reaches the left-hand side again. Numerals 0–7 indicate the direction of travel of the edge follower. (same as chain code) 'S' is a skeleton point. (a) First cycle: '*' indicates the starting point. When the edge follower reaches the top-most '0', it tries to revisit the starting point. This signals the beginning of the second cycle. (b) The second cycle consists of these moves: S, S,5,6,6,7,7,0,0,...,1,0,1,1,S,S,S,S. (c) The third cycle consists of these moves: 7,7,0,0,...,1,0,1,1,S. (d) The fourth cycle consists of these moves: 7,7,0,0,...,1,0,1,1,S

Square or Manhattan distance measures instead. These can often be substituted directly for the Euclidean distance. We have shown that it is often possible to avoid the distance calculation altogether, by using iterative filtering techniques. We can also use iterative processing to implement the grass-fire transform, while the skeleton can be derived by a process that is effectively a modified form of edge tracing. Morphological operators can measure object size in a binary image without the use of distance measurement and, as we have seen in Sect. 17.3.3, we can implement these using look-up-tables. Although the nearest neighbour classifier is implicitly based on the concept of distance, the use of *near* neighbours allows us to improve the processing speed, by the simple expedient of eliminating calculations that are obviously unnecessary.

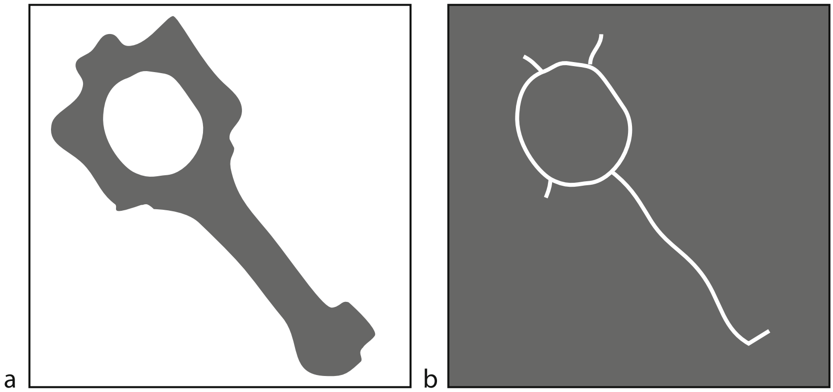


Fig. 17.41
An undesirable feature of the simple onion-peeling procedure. The skeleton is "tightened like a noose" around the lakes

17.4.2 Fitting Circles and Polygons to a Binary Object

In this section, we will consider various ways of representing a blob-like object with another figure (i.e., a circle or polygon). We have already considered this topic in some detail in Sect. 17.3.6, where we discussed the derivation of the convex hull. Intuitively, we may feel that this figure has a unique relationship to the shape that it represents, since it satisfies these three criteria:

(a) It is convex.
(b) It totally encloses the given shape.
(c) It is of minimal area.

While the figures about to be discussed do not possess all of these properties, they do, nevertheless, provide the basis for a range of useful analytical tools and in some circumstances can be used in lieu of the convex hull. Of course, there is no unique circle or polygon, including the convex hull, that properly represents *all* important features of a given blob. For this reason, we have to be careful to use the most appropriate one for a given application. In the following discussion, we will relax conditions (b) and/or (c), and impose other, equally valid criteria. Some of the figures that result are larger than the convex hull, while others do not contain all of the pixels within the given shape. We must not presuppose, however, that this invalidates them or makes them any less useful. Indeed, most of these alternative procedures are easier/cheaper to compute than the convex hull and, in many cases, possess features that are just as useful in practice. Since there is no "optimal" way of representing a blob by a circle or a polygon, the choice of which one to use in a given application depends upon its specific requirements.

17.4.2.1 Coin-in-a-Slot Procedures

Consider the task of inserting a coin into a slot-machine. We will generalise this to consider shapes other than circles being pushed through a slot. The size of the slot is, of course, deliberately chosen to match that of the object to be passed through it. An important parameter of a shape is the size of the smallest slot that it can pass through. However, this statement does not specify the problem fully In effect, there are several possible solutions, depending on the answers to the following questions:

(a) Are we allowed to manoeuver the shape as it goes through the slot, or must it pass "straight through"?
(b) Are we allowed to reorient the shape before we pass it "straight through" the slot?

A complex iterative procedure is required to draw the minimum-size circle that encloses a given blob, touching it at three points (➜ *Fig. 17.42*). One important circle that can be drawn rather more easily around a given blob is the one that intersects the two edge points that are farthest apart. This can be obtained by the edge-following process, and generates the bounding circle of minimum area.

Now, suppose that we are allowed to rotate the shape before passing it through the smallest possible slot. In this case, to find the minimum size of the slot we can use another edge-tracing procedure. Let E be the set of edge points, which contains N_E pixels. The Euclidean distance between two edge points A and B will be denoted by D(A,B), $A \in E$, $B \in E$. Given an edge point P, we simply find another edge point Q that maximises D(P,Q). Let us denote this value by D_P. We repeat this process for all edge points ($P \in E$), and then calculate the minimum value of D_P. A simple procedure of this kind requires $O({N_E}^2)$ time. A modest reduction in computational effort (about 50%) can be achieved by tracing only those edge pixels that have not already been considered. (Recall that D(A,B) = D(B,A). See ➜ *Fig. 17.43*.)

We will now describe a faster but approximate procedure for estimating the same parameter (➜ *Fig. 17.44*). Let us take two edge points A and B that are spaced $\lceil N_E/2 \rceil$ chain-code steps apart. (We are forced to round ($N_E/2$) up or down quite arbitrarily, since we do not know whether N_E is odd or even. The chain code must be regarded as forming a circular list structure, since the starting point is quite arbitrary and must be treated in the same way as all other

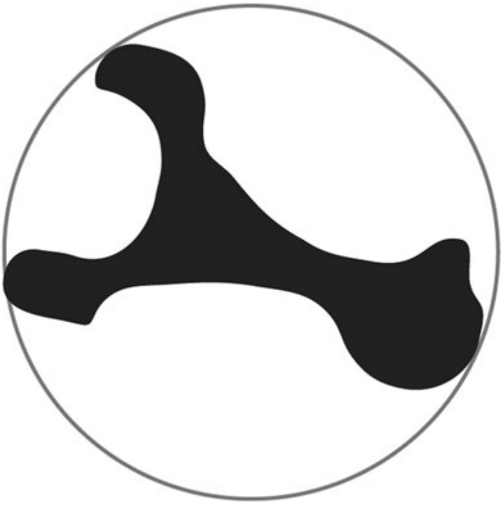


Fig. 17.42
The result of applying *mbc*. It usually touches the edge at three points

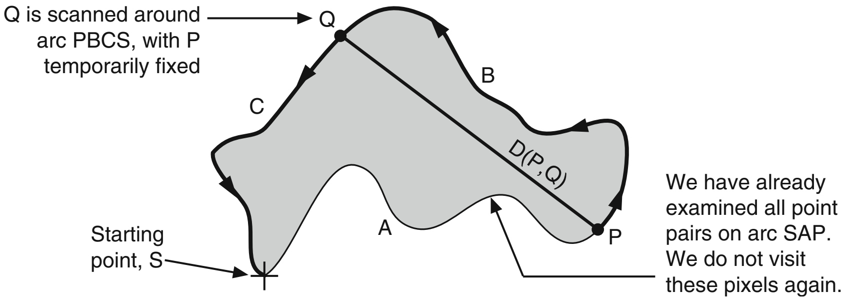

Q is scanned around arc PBCS, with P temporarily fixed
Q
B
C
D(P,Q)
We have already examined all point pairs on arc SAP. We do not visit these pixels again.
Starting point, S
A
P

Fig. 17.43
Scanning the edge to find the two points which are farthest apart

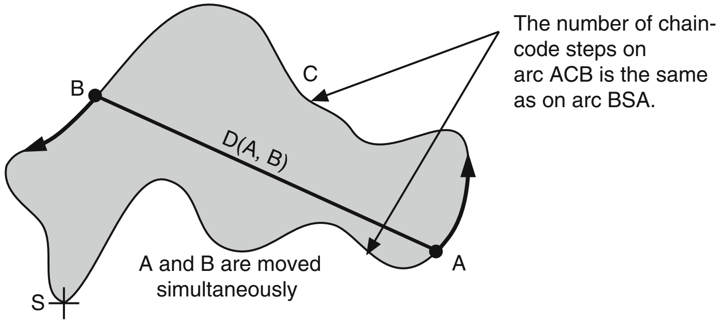

The number of chain-code steps on arc ACB is the same as on arc BSA.
C
B
D(A, B)
A and B are moved simultaneously
A
S

Fig. 17.44
Scanning the edge using two points (A,B) which are separated by ⌈$N_E/2$⌉ chain-code steps

points.) We then calculate the Euclidean distance between them, D(A,B). We then move A and B simultaneously around the edge of the blob, and find the position which maximises the value of D(A,B). At this position, we find the mid-point of the line AB, which is then taken to be the centre of the circle to be fitted to the given blob. This simple procedure does not always find the same circle as the previous method but it is likely to be much faster, since the computation time rises as $O(N_E)$.

17.4.2.2 Circumcircle Based on the Centroid

Another circle whose parameters are easy to compute is concentric with the centroid of the blob and has a radius equal to the distance from the centroid to the furthest edge point (➋ *Fig. 17.45*). This circle is usually larger than that produced by the three previous methods but is usually faster in software. Estimating the centroid requires $O(N^2)$ time, where N defines the image resolution, and locating the most distant edge point from the centroid takes $O(N_E)$ time. The following PQT command sequence also draws this circle:

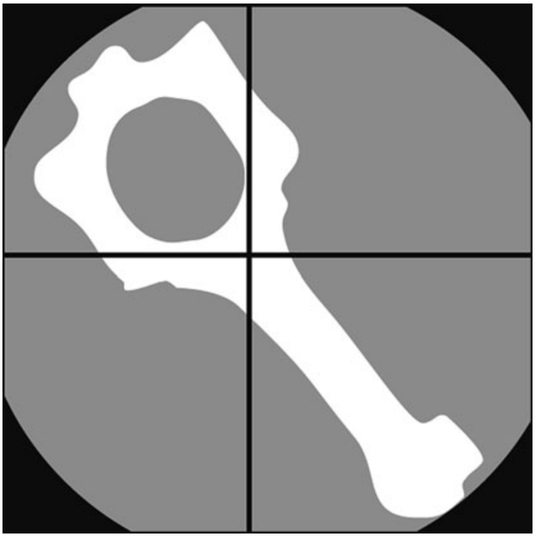
```
cbl(1),        % Check that there is only one blob
[X,Y] = cgr    % Centroid is at [X,Y]
```

```
hic(X,Y),       % Draw intensity cone
mni,            % Mask cone by the input blob
Z = gli         % Find intensity at the furthest point from centroid
swi,            % Switch current and alternate images
thr(Z),         % Threshold the cone – produces a solid white disc
```

17.4.2.3 Circle of Equivalent Area

Our next method for fitting a circle is to find the area (A) of the given blob. Then, take the radius to be √(A/π). Again, the centre of the circle is set to coincide with the centroid of the blob. Of course, this circle does not contain all of the points within the blob unless it happens to be a circle. This procedure is particularly useful for display for a human operator, since both size (area) and position are preserved (➔ *Fig. 17.46*). Here is the QT program for generating this circle.

```
[X,Y] = cgr,            % Centroid is at [X,Y]
A = cwp,                % Count white points
B = int(sqrt(A/pi)), X1 = X-B, Y1 = Y-B, X2 = X+B, Y2 = Y+B,
zer,                    % Black image
cir(X1,Y1,X2,           % Draw disc – specify corners of enclosing rectangle
Y2,255),
```



■ **Fig. 17.45**

The cross-lines indicate the centroid and the circle is tangential to the edge at its farthest point from the centroid

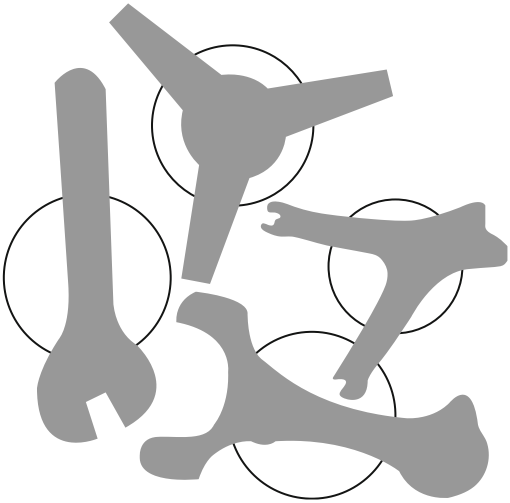


Fig. 17.46

Circles drawn by the QT command *dab*('*dec*'). Each circle has the same area as the shape it represents

17.4.2.4 Fitting a Circle to a Nearly Circular Object

The next method for generating a circle to represent a given blob is most appropriately used when that blob is itself nearly circular, or it contains a well-defined region where the edge is a circular arc. It is particularly useful for estimating the radius and centre of curvature and is used, for example in Chap. 31 to compute the radius of curvature of the shoulder of a glass vial. Given three edge points, $[A_x,A_y]$, $[B_x,B_y]$ and $[C_x,C_y]$, we can use the following formulae to estimate the position of the centre $[P_x,P_y]$ and radius, R, of that circle which intersects all three points:

$$\begin{aligned}
D &= 2\left(A_yC_x + B_yA_x - B_yC_x - A_yB_x - C_yA_x + C_yB_x\right)\\
P_x &= \left(B_yA_x{}^2 - C_yA_x{}^2 - A_yB_y{}^2 + A_yC_y{}^2 + C_y{*}B_x{}^2 + B_yA_y{}^2 + A_yC_x{}^2\right.\\
&\quad \left. - B_yC_y{}^2 - B_yC_x{}^2 - A_yB_x{}^2 + C_yB_y{}^2 - C_yA_y{}^2\right)/D\\
P_y &= \left(C_xA_x{}^2 + C_xA_y{}^2 + A_xB_x{}^2 - C_xB_x{}^2 + A_xB_y{}^2 - C_xB_y{}^2 - B_xA_x{}^2\right.\\
&\quad \left. - BxA_y{}^2 - A_xC_x{}^2 + B_xC_x{}^2 - A_xC_y{}^2 + B_xC_y{}^2\right)/D\\
R &= \sqrt{\left(\left(A_x - P_x\right)^2 + \left(A_y - P_y\right)^2\right)}.
\end{aligned}$$

(This is the basis of the QT command *fcd.*) To estimate the radius of curvature of a supposedly circular arc, three well-spaced edge points are chosen. The position coordinates (Px, Py) and radius (R) are then calculated. Notice that, if $[A_x,A_y]$, $[B_x,B_y]$ and $[C_x,C_y]$ lie too close together, wild fluctuations in the estimated values for Px, Py and R can occur in response to quantisation effects and camera noise. In order to obtain an accurate fit, we can perform this process repeatedly, as $[A_x,A_y]$, $[B_x,B_y]$ and $[C_x,C_y]$ are moved progressively around the arc. At each position, we place a spot at the centre of the estimated circle. Gradually, as more and more edge points are analysed, a pattern emerges, often in the form of several bright fuzzy spots, each one indicating the centre of curvature of a separate circular arc (➔ *Fig. 17.47*). This provides a useful method for detecting circular arcs arising from partially occluded circular discs. The other methods described here are unable to perform this function. If the arc being analysed is a nearly circular ellipse, or in the shape of an egg or rugby-ball, the set of points plotted in this way may trace a curve, rather than form a compact cluster (➔ *Fig. 17.48*).

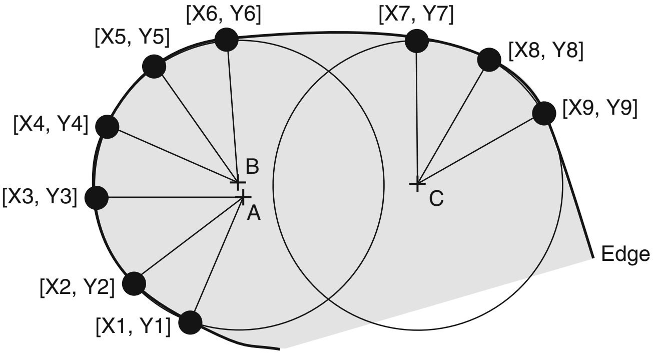


Fig. 17.47
Three edge points, such as [X1,Y1], [X2,Y2], [X3,Y3], can be used to fit a circle (centre A) to a small portion of the edge. This process can be repeated many times to yield a succession of circles (centres B and C). Notice the tendency of the centres of these circles to cluster together (points A and B), if the edge contains a long circular arc. This clustering is evident in ➔ *Fig. 17.48*

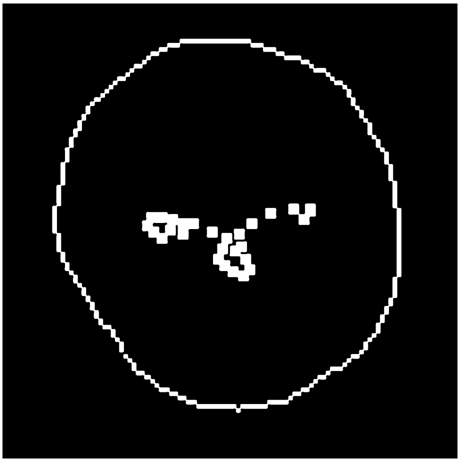


Fig. 17.48
The centres of the circles fitted to the edge of a satsuma (a member of the citrus family of fruit) form a diffuse cluster, since the edge is not a perfect circle

17.4.2.5 Polygonal Representations of a Blob

Several algorithms for calculating the convex hull of a blob-like figure have already been described in ➲ Sect. 17.3.6. In contrast to this, our objective here is to demonstrate that, for many purposes, there exist perfectly acceptable alternatives (heuristics) that are likely to be easier to calculate but which generate approximations to the convex hull. It is possible, for example, to define a tree-like structure (akin to a concavity tree) to represent a shape in terms of its minimum bounding circle [*mbc*] rather than its convex hull. Sometimes the approximation to the convex hull can be quite crude. Apart from rectangles, we might choose to use triangles, pentagons, hexagons, octagons, hexadecagons or one of the circles defined earlier in this section. In many application, these are all valid approximations. We need to adopt a pragmatic attitude, so that any technique that produces acceptable results at a tolerable computational cost can be used. Whether or not a given approximation will suffice depends, of course, on the application.

17.4.2.6 Minimum-Area Rectangle

One of the simplest of these approximate methods uses the minimum-area rectangle (QT operator *mar*, ➲ *Fig. 17.49*). Notice that the sides of this rectangle are parallel with those of the image. The minimum and maximum X- and Y-limits of a blob can be found very easily, whether we base our calculations on an array, the run-code, chain code or the polar vector representation. The procedure for finding the maximum/minimum of a sequence of numbers is straightforward, and we will not dwell further on this point.

17.4.2.7 Minimum-Area Octagon and Higher-Order Polygons

A minimum-area octagon whose sides are all inclined at an integer multiple of 45° relative to the sides of the image can also be generated easily. One way to derive this figure is explained in ➲ *Fig. 17.50* and relies on the use of various intensity wedges. (These have intensity contours (isophotes) inclined at different angles.) The wedge images can be stored, rather than recomputed for each new image that is to be processed. This ploy saves computation time

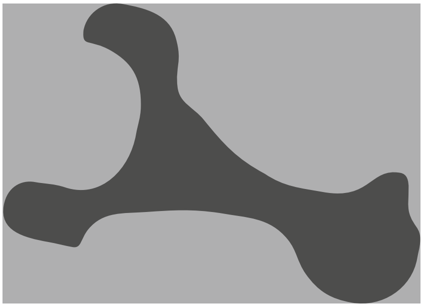


Fig. 17.49
The minimum-area rectangle whose sides are parallel to the image border

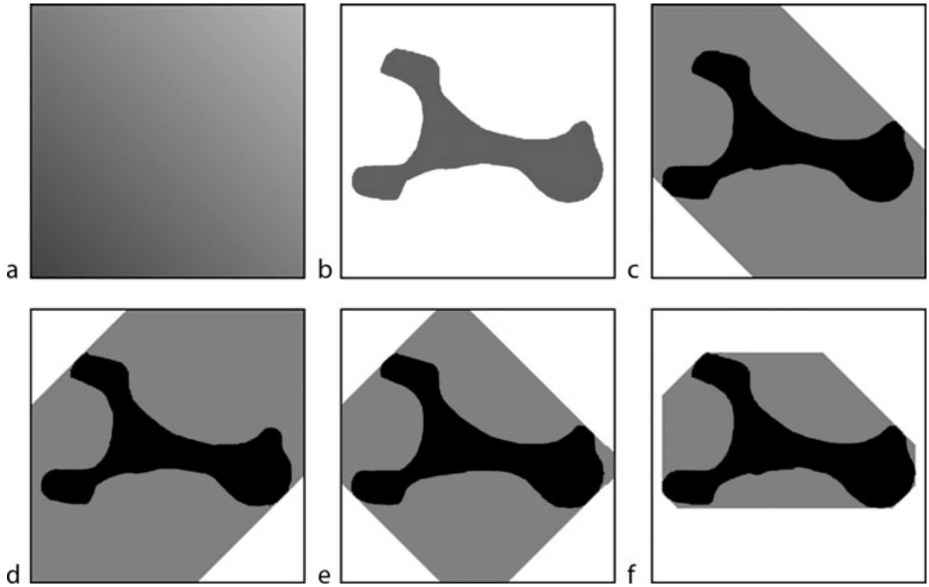


Fig. 17.50

Generating a minimal octagon to fit a given blob (i.e., automobile connecting rod). (a) Intensity "wedge." (b) Input shape applied as a mask to intensity wedge. (Grey-scale range expanded for clarity.) (c) Maximum and minimum intensities found in (b) are used as parameters for thresholding (a). (d) As in (c) but using a different wedge. (e) The stripes in (c) and (d) are combined [QT: *mni*]. (f) The octagon was generated in the same way as (e) but *four* stripe images were combined

but may require a little more hardware, i.e., extra memory. (Digital storage is so inexpensive nowadays that this is a small price to pay for increasing the processing speed.)

The minimum-area octagon is a closer approximation to the convex hull than is the minimum-area rectangle, simply because the former has twice as many sides. A multi-sided polygon can make the approximation even more accurate and can be generated by a simple extension of the same procedure. For example, we might choose to approximate the convex hull by a hexadecagon (16 sides), in which case, we generate and store eight wedge images (➲ *Fig. 17.51a*). However, there is no reason why we should restrict ourselves to generating a *regular* polygon. Using a simple QT program, we can generate a polygon with as many sides, inclined at any angles, as we wish. The program details need not concern us here.

17.4.2.8 Minimum-Area Rectangle

It is quite easy to construct a minimum-area rectangle whose sides are inclined at some arbitrary angle to the image axes. This can be achieved simply by rotating the input blob, then constructing the minimum-area rectangle as before, and finally reversing the rotation. As an illustration of this, the following QT program derives the smallest rectangle whose sides are parallel/normal to the principal axis derived from the original image (➲ *Fig. 17.51b*).

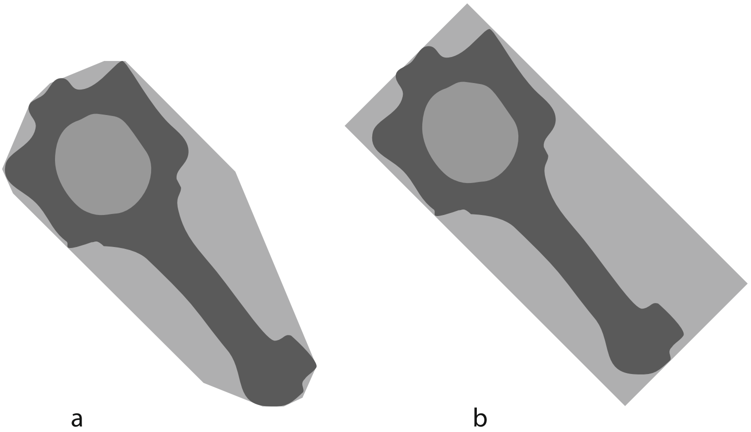


Fig. 17.51
(a) Minimal area hexadecagon whose sides are oriented at angles of 0°, 22.5°, 45°, . . ., 337.5°.
(b) The minimum area rectangle whose orientation is determined by the principal axis

```
[X,Y,A] = lmi               % Centroid (X,Y) & orientation of principal axis
[Xw1,Yw1,Xw2,Yw2] = dgw,    % Image size
Xw = (Xw2−Xw1)//2,          % Width of the image
Yw = (Yw2−Yw1)//2,          % Height of the image
X1 = Xw−X,                  % How much to shift image in X direction
Y1 = Yw−Y,                  % How much to shift image in Y direction
X2 = −X1,                   % How much to perform reverse shift (X direction)
Y2 = −Y1,                   % How much to perform reverse shift (Y direction)
psh(X1,Y1),                 % Shift so blob is at centre of the image
B = −A,                     % Negative of angle of orientation
tur(A),                     % Normalise orientation
mar,                        % Minimum area rectangle (hollow)
blb,                        % Fill the minimum area rectangle
tur(B),                     % Reverse rotation
psh(X2,Y2),                 % Reverse shift
```

Image rotation is not normally regarded as being very satisfactory because it frequently creates ragged edges. Notice, however, that the program above always rotates a rectangle, which makes the effect less pronounced than it would be if we rotated the input blob a second time. The second image rotation can be avoided altogether, since we need only compute the coordinates of the *corners* of the rectangle generated by *mar*. We then calculate their new positions after rotation and construct a (hollow) rectangle, simply by drawing its sides as four digital straight lines.

The astute reader will have noticed by now that we do not need to rotate the input image either, since we could derive the orientation of the axis of minimum second moment (using *lmi*), then trace the blob edge to find the extreme points along that axis and normal to it. There are many ways to implement nearly every algorithm!

17.4.2.9 Remarks

In this section, we have discussed a variety of techniques for generating figures such as rectangles, polygons and circles to represent a given shape. By the word "represent" we mean that we replace the given shape by another which in some way simplifies the subsequent calculation or display of the image. By choosing the most appropriate data-extraction procedure, we can remove unwanted information while retaining other data that is able to summarise some significant property of the given shape. We may wish to preserve position and orientation, but ignore the complexity of a meandering edge contour. On the other hand, we may wish to compute the "density" of a shape by comparing its area to that of its convex hull, or the minimum bounding circle. There are numerous options; the biggest mistake of all is to believe that just one of them (e.g., the convex hull) is inherently "better" than all of the others.

17.4.3 Determining Object/Feature Orientation

Perusal of the Computer Vision literature quickly reveals that the Hough transform is a very popular method for fitting a straight line to a set of disconnected colinear spots in a digital image. On the other hand, it has not gained anything like the same level of support with Machine Vision system designers. In this section, we will discuss why this has happened and we will look at alternatives to the Hough transform (➋ Chap. 18) that can perform a similar function.

First, let us explain what the Hough transform (HT) does. Given a digital image containing a set of disconnected spots, the HT can detect a subset of those spots that lie along a straight line (➋ *Fig. 17.52*). That line can be at any orientation and be in any position. The HT can also be used to join together fragments of a "broken" straight line that might arise, for example, from applying an edge detector (e.g., *sed*) and thresholding to a low-contrast image. The fact that the HT can detect sets of collinear spots aligned at any orientation is both its strength and its weakness. Obviously, it is necessary sometimes to work with images whose structure is completely unpredictable. This is the domain of Computer Vision research. On the other hand, Machine Vision system designers nearly always know, in general terms, what format the images will have. This knowledge means that very general procedures such as the HT are often not needed, since less expensive techniques will suffice and may give better results, since they are more specific.

Consider the task of determining the orientation of an object such as a machine component. In many cases, we can use the outer edge to determine the object orientation. We might instead use holes, 3-D shape or vividly coloured regions, etc., to perform the same function. We might sometimes be forced to use "internal" edges but we can, in some instances, make them more pronounced by using specialised lighting techniques. We must always be careful to distinguish between the primary task (i.e. finding the orientation of the object) and the secondary task (e.g., finding the orientation of a given edge in an image). Consider the image shown in ➋ *Fig. 17.53*. This has several "strong" features, which create a high local contrast (e.g., holes, outer edge) and "weaker" ones, principally the "internal" edges. In order to determine the orientation and position of this particular object prior to directing a robot to pick it up, we would be foolish to rely on the weak features, whose structure is, in any case, complicated compared to that of some of the stronger ones. As a general principle we should,

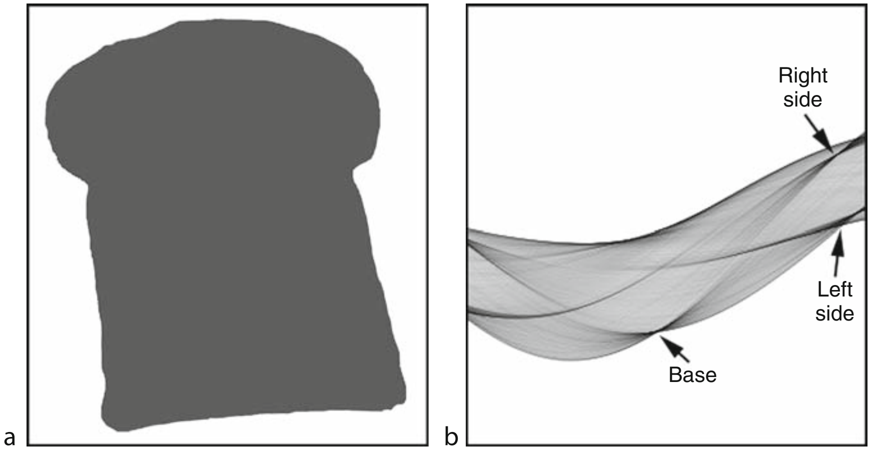


Fig. 17.52
Hough transform applied to the silhouette of a slice of bread. (a) Original image. (b) [*QT: bed, huf, neg, 2•sqr*] applied to (a). (The sub-sequence [*neg, 2•sqr*] simply improves the contrast for human viewing.) Notice the three dark points, which indicate the base and sides of the loaf, and that the sides are nearly parallel (i.e., nearly same position along the horizontal axis)

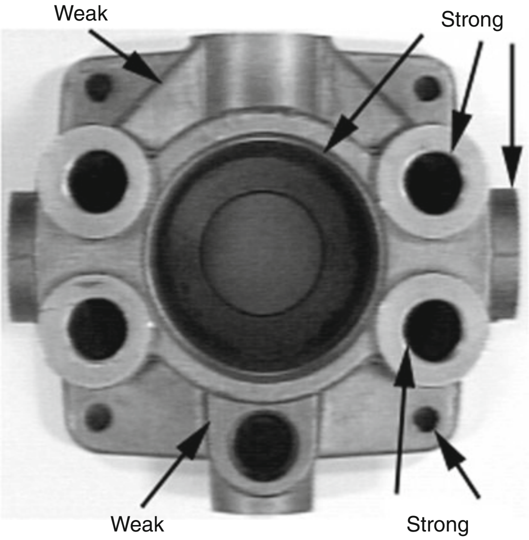


Fig. 17.53
The Hough transform could, in theory, be used to determine the orientation of this object (an hydraulics manifold) by using it to detect one of the four linear features, two of which are labelled ''weak.'' However, this would not provide as reliable a result as a technique based on higher contrast features, such as the holes and the outer edge contour (labelled ''strong'')

wherever possible, extract and measure features that are both robust and easy to analyse. In most case, internal edges are often not so satisfactory in one or both of these respects.

Among the alternative procedures for determining object orientation that are open to us are these:

- Principal axis (axis of minimum second moment) (PQT command *lmi*).
- The line joining the centroid to the edge point that is farthest from the centroid.
- The line joining the centroid of the object silhouette to that of the largest bay.
- The line joining the centroid of the object silhouette to that of the largest lake.
- The line joining the centroid of the largest bay to that of the second largest bay.
- The line joining the centroid of the largest lake to that of the second largest lake.
- The line joining the centroid of the largest bay to the centroid of the largest lake.

Clearly, there are many other options (Appendix F). To be accepted for use in a vision system, the designer must believe that the HT is demonstrably better than all of these competitors. Moreover, to avoid using a computationally expensive technique such as the HT, the designer has many optical tricks at his disposal to make the task simpler for the image processor.

Now, let us consider analysing sheet material made in the form of a web that is formed, treated, coated and printed using rollers. All predictable features, such as printed patterns, are precisely aligned to the edge of the web, assuming that it is well made. Faults on the web are, by definition, not predictable and could, in theory, occur in any orientation. However, there are two points to note:

1. We do not normally need to know the orientation of a single blob-like fault, such as a spot caused by localised chemical contamination, material fault, coating blemish or a squashed insect.
2. Certain faults which are caused by the manufacturing process are always aligned along the up-down axis of the web. For example, a very common type of fault is caused by a piece of material being dragged along the surface of the web, leading to a scratch in the direction of the web motion. Another commonly occurring fault is manifest as a point-like fault repeated periodically. It is not difficult to see that this is caused by a foreign body adhering to one of the rollers used to manufacture the web.

In neither case do we need to measure the orientation of anything on the web, except perhaps along accurately defined axes. We will return to this later, since it enables far less computational work to be used. It is a common requirement for web products to have well defined surface-texture properties but, once again, these can almost always be related to the web axes.

These arguments suggest that the HT is not very useful for machine vision, where verification rather than recognition is required. There are, of course, many other application tasks, such as robotic trimming of vegetables, decorating cakes, guiding a sewing machine, etc., that we should consider to make our argument comprehensive in its scope. However, despite their many years of experience in this field, the author has never felt the temptation to resort to using the Hough transform, because other techniques have proved equally or more effective and certainly more attractive in terms of the computational effort they require. To be fair in this discussion, we should point out that certain authors do use the HT extensively and have demonstrated good results (see ➲ Chap. 18) [6]. While not criticising our colleague, we simply point out that there are many instances where other techniques can achieve the same primary purpose as the HT without incurring its high computational cost. For this reason, we will discuss other procedures below:

1. Radon transform
2. “Island hopping”
3. Polar-to-Cartesian coordinate-axis transformation

17.4.3.1 Radon Transform

The Radon transform (RT) is effectively a reformulation of the HT but does lend itself to greater versatility, as we will see later. To compute the Radon transform of a given binary image, we consider a thin strip inclined at an angle ϕ to the coordinate axes and located at a distance d from the centre of the image (➋ *Fig. 17.54*). This strip will be denoted by S(ϕ,d). Its precise width need not concern us for the moment. Let us count the number of white pixels in S(ϕ,d) and denote this number by μ(ϕ,d). Then, we compute μ(ϕ,d) for all possible values of ϕ and d. As we do so, we construct an image whose coordinate axes are ϕ and d, and whose intensity is given by μ(ϕ,d). This is the Radon transform of the input image. The Radon transform should, of course, be defined in formal mathematical terms as an integral involving two continuous variables, ϕ and d. In fact, the discrete nature of a digital image means that we can compute only an approximation to the Radon transform.

One of the most obvious ways to implement the Radon transform is to rotate the image by an amount ϕ and then integrate the intensities along the picture rows. (The QT sequence [*tur(A), rin*] expresses the rotation-integration operation succinctly.) The right-most column of the resulting image will exhibit a strong peak, if we happen to have rotated the input image so that a set of collinear points is aligned horizontally (see ➋ *Fig. 17.55*). We do not of course,

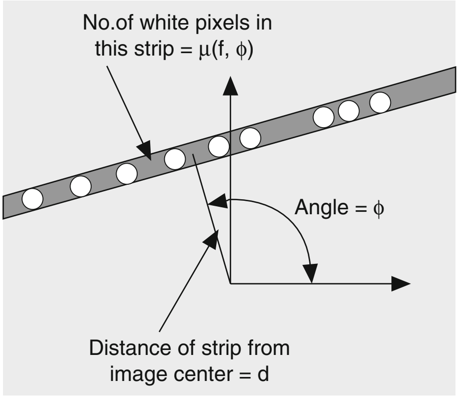


Fig. 17.54
The intensity is integrated along the dark grey strip S(ϕ,d). (In a binary image, we simply count white pixels within S(ϕ,d).) This yields the value μ(ϕ,d)

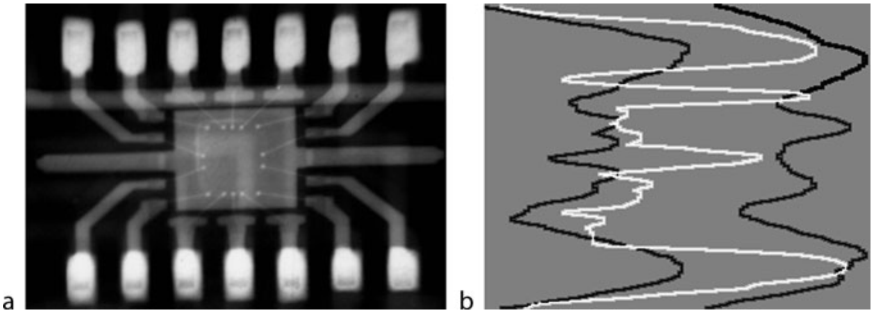


Fig. 17.55
The Radon transform detects linear features in grey-scale images. (a) Original image: microfocal x-ray of a small integrated circuit. (b) Three intensity profiles, obtained after applying [*tur(Angle), rin*]. (White curve, Angle = 0°. Black curves, Angle = ±10°.) Notice the strong narrow peaks that are visible only in the white curve. These identify the linear features just below the upper-most row of pads (i.e., very bright rectangles) and across the centre of the image

always need to construct the whole RT image. For example, we may know that the spots are likely to lie along a line whose orientation is within tightly defined limits. There are several special cases which make this ploy effective:

1. The horizontal and/or vertical axes of the input image may have some special significance in physical space. For example, the horizontal axis of the image may represent the vertical (Z) axis in [X,Y,Z] space, or the vertical axis of the image may correspond to the direction of travel of a moving web or conveyor belt. In the latter case, the photo-detector array on a line-scan sensor would correspond to the horizontal axis of the image.
2. We can obtain a crude measure of orientation quickly and easily from some other features before applying a more refined procedure based on the RT.
3. We know beforehand what angles are critical. (For example, we may have learned that diagonal streaks are indicators of some undesirable "fault" condition, whereas, horizontal ones are benign.)

The point to note is that in many instances we do not have to compute and analyse the full Radon Transform image.

The QT operator sequence [*tur(A), rin*] may be used as the basis for simply detecting the presence of a collection of "colinear" spots, without actually constructing the Radon transform image. This leads us to a simplified version of the Radon transform which is perfectly adequate for many purposes. Of course, we need to repeat the procedure many times in order to detect a set of spots, that can be aligned at any angle.

In some cases, we might usefully split the analysis in two stages: a low-resolution initial RT scan, leading to and a second, more refined process based on the first. This can be expressed in QT, but again the program cannot be included here because we have not yet described all of QT's relevant features in sufficient detail.

17.4.3.2 "Island Hopping"

In order to appreciate how straight lines can be reconstructed from a set of disconnected spots by the so-called "Island Hopping" procedure, we need to describe another idea first. We will therefore describe the *fan* operator. Consider two points [X1, Y1] and [X2, Y2]. The *fan* operator searches a wedge or fan-shaped region of the image for a white pixel (see *Fig. 17.56*). The search begins at the point [X2, Y2] and is concentrated within a circular segment of angular width defined by the parameter A. The fan-shaped search area is centred on the line defined by [X1, Y1] and [X2, Y2]. The address of the first white pixel found defines two variables X and Y. It is easiest to think of the fan program being applied to an image consisting of single isolated white pixels. Then, given two starting points, a third point that is approximately aligned with them is found. A fourth point can be found by applying the fan operator recursively. The mental picture that inspired this procedure, which the author called "island hopping" was that of a frog jumping from one lily pad to another. The frog has a limited range for each jump and has a narrow field of view.

Figure 17.57 shows the result of applying *island hopping* to a picture containing isolated white pixels, some of which are aligned approximately. In this example, the angular width of the fan-shaped scan area is quite large.

17.4.3.3 General Remarks

The Radon transform can be used to detect thin streaks and aligned spots in a grey-scale image, whereas the Hough transform, as it was defined in Chap. 14, cannot. Thus, the Radon transform provides not just a convenient alternative to the Hough transform but also offers interesting possibilities for analysing grey-scale images, which the HT cannot match.

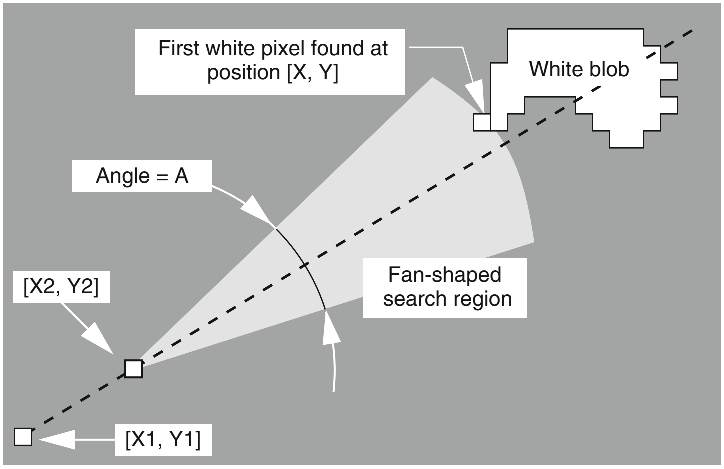


Fig. 17.56
The fan predicate used as the basis for "island hopping." The points [X1,Y1], [X2,Y2] and the angle A are specified when fan is called. The point [X,Y] and its distance from [X2,Y2] are then computed

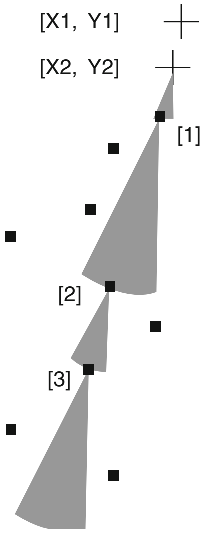


Fig. 17.57
Applying *island_hopping* to a simple test image. [X1,Y1], [X2,Y2] are the starting points. Numerals in brackets indicate the order in which the other points lying along the same line are detected

On the other hand, the Hough transform can be modified to detect other curves of known form. For example, it can be used to detect circles, even when they are overlapping. The generalised Hough transform [6] can even detect partially occluded shapes, such as touching/overlapping automobile connecting rods. While the Hough transform is able to accommodate broken contours, this property is shared with the Radon transform too. Although this chapter is not concerned with implementation hardware, it is interesting to note that a cylindrical lens and a line-scan camera provide a very convenient means of performing the calculations necessary to implement the RT.

17.5 Additional Remarks

We are near the end of this chapter, which has touched on a wide variety of algorithms and heuristics. It is appropriate therefore to review the lessons learned. The discussion above does not constitute a comprehensive review of these computational techniques; it is merely a discourse on the folly of taking too a narrow view of which algorithm to use. In the following chapters, we should bear in mind that virtually no algorithm is so important as to be regarded as “essential.” Of course, it would be difficult to manage without using operators such as *threshold* [*thr*] and *negate* [*neg*] but bear in mind that even these can be implemented in several different ways.

17.5.1 Solving the Right Problem

Although much of this book concentrates on algorithmic/heuristic techniques suitable for use in machine vision systems, we have repeatedly emphasised that they must operate in harmony with the other parts of the system. Indeed, they should be designed/selected specifically with this in mind. Thus, we should not quibble about whether, for example, the Sobel edge detector is theoretically "better" than the Roberts operator, since they are similar in their performance. If it matters which one we use, then something else is wrong, probably the lighting-viewing sub-system (Appendix C). There are numerous other examples of this general principle. We have at our disposal many different computational tools and often have several options for each algorithmic step in a processing sequence. We must be careful, therefore, that we address the most important questions. Very often, we find that whether we choose algorithm A or B is nearly irrelevant in terms of the results we finally achieve, compared to the far great differences that exist in the speed and cost of implementing them. We have constantly asserted that Machine Vision is a specialised area within Systems Engineering and, as a result, we must optimise all parts of a system separately and ensure that they will work together harmoniously. The real challenge is to ensure good system integration, since without this no system will ever be truly successful.

We must be clear about our primary purpose in building a vision system. It is not to prove how clever we are by using sophisticated techniques. Nor is it to analyse images in a preconceived way; we must be flexible in our thinking. Consider, for example, the task of finding the orientation of a machine component. We might be tempted to employ the Hough or Radon transforms, since they are so general. When demonstrating our academic prowess becomes a factor in our thinking, the design will inevitably be a poor one. We have at our disposal many different techniques for determining object orientation and these can be based on a variety of features, such as corners, bays, holes, coloured regions (e.g., flashes, logos, etc.) We must bear in mind that our primary objective is to determine the orientation of an object, not find the orientation of a straight line in an image. We might decide that the latter forms a useful secondary goal. However, we might find instead that, by altering the image acquisition sub-system a little, we can achieve our primary goal very much more cheaply, and more reliably.

17.5.2 Democracy: No Small Subset of Operators Dominates

One thing has emerged clearly but has, so far, not been stated yet with sufficient force, namely that image processing procedures for machine vision are highly varied. A machine vision system can and, at the moment, almost certainly should be designed for a specific task. Another important lesson is that the image processing procedure needed for a real-world Machine Vision application is complex. In QT terminology, it is not a single operator. Instead, it requires a large set of operators, controlled in a sophisticated way. We cannot therefore pick on any one step and say that implementing this is crucial to solving the problem. Thus, we must never concentrate on implementing only one operator, since they are all vitally important. Hence, we must not allow ourselves to be persuaded to use a particular implementation technology simply because one of the elements in a processing sequence could make good use of it. A little while ago we mentioned the fact that the Radon transform can be implemented using a cylindrical lens and a line-scan camera. This combination is of no use whatsoever for such tasks as implementing the convex hull or Sobel edge detector, thresholding an image, or computing its histogram, etc. In the same way, a parallel processor that is appropriate for implementing a convolution operator is virtually useless for processing the chain code. A symbolic processor (e.g., Prolog) that can make inferences about canary yellow being "similar" to sulphur yellow is similarly limited when we need to perform histogram equalisation. Thus, we must be prepared for the inevitable fact that we will have to use an implementation technology that is nearly optimal on part of the algorithmic sequence and less efficient on others. This is why we felt it desirable to include in this book a chapter where we highlight the potential for reformulating the algorithms. In this way, we can circumvent at least some of the difficulties that would ensue when we fix the implementation technology.

Let us end this section with a question for the reader to ponder. Suppose that we had an implementation technology that was nearly ideal for all component parts of an algorithm except for one important element, such as measuring blobs. Our hypothetical machine might be able to perform all of the "difficult" calculations, such as convex hull, blob shading, Hough transform, "island hopping," colour recognition, etc. Would not such a machine be virtually useless in practice? We must consider the implementation of the algorithm in its entirety?

17.5.3 Lessons of This Chapter

- Every algorithm can be implemented in many different ways. (➲ Sects. 17.1.1, ➲ 17.3.5–17.3.7) Even standard arithmetic operations, such as multiplication, can be implemented in more than one way.
- We should not judge an algorithm by its accuracy alone (➲ Sect. 17.1.2) Many other factors are at least as important. These include speed, cost of implementation, compatibility with existing equipment, interfacing to people and machines, conforming with company policy, etc.
- Reformulating an algorithm may make its implementation very much easier.
- We must be careful to address the important primary questions and must not fall into the trap of believing that secondary issues are all-important (➲ Sect. 17.5.1).
- No single algorithmic/heuristic procedure should be allowed to dominate our design, since all elements within an algorithmic sequence are of equal importance (➲ Sect. 17.5.2).
- We can often use heuristics in lieu of algorithms to good effect (➲ Sect. 17.1.4).
- Changing the image representation can make certain algorithms/heuristics very much easier to implement.
- We can sometimes employ grey-scale operators on binary images to good effect (➲ Sect. 17.2.2).
- We can use images as look-up tables for operations such as image warping (➲ Sect. 17.2.3).
- We can represent many of the more useful convolution operators as separate row and column operations (➲ Sect. 17.3.1).
- We can approximate many of the more useful large-window convolution operators by iteratively applying a sequence of small-window operators (➲ Sect. 17.3.1).
- Several non-linear neighbourhood operators (both binary and grey-scale) can be implemented by separating row and column operations.
- Separated-Kernel Image Processing using finite State Machines (SKIPSM, ➲ Sect. 17.3.3 and ➲ Chap. 20) allows a wide variety of morphological and other binary and grey-scale image processing operators to be implemented (exactly) at high speed in both software and hardware.
- There is no unique algorithmic definition for certain operations that are specified in natural language (e.g., edge smoothing). Hence many operators exist.

- Histograms calculated on one image can be used as the basis for processing another image, if the images are derived in quick succession from a slowly changing scene.
- Image statistics derived using a low-resolution sampling of an image may be accurate enough for certain purposes.
- We can save a lot of processing effort by concentrating on those regions of a picture where the processing is relevant to achieving the desired result. These are called *Regions of Interest.*
- Heuristics may be faster than algorithmic methods.
- Algorithms do not produce a reliable result if we violate their conditions of use.
- A given algorithm may be theoretically attractive but it may not solve our primary application task. The latter is not image processing *per se* but making decisions about real objects in the physical world.
- Heuristics can fail but this may be insignificant in practice, if they are designed and tested properly.
- Since heuristics are rules of thumb, we can add broad general rules which are safe, even if they are not always efficient.
- A heuristic may, in fact, be algorithmic, even though we do not know it to be so, because we have not been able to derive the necessary mathematical proof.
- Heuristics may be used to replace an algorithm, or to augment it.
- Heuristics may be used to make an algorithm run faster, without necessarily making its performance any less satisfactory.
- Heuristics may be used to extend the range of application of an algorithm.
- Heuristics can often "save the day" when there is no known algorithmic alternative.
- Heuristics can often achieve useful results when the use of an algorithm is impractical.
- We can often make heuristics as accurate as we wish, simply by doing more computational work.
- Elements in a processing sequence can often be combined to produce a simpler, more direct result. For example, the QT sequence [*neg, thr(100,123)*] could be implemented in one new single-pass operator.
- Look-up tables are very effective for implementing a wide range of image processing operators and are potentially very fast, when implemented in both software and hardware.
- Techniques that are attractive for Computer Vision and other (i.e., non-industrial) application areas for Machine Vision are not necessarily of much use in our subject area. (For example, they may be too general, and therefore unable to exploit application constraints.)
- Heuristics may be very different in concept from the algorithms they displace.
- We must judge heuristics objectively in the light of statistically rigourous testing.

References

1. Batchelor BG (1974) Practical approach to pattern classification. Plenum, London/New York. ISBN 0-306-30796-0
2. Batchelor BG (1980) Two methods for finding convex hulls of planar figures. J Cybernetics & Systems 11:105–113
3. Batchelor BG, Marlow BK (1981) Converting run code to chain code. Cybern Syst 12:237–246
4. Batchelor BG (1982) A laboratory-based approach for designing automated inspection systems. In: Proceedings international workshop on industrial applications of machine vision, research triangle, NC. May 1982, pub. IEEE Comput Soc, pp 80–86
5. Batchelor BG, Whelan PF (1997) Intelligent vision systems for industry. Springer, London/Berlin. ISBN 3-540-19969-1

6. Davies ER (2000) Image processing for the food industry. World Scientific, Singapore. ISBN 981-02-4022-8
7. Marlow BK, Batchelor BG (1980) Improving the speed of convex hull calculations. Electronics Lett 16(9):19–321
8. McCollum AJ, Bowman CC, Daniels PA, Batchelor BG (1988) A histogram modification unit for real-time image enhancement. Comput Graphics Image Process 42:387–398
9. Machine Vision Proverbs, opinion and folk-lore. http://www.eeng.dcu.ie/~whelanp/proverbs/proverbs.html
10. Rutovitz D et al. (1978) Pattern recognition procedures in the cytogenetic laboratory. In: Pattern recognition: ideas in practice. Batchelor BG (ed), Plenum, London/New York. ISBN 0-306-31020-1, pp 303–329
11. Simpson S (2000) For the bees: glowing paint may highlight the forces that make insects fly. Scientific American

18 Object Location Using the HOUGH Transform

E. Roy Davies
Royal Holloway, University of London, Egham, Surrey, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_18,

Abstract: This chapter considers how best to locate 2D object shapes in digital images. Simple paradigms such as boundary tracking followed by analysis of centroidal shape profiles lack robustness and are incapable of coping with occlusion, breakage or gross shape distortions, whereas the Hough transform readily overcomes these problems as an integral property of its voting schema. This is an important advantage, especially in practical situations such as those where lighting is non-ideal. In addition, the Hough transform is excellent at locating straight lines, circles, ellipses and other well defined shapes: it can even locate arbitrary shapes that are defined by so-called *R*-tables. The method also has the advantage of being able to make sense of patterns of point features, and is far less computation intensive than the maximal clique graph matching method often used for this purpose. Being a form of spatial matched filter, the Hough transform has high signal-to-noise sensitivity, and for simple shapes is highly efficient, but when applied to arbitrary shapes of variable orientation, size and individual variability, measures have to be taken to limit its computation. Overall, the Hough transform provides a very useful, robust search technique for locating known types of object in digital images.

18.1 Introduction

This handbook has already examined basic methods of image processing and has shown how objects can be located in 2D images using techniques such as thresholding and morphology. Another elementary procedure is boundary tracking: this normally leads to shape analysis via 1D representations such as the well-known centroidal profile, which is a polar (r, θ) plot of the object boundary [1]. Unfortunately, this approach is limited when the shape is at all complex because then multiple values of r occur for one or more ranges of values of θ. While this problem can be overcome – for example, by taking the maximum value of r for any θ – the method has very poor robustness. Thus serious problems arise if the boundary is broken, if part of the boundary is occluded, or even if two objects are merely touching. This demonstrates that more advanced methods are needed for recognising objects in practical situations when the images are non-ideal or when "nasty realities" occur. Bearing in mind that robust solutions are mandatory in factory environments, that lighting can never be perfect, and that we have to deal with real objects which may have subtle shapes and profiles, there is a definite need for more sophisticated methods. At the same time, rapid search through large images is often required, since factory applications have to work in real time – a very demanding scenario. Thus any method that is to be adopted has to fulfil a number of important conditions.

In this chapter we shall find that inference methods provide a way forward, and that the Hough transform is perhaps the most important example of such methods. Towards the end of the chapter we shall describe other approaches such as graph matching, and demonstrate certain equivalences to the Hough transform technique. However, to start the discussion, we shall return to fundamentals and explore the intrinsic value of template matching, which is probably the most obvious solution to the matching problem but yet is subject to horrendous computational difficulties that have to be solved in one way or another.

18.2 Template Matching: The Need for Inference

Template matching is probably the most obvious method that can be used for object recognition. It involves systematically moving a template of the object to be recognised over the image – typically in a raster scan – and recording the positions of any matches. In this way it is intended that all objects of the selected type be both recognised and accurately located. The first problem that arises with this method is its unreliability. Because of the variability of both the original objects and their images, exact matches cannot be expected, so some relaxation of the recognition conditions is vital. Variability takes a great many forms:

1. The objects may have slightly different sizes and shapes, and in general 3D sizes and shapes will have to be considered.
2. They may have surfaces that have varying roughness, brightness and colour.
3. They may have gross defects in size, shape, roughness, brightness and colour.
4. They may appear at different distances and orientations in both 2D and 3D.
5. They may be viewed in different lighting conditions, and the lighting conditions may vary.
6. There will be digital quantisation effects, due to the particular spatial, grey-scale or colour resolution that is used.
7. There may be added digital noise.
8. There will in general be lens distortion, though this may largely be corrected.
9. There will be camera calibration problems, though these may be solved with fair accuracy.

Overall, various degrees of normalisation against camera and viewpoint variation can be applied, and with care the remaining errors will be small. Resolution can be made adequate and digital noise can be suppressed. Thus, in general, imaging problems can largely be overcome. However, the intrinsic variability of objects is a problem. Varying surface roughness – or texture, as it appears in the images – has to be tackled statistically rather than logically, since it contains a large random component. This leaves the large variations in shape of many items such as cereal grains and vegetables, not to mention articulated objects including items as different as insects and pliers. It is also the case that many small objects cannot be lit uniformly, and therefore acquire shadows, while their images often have a degree of fuzziness, which amounts to a perception of additional variability or ambiguity in shape. In general, real or perceived variability is a major problem, and much of it is due ultimately to the quality and maintainability of the illumination field.

It is useful to consider these variations as degrees of freedom of the images that are obtained from the camera. In any 2D image we have two degrees of freedom for position: this means that for a 256 × 256 pixel image and a 32 × 32 pixel object, some 32 million operations are required to search the entire image. However, the 2D orientation of any object will normally be unknown, and this extra degree of freedom will boost the number of operations to over 10,000 million. Variable size adds another degree of freedom, so the number of operations may well exceed 10^{12}. However, this is far from the end of the story, since variable shape adds many additional variables, as does variable surface texture, variable intensity and variable colour. While each of these additional variables may have only restricted range, their number (as in the case of shape) is enormous. This leads to so many potential operations for any rigorous template matching operation that a complete rethink of the position is called for. Essentially, the problem is that the whole template matching approach is one of *deductive* recognition, and this embodies a serious flaw. The solution is to *infer* rather than to deduce the presence of objects.

The crucial step to achieving this is to note that very small objects or features have very few degrees of freedom, so the computational penalty for locating them is tolerable. For example, finding edge points in a 256 × 256 pixel image can involve as few as 1 million operations, if say a Sobel operator is used: such operations need only take a few milliseconds on a PC and would not require the use of any special hardware accelerator chips. The problem is that once all the edge points in an image have been found, integration of the knowledge is non-trivial, and could itself require considerable computation. It should now be noted that such knowledge is incomplete, so it is no longer deducible, with certainty, what objects may appear in any image: deduction thus has to give way to inference; i.e. we have to infer the presence of the objects from their features.

Inference requires special techniques, and it ought to be said that these techniques are not as infallible as deduction: they can go wrong and can lead to erroneous hypotheses. Thus, in principle, the hypotheses need to be checked before they can be used in real applications. We say "in principle" because there are situations where checking may be superfluous. For example, on a widget line we only expect widgets, and on a biscuit line we only expect biscuits: we don't need to check or even test for biscuits on a widget line. In general, there may indeed be an overwhelming probability that any hypothesis that is made will actually be correct – in which case additional tests will not be called for. On the other hand, while chocolate biscuits may be the only objects on a chocolate biscuit line, biscuits that touch during the chocolate coating process are liable to end up stuck together. This, then, is a condition that does need to be checked for. Overall, it is the balance of probabilities that has to be borne in mind when considering any checks that have to be made. Note that, in general, objects will be checked and inspected immediately after detection, so false alarms are not overly harmful except to the extent that processing speed may be suboptimal. In what follows, we shall refer to the hypothesis checking process from time to time, as relevant in particular cases.

Finally, another important factor is that inference is not simply defective in its need for post-checking: relinquishing pure deduction actually leads to cognitive advantages in that the presence of partially occluded or defective objects is automatically allowed for. What has to be resolved at the checking stage is not only whether the object is in the hypothesised class but also whether it is a perfect example of that class or otherwise what deficiencies it may have.

Before proceeding further on the methodology of inference, we consider the types of feature that can profitably be used to trigger object detection.

18.3 Relevant Features

As pointed out above, the reason for finding objects from their features ultimately devolves into the much smaller size of the latter. In fact, one of the effects of small size is a limit on the variability and indeed on the number of types of feature it is possible or desirable to use. Perhaps the most obvious types of feature are edge points, which can be detected using Prewitt or Sobel edge detection masks of size 3 × 3 pixels [1]. Nevertheless there are three other types of feature that can be recognised using masks of this size: line segments, small holes and corners. Typical masks for the four respective features are:

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} -4 & -4 & -4 \\ -4 & 5 & 5 \\ -4 & 5 & 5 \end{bmatrix}$$

However, it turns out that in different practical applications, somewhat larger masks may be preferable. For example, for line segment detection it may be necessary to match wider line segments, so 5 × 5 or 7 × 7 masks may be needed. A similar situation may arise if larger holes have to be located. Finally, rounded corners may necessitate use of larger masks. It will also be necessary to use a number of masks of different orientations, and the number of masks will tend to increase with size. Subtleties of shape, as with sharpness of the corners, may also be more important with larger mask sizes. Here we shall not be prescriptive, but merely note that the numbers, sizes and coefficients of the masks will need to be adapted to the particular cases and applications. Our specific concern in this chapter is how to collate the information emerging from the feature detectors.

One final point is that edge and line segment detectors will be intrinsically capable of providing estimates of direction as well as information on feature location. This additional information can be of very great value for improving not only the accuracy of whole-object location but also the efficiency and speed with which it can be achieved. This point will be emphasised a number of times below. Suffice it to say here that once the *x*- and *y*-components (g_x, g_y) of intensity gradient have been ascertained, the intensity gradient vector $\mathbf{g} = ge^{i\theta}$ may be determined from the vector-based equations:

$$g = \left(g_x{}^2 + g_y{}^2\right)^{1/2} \tag{18.1}$$

$$\theta = \arctan\left(g_y/g_x\right) \tag{18.2}$$

Similar formulae have been shown to apply for suitably designed line segment masks [1].

18.4 The Hough Transform Approach

As indicated above, the Hough transform [2] provides a suitable strategy for inference. It relies on the representation of objects of a given type by a set of *p* parameters: thus, any particular object of this type which appears at a particular position with a particular orientation is allocated a certain location in a *p*-dimensional parameter space spanned by the *p* parameters. It is not necessary for the *p* parameters to be the *x* and *y* coordinates of the centre of mass of the object or its orientation *θ* relative to a known axis in it, though that is possible. However, there must be a one-to-one relation (In fact, we shall see below that it need not be as restricted as one-to-one, though this is the simplest scenario to envisage at this stage.) between the position and orientation parameters and the parameter space coordinates. The crucial point is that we must be able to transfer feature points from image space to the parameter space, to form the Hough transform, and transfer data back from the parameter space to image space, to show where particular objects have been found in the image.

The other important aspect of the Hough transform is that the feature points vote in the parameter space for successful parameter space combinations, so that by the time all the feature points have been considered, the points in parameter space which have the most votes can be taken to represent the most likely sets of parameters for objects appearing in the original image: at that stage the objects have essentially been located – though as indicated earlier, their existence is in the nature of a hypothesis until further verified by direct matching in the image.

The importance of all these details will become clearer in the following sections, where we will show how the Hough transform can be applied to recognise and locate straight lines, circles, ellipses and more general shapes.

18.5 Line Detection

In this section we show how the Hough transform may be used to locate straight lines in digital images. Following the statements made in the previous section, it is first necessary to decide upon a parametrisation for the straight line. The most obvious one corresponds to the equation

$$y = mx + c \tag{18.3}$$

The two relevant parameters, which have particular values for a particular line, being m and c. Accordingly we construct a parameter space with coordinates (m, c). Once an edge point (x, y) is found in an image, all possible values of m and c that lie on the dual line:

$$c = y - xm \tag{18.4}$$

are taken and accumulated as votes in the parameter space. In this case the set of votes will itself form a line in the parameter space. Any other edge point in the original image will give rise to another line in the parameter space. Clearly, after many edge points have been dealt with, many lines of votes will be found to cross at a particular point in parameter space, and this point is essentially a dual of the original line in image space. Hence the parameter space crossing point – which is the highest peak of votes in that space – can be projected back to identify the relevant line in the original image.

While the procedure outlined above is adequate in principle, it is actually clumsy and computation intensive in practice. First, it cannot cope well when the line is nearly parallel to the y-axis, as then the gradient m will be very large or even infinite. This means that this particular parametrisation is inconvenient: it is usually replaced by the normal co-ordinate (ρ, θ) parametrisation of the line (➲ *Fig. 18.1*). The normal co-ordinates can be calculated by replacing m and c by the following expressions. (The significance of ➲ Eq. 18.5 is that the gradient of the line is perpendicular to the edge normal.)

$$m = -\cot\theta \tag{18.5}$$

$$c = \rho \operatorname{cosec}\theta \tag{18.6}$$

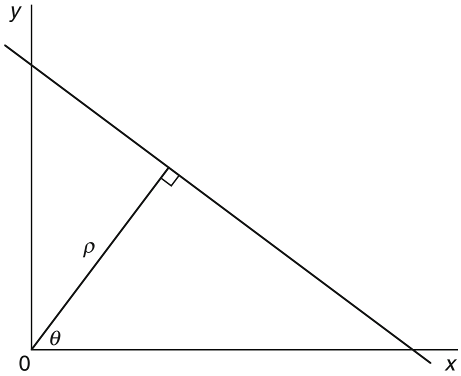


Fig. 18.1
Normal parametrisation of a straight line

Hence, we deduce:

$$y = \rho \operatorname{cosec} \theta - x \cot \theta \tag{18.7}$$

$$\rho = x \cos\, \theta + y \sin\, \theta \tag{18.8}$$

Once the parametrisation is fully defined, the computation of the Hough transform and the projection of the result back to image space can proceed as before (Note that under the normal parametrisation, the line of votes in the (m, c) parametrisation becomes a sine wave of votes, which amounts to a change of detail but not of principle).

The second improvement in the process is to make it less computation intensive [3]. This can be achieved by using local edge orientation information to arrange that a *single* vote is placed in parameter space for each edge point rather than a whole line (or other locus) of votes. This is possible because the value of θ is equal to the orientation of the edge normal, so $\theta = \arctan(g_y/g_x)$, g_x and g_y being the intensity gradient components obtained from the Sobel or a similar operator (see ➲ Sect. 18.3); hence a unique value of ρ can be deduced from ➲ Eq. 18.8.

18.5.1 Alternative Parametrisations

We have already seen two parametrisations that have been employed when the Hough transform is used for line detection. A number of others have also been developed for special reasons. In particular, the "foot-of-normal" transform was designed to save computation, in particular to eliminate the need for repeated evaluation of the arctan function [4]. In this parametrisation, the coordinates of the foot of the normal from the origin O to the line being detected are used as parameters. The method also conveys the advantage that the Hough transform peak position appears in a recognisable position when plotted in the original image, thereby confirming exactly what evidence the transform is finding. Perhaps oddly, the peak sometimes appears on an extension of the real line rather than on the actual straight boundary of an object (➲ *Fig. 18.2*). To calculate the foot of normal position F (x_f, y_f), it is first noted that the direction OF is identical to that of the edge normal vector g, so that:

$$g_y/g_x = y_f/x_f \tag{18.9}$$

In addition, the line is normal to OF, so its equation has to be:

$$(x - x_f)x_f + (y - y_f)y_f = 0 \tag{18.10}$$

Solving these two equations now yields the values of x_f and y_f:

$$x_f = \gamma g_x \tag{18.11}$$

$$y_f = \gamma g_y \tag{18.12}$$

where:

$$\gamma = \frac{x g_x + y g_y}{g_x^2 + g_y^2} \tag{18.13}$$

Clearly, these formulae lead to no calculation more complicated than division, thus saving computation.

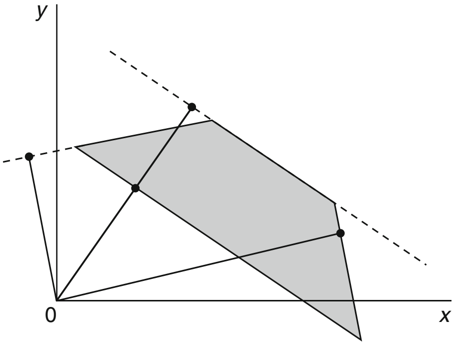


Fig. 18.2
Foot of normal parametrisation. Here the points at which the foot of normal Hough transform peaks appear are indicated by dots for the four sides of a trapezium shape (similar to that of a lino cutter blade). Note that in two cases these appear on extended sections of the object sides (see *dotted lines*)

An even more unusual parametrisation is the "muff" transform [5], which involves producing (In this chapter we use the term "producing" in the elementary geometry sense of extending a straight line in either or both directions to achieve some goal) every line segment that is located by an edge detector to the boundaries of the image, and accumulating the resulting crossing points in a set of four 1D histograms, which can be considered as simplified forms of Hough transform. Searching for peaks in four 1D histograms is many times faster than searching through a whole image (over 100 times faster for a 256 × 256 pixel image). Hence this parametrisation offers much faster search times. However, voting is twice as slow, and inferring the presence of lines from the histogram peaks can lead to significant ambiguities if many lines are present in the image. In fact, when choosing the most appropriate parametrisation, it is necessary to have a lot of information on the particular types of image data that will arise in practice.

18.5.2 Longitudinal Line Localisation

We have demonstrated the value of the Hough transform for locating straight lines in digital images, but it will be noted that each line is located in the form of an infinite straight line, whether or not the original line passed from one end to the other of the image. This is perhaps inevitable for a transform that is intended not to be confused by occlusions or breakages. Yet it demonstrates a weakness of the approach in situations where object boundaries are to be demarcated and no such distortions are expected. What is needed is a way of finding the longitudinal profile of any line, and in particular the locations of its ends and a measure of its length. This is quite straightforward to achieve [6]. All that is required is to re-run the Hough transform, and to plot an x or y projection of the edge points (depending on the orientation of the line) each time an edge point contributes to a known peak: examining the connectivity of the projection will then permit any end points to be determined – though small

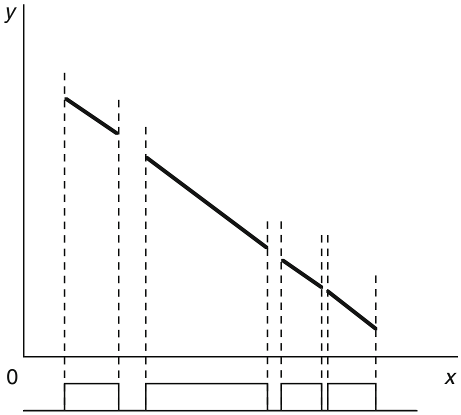


Fig. 18.3
Longitudinal line localisation. The parts of a straight line that have been located by a Hough transform are projected to form a histogram, from which connectivity can be ascertained. Typically, gaps smaller than a threshold distance are ignored, so that noise cannot confuse any following line interpretation algorithm

discontinuities may have to be disregarded as due to noise rather than gross occlusion or breakage (*Fig. 18.3*).

18.5.3 Final Line Fitting

Another factor in the design of a complete algorithm for line location is the optimisation of accuracy. In general the Hough transform itself is optimised for search capability in the presence of noise and other artefacts, i.e. it is an optimal detector rather than an optimal locator. To improve matters, a final line fitting stage based on least squares analysis is called for [6]. It is well known that least squares analysis is subject to inaccuracy if outliers are present (Inliers and outliers are defined as follows: when managing datasets it is usual to try to improve accuracy by averaging the data points; however, some points can have such extreme values that they introduce undue inaccuracy in the averaging process; these points are called outliers, and it is beneficial to remove them before averaging commences; the remaining points are called inliers). However, it is the function of the Hough transform to locate the line and to eliminate false data, so outliers should not be present at the final, accurate line fitting stage.

Overall, we now have an algorithm that operates in a number of stages:

1. Initialisation of parameter space
2. Edge detection in image space
3. Voting for peaks in parameter space
4. Locating significant peaks in parameter space
5. Transferring line information back to image space
6. Locating the ends of any lines in image space
7. Obtaining best fit line segments in image space

Occasionally, the peaks that are found in parameter space are noisy: in that case it is necessary to perform some smoothing before accurate estimates of peak position can be made. Other processes may be needed to suppress minor peaks, particularly if these arise from shadows in the original image. However, ad hoc suppression of this sort is not desirable, as it can destroy real data: if shadows are a problem, it is often better to eliminate them by more careful setup of the illumination, as described elsewhere in this handbook.

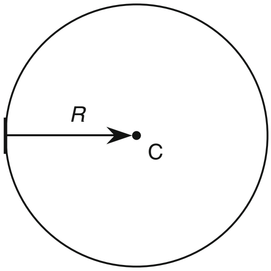
18.6 Circle Detection

Circle detection follows much the same pattern as line detection, but with a different object model, as appropriate to the situation. Clearly, any edge point could arise from a whole array of circles of different centre positions and sizes. It is convenient to consider first only circles of a given size – of radius *R*. In that case, the locus of circle centres is itself a circle of radius *R* centred at the given edge point. These observations lead to a Hough transform for circle detection in which votes are accumulated at a circle of points a distance *R* from each edge point found in the original image.

At this stage we again have the opportunity to save computation by making use of local edge orientation information [7]. Specifically, we move a distance *R* along the local edge normal and accumulate a single vote in parameter space (➲ *Fig. 18.4*). This means that for each edge point that is found in the original image we accumulate just one vote in parameter space. This procedure leads to a much greater relative concentration of votes in parameter space at the peaks corresponding to circle centres. In most other respects the Hough transform procedure is the same as for line detection, the stages in the detection algorithm now being the following:

1. Initialisation of parameter space
2. Edge detection in image space
3. Voting for peaks in parameter space
4. Locating significant peaks in parameter space
5. Transferring circle information back to image space
6. Obtaining best fit circles in image space

Note, however, that the need to locate ends that arose for lines is no longer relevant – though an adaptation of it would still be needed if semicircular objects were being located in place of circles.



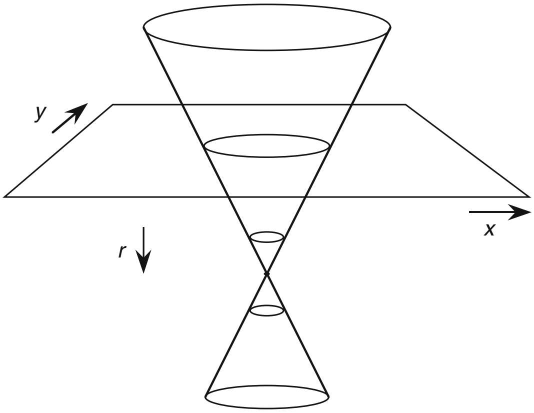
◘ Fig. 18.4
Circle detection by the Hough transform. Here an edge point on the left leads to a vote being accumulated at the centre C of a circular object

There are two more subtleties and differences relative to the case of line detection. The first is that the parameter space for circle detection is congruent to (i.e. of the same parametrisation as) that of the original image: this did not apply to the standard line detection Hough transform, though it did apply for the foot-of-normal Hough transform. The second difference is that circles have size, so there is an additional parameter to be determined. This problem will be discussed in the following section.

18.6.1 Case of Unknown Circle Size

The case of unknown circle size may be tackled by the simple expedient of applying a different version of the Hough transform for every distinguishable circle radius. Note that in the context of inspection, only a limited number of circle radii would be expected: e.g. for detecting and sorting coins, washers or ball bearings, only half a dozen sizes would be expected on a given product line or in a given application; similarly, on a biscuit line, only a small range of values – differing at most by ±10% from a standard product – would have to be coped with. Thus very little additional computation would be needed in such situations.

Nevertheless, it should be asked how the information from these various Hough transforms, each with its own 2D parameter space, should be combined. In fact, the answer is very simple: we interrogate each parameter space and select the one with the highest local peak; each such peak gives information not only on location, but also, though the value of the radius R that was used to construct it, it provides radius information as well. Interestingly, we can regard the complete set of Hough transforms used here as a single 3D (x, y, r) Hough transform, providing information on the three relevant circle parameter values, x_c, y_c, R (➲ *Fig. 18.5*). Indeed, for any application, we should first ask how many parameters p are needed to define the relevant curve: then we should design a Hough transform with p parameters to span the



◘ Fig. 18.5
Voting for circles in a 3D parameter space. This diagram shows the topmost circle being detected in a 3D parameter space. In any section of the parameter space a circle of reduced size appears. For one lower section the circle becomes a dot: at this location the density of votes accumulated is the highest in the whole 3D parameter space, indicating that this point gives the correct position and radius value of the original circle

relevant p-dimensional parameter space. This rationalisation makes the method described less ad hoc and part of a proper formalisation of the Hough transform, as originally outlined in ➲ Sect. 18.4.

A further subtlety is that, even in the case of unknown radius, circle location may still be carried out in a 2D parameter space [8]. This is achieved by tracing out a whole line of votes along the local edge normal, for whatever range of radius values is appropriate in the given application. This procedure places a great many votes in the 2D parameter space, and could in the end be extremely confusing for the peak location algorithm to negotiate. However, it has been found that in many situations good results can be achieved in this way. (Specifically, there must not be too much background clutter in the images, leading to too many votes arising in the same localities of parameter space.) The particular gains from this approach are the reduction in size of the parameter space (for a spatial resolution of 256 × 256 the effect is a very substantial reduction from ∼16 million to ∼64,000), and the consequent reduction in the computational effort, and time, needed to search it for peaks. The disadvantage of this approach is that the radii of the circles are not determined. However, once each centre has been located, a quick re-run of the transform can determine which points gave rise to it: then a separate 1D transform can be constructed showing for each peak what the distribution of radius values was, and leading to an accurate value for it. In this way we have replaced the original 3D Hough transform by a 2D plus a 1D Hough transform (The fact that this is a 1D Hough transform means that it does not merely average the radius values but can lead to identification and evaluation of radii for several concentric circles [as in the case of a set of washers].): as the computational complexity of the latter is rather small, the overall amount of computation is still close to that of a 2D transform. As remarked in ➲ Sect. 18.5.1, when choosing the most appropriate parametrisation, it is necessary to have a lot of information on the particular types of image data that will arise in practice, and this is definitely the case here too: in fact, the critical factor in this case is the amount of background clutter that appears in the image.

Finally, note that we have isolated one of the main characteristics of the Hough transform: its capability for efficient search for objects in images. Once a search has been carried out and objects have been characterised in any way, the problem has essentially been "cracked." Any remanent lack of information is far less relevant as the difficult part – the search through huge numbers of pixels for relevant information – has already been done, and the remaining part tends to be relatively trivial.

18.6.2 Overcoming the Speed Problem

In this chapter speed of processing has been alluded to a number of times, as it can be very important in real-time applications. It can be highly relevant in the case of circle detection. Two approaches aimed at speeding up the process of circle detection will now be outlined, the first involving simplified forms of edge detection and the second involving sampling.

Edge detection normally requires the application of at least two masks, as in the cases of the Sobel and Prewitt operators. Indeed, using two masks is necessary if local edge orientation is to be determined, and in turn this seemed to be necessary if the Hough transform is to be applied in such a way as to reduce the numbers of votes. However, if we try to use vertical edges – or rather, edges which have a substantial value of g_x – to detect circles, and couple the results for pairs of edges appearing on the same horizontal scan lines, we can efficiently obtain mid-points of horizontal circle chords. We can achieve similar results for vertical chords. We can then search for *x* and *y* values of circle centre locations, and obtain candidate centre locations. The process can be highly efficient because reduced size (e.g. 3 × 1 or 1 × 3) edge detection masks can be used. This means that if the image does not contain too many objects, and in particular *circular* objects, speedup factors of at least three may be attained [9].

The method outlined above requires two 1D search spaces to be used for estimating *x* and *y* coordinates, though these then need to be correlated with each other before circles can be identified. However, if the circle radius *R* is known, the circle centre (x_c, y_c) can be deduced and a normal 2D Hough transform vote can be accumulated [10]. For a horizontal chord with ends P (x_1, y) and Q (x_2, y), this is achieved (➋ *Fig. 18.6*) by noting that:

$$x_c = \frac{x_2 + x_1}{2} \tag{18.14}$$

$$(y_c - y)^2 + \frac{(x_2 - x_1)^2}{4} = R^2 \tag{18.15}$$

$$y_c = y \pm \left[R^2 - \frac{(x_2 - x_1)^2}{4} \right]^{1/2} \tag{18.16}$$

Note that there is a sign ambiguity, which necessitates double voting (In fact, each redundant vote can be eliminated by checking for an edge point at two suitable locations vertically above and below P or Q: this amounts to using triples of edge points as composite features): however, the Hough transform is able to cope with this, as the accumulation of votes will lead to only one peak of any size for any given circle.

For either of the two chord bisection methods given so far, considerable further speed improvements can be obtained by reducing the number of chords examined. In fact, we can sample the image on every *n*th horizontal and every *n*th vertical line, and thus speed up the processing by a factor close to *n*. The potential savings can be substantial, both in principle and in practice: e.g. in biscuit inspection, the combined speedup factor from simplified edge detection and sampling can be as high as a 25 [9].

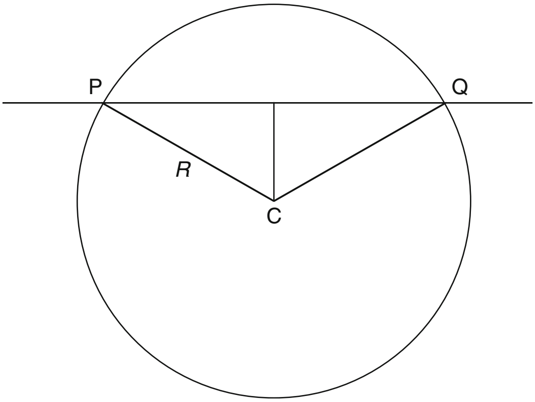


Fig. 18.6
Geometry for locating circle centres from single chords. Here the centre of a circle of known radius *R* is located from a single chord PQ. Note that there is an ambiguity in that C could be either above or below the chord PQ

The disadvantage of sampling is that circle detection becomes less robust and accurate as n increases. Reduced robustness (Although robustness is reduced, the amount it is reduced is known and controlled: it does not correspond to an arbitrary, unexpected loss of reliability.) follows since there may be insufficient evidence for the presence of the object in question and the sampling lines fail to identify sufficient numbers of whole chords – an especially important factor if the object surface is textured. Similarly, accuracy of location will fall because averaging over k data points (where $k \approx 2R/n$) results in errors being reduced by the factor $\sqrt{k}$, and hence the errors increase as n increases.

18.7 Ellipse Detection

Detection of ellipses in digital images presents an altogether more complicated problem than circle or line detection. This is because five parameters are required to define a general ellipse, compared with three for a circle and two for a line. (At first it might appear that only one more parameter – the eccentricity – would be needed to define an ellipse instead of a circle. However, the eccentricity makes the circle anisotropic and so it also acquires an additional parameter representing the direction of the major axis.) On the basis that each dimension brings with it something like a factor ∼256 more variation, the extra two dimensions are problematic. There is also a problem in deciding how exactly to parametrise the ellipse.

One approach is to make a simple modification to the Hough transform for a circle [11]. In the case of the circle, we move a given distance R along the local edge normal and accumulate a vote. However, in the case of a particular point on an ellipse we need to move a given distance R in a given direction ψ and accumulate a vote. Clearly, R and ψ will be different for different edge points. A solution to this is to ensure that at every edge point we make all possible votes corresponding to all possible positions we may be at on the ellipse boundary: in this way we are certain to vote for the ellipse centre co-ordinate, whichever edge point we are currently sitting on. We have already seen an example of this type of Hough transform, which occurred when we were searching for circles of unknown radius: we simply assume all possible values of radius in turn and make an appropriate vote for each of them. In the ellipse case, the votes will have to be made at distances from the edge point which vary between the values a and b (the semi-major and semi-minor axes lengths). It turns out that the set of votes we have to make fall on a locus which approximates to a small ellipse (➋ *Fig. 18.7*). The resulting method is effective and relatively fast, being of similar speed to that for circle location when the radius is unknown. However, to achieve optimum efficiency the locus of votes needs to be stored as a lookup table and then applied at the right position and orientation (for details see Davies [11]).

The procedure outlined above is rather complex and two alternative methods are also in common use. These are the diameter-bisection and the chord–tangent methods, which will be described next.

18.7.1 The Diameter-Bisection Method

The diameter-bisection method achieves simplicity by relying on the symmetry of the ellipse [12]. It examines all the edge points in the image and searches for pairs that are anti-parallel (oppositely orientated) so that they could lie at opposite ends of ellipse diameters; it then

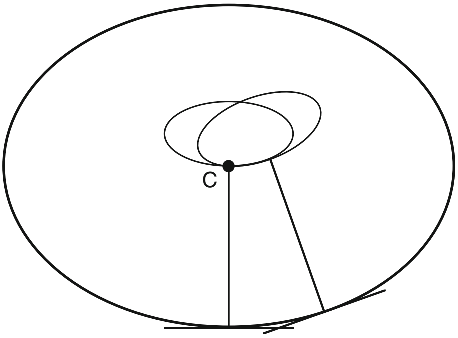


Fig. 18.7

Application of the generalised Hough transform for ellipse detection. Here a small locus of votes is accumulated for every boundary point. The diagram shows two such loci: the *straight lines* represent the tangents and normals at two boundary points, and merely serve to indicate the positions of the relevant loci

determines the mid-point of each such pair and makes a vote at this location in an image-like parameter space. Clearly, all such votes are potentially situated at ellipse centres. Thus, peak location will lead straight to ellipse detection.

To achieve this, it is first necessary to make a list of edge points for any image. Such a first stage is permissible for any of the Hough transform methods outlined so far, but in this case it is mandatory, as a systematic search through the edge points must be made. It should be noted that for E edge points, the number of pairs of edge points is quite large, namely $\binom{E}{2} = \frac{1}{2}E(E-1)$, and for $E \approx 1{,}000$ some half million pairs of points have to be considered. However, this does not mean that huge numbers of votes need be made in parameter space, the actual number being closer to $E/2$.

In general, the technique is highly effective, though it is seriously lacking in robustness. For example, if half of the ellipse is missing, no diameters will be definable and no valid votes will be cast in parameter space. Of course, if two opposite quarters of the circumference exist, there should be sufficient diameters to permit ellipse detection and location.

Another interesting characteristic of the method is that its reliance on symmetry will permit it to detect any symmetrical shape, such as a superellipse, though not an oval (egg-shape). Superellipses are defined by equations of the form:

$$\frac{x^m}{a^m} + \frac{y^m}{b^m} = 1 \tag{18.17}$$

where $m = 2$ corresponds to an ellipse, $m = 1$ corresponds to a diamond, and $m \rightarrow \infty$ corresponds to a rectangle. Thus the method is slightly non-specific regarding the shapes it is targeting (➋ *Fig. 18.8*). In some circumstances this could be advantageous. In others it may be less appropriate. However, presence of an ellipse can be confirmed by applying the test:

$$\frac{1}{(\mathrm{OP})^2} + \frac{1}{(\mathrm{OQ})^2} = \text{constant} \tag{18.18}$$

where OP and OQ are perpendicular chords of the shape. (In principle, the test must be applied to *all* pairs of perpendicular chords of the shape, but sampling will in most cases be sufficient.)

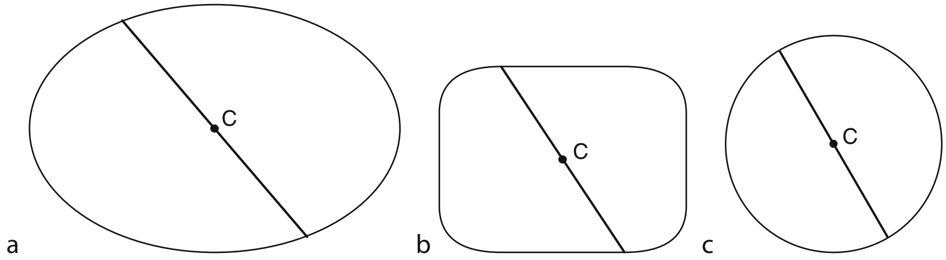


Fig. 18.8
Diameter-bisection method for ellipse detection. The examples make it clear that the technique also locates *superellipses* and *circles*, as well as a variety of other *symmetrical shapes*

Finally, it is interesting that the method will also find circles of any size: this approach can therefore be used in place of the 3D (x_c, y_c, R) circle Hough transform, though it is doubtful whether doing so would incur any additional advantage such as reduced computation. Suffice it to say that in some applications the reduced specificity of the diameter-bisection method may be an advantage, while in other applications it may constitute a serious limitation.

18.7.2 The Chord–Tangent Method

The chord–tangent method aims to make good the deficiencies of the diameter-bisection method – specifically, its distinct lack of robustness [13]. A list of edge points is again made, and pairs of these are selected in turn and used to create votes in parameter space. For any given pair of edge points, the chord joining them is constructed and the two tangents are drawn. The point T where the tangents cross is identified and this is joined to the midpoint M of the chord. Then the line TM is produced and the votes are cast along the produced section of the line. This line passes through the centre of the ellipse, so each set of votes includes one at the centre of the ellipse (➲ *Fig. 18.9*). When all pairs of edge points have been considered and all lines of votes have been cast, the parameter space is searched in the usual way for peaks, and the centres of any ellipses can be identified.

To prove that this method is valid, we note that it works for a circle: i.e. the line TM produced passes through the centre of the circle. This is clear from the symmetry of the situation. Next, we project the circle diagram so that it is viewed obliquely: this can always be done in such a way that an ellipse of any eccentricity can be produced. In orthographic projection, points project into points, chords project into chords, tangents project into tangents, and mid-points project into mid-points. Hence the property transfers validly from the case of a circle to the case for any ellipse, which proves that this form of the Hough transform does detect ellipses.

The next factor to consider is that of computational complexity. Again we have E edge points and $\binom{E}{2}$ pairs of edge points. However, this time we have to accumulate a line of votes for all of them. This means that the amount of computation scales as $O(E^3)$. Thus the method is extremely wasteful in accumulating unwanted votes – a factor to be balanced against the high level of robustness it offers. To keep computation within reasonable limits, votes should only be cast along the produced section of each line TM; they should only be cast over a distance of at

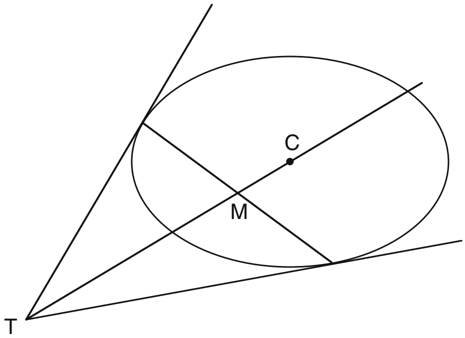


Fig. 18.9
Chord–tangent method for ellipse detection. Points on the line TM are accumulated in parameter space: note that there is no gain from accumulating votes between T and M, or beyond a certain point on the extended line

most the longest diameter ($2a$) of the expected types of ellipse; they should not be cast where the two edge points under consideration are more than a distance $2a$ apart. If necessary, various conditions on the directions of the edges and those of the line TM could be imposed, but such techniques could impose greater computational penalty than they save.

Overall, this method is robust, and is fairly computation intensive. However, it is specific to ellipses and will not detect superellipses or other symmetric shapes. Hence no special tests need be applied to confirm that the shapes being detected are actually ellipses.

18.7.3 Ultra-Fast Ellipse Location

Ellipse location is especially important because ellipses arise not only in their own right but also as projections of circles, which are themselves ubiquitous in manufactured products. Indeed, a purist could argue that in general one never sees circles – only ellipses. In any case, it is often the case that rapidly operating means must be found for locating ellipses. In recent work aimed at achieving the most rapid possible means of ellipse location, the "triple bisection algorithm" was developed [14]. This involved taking two parallel chords of the ellipse, bisecting them, joining the two bisectors, producing the resulting line to determine its ends, and then bisecting the resulting chord (➜ *Fig. 18.10*). The final chord bisector will be at the centre of the ellipse, if indeed the object found is an ellipse. In applications such as wheat grain location, when the wheat grains are on a moving conveyor, there are no other objects present, and the wheat grains appear as ellipses with an approximately 2:1 aspect ratio (i.e. $a/b \approx 2$).

The triple bisection algorithm generally works well, but may fail if two objects are touching (in which case one of the chords may pass from one object into the other). If that is a possibility, the method needs to be adapted to a more robust Hough transform format. (Strictly, the triple bisection algorithm is not a Hough transform, as votes are not accumulated, but a minimal computation alternative to it. However, in the present context, it does help to rehearse the idea of what is and what is not a Hough transform, and what the gains and losses are.) The resulting "eight-chord algorithm" has the merit of being sensitive to errors and of being able to revert progressively to a more reliable approach, as necessary to solve the problem [15].

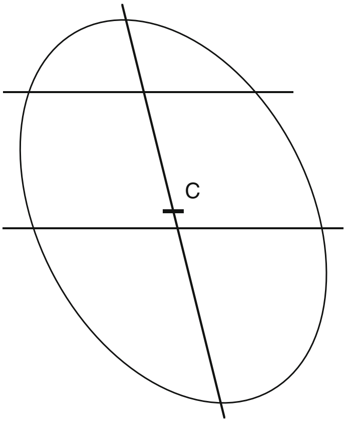


Fig. 18.10
Triple bisection algorithm. Here two parallel chords are bisected, and the bisector of the *line* joining the midpoints is taken to be the centre C of the ellipse

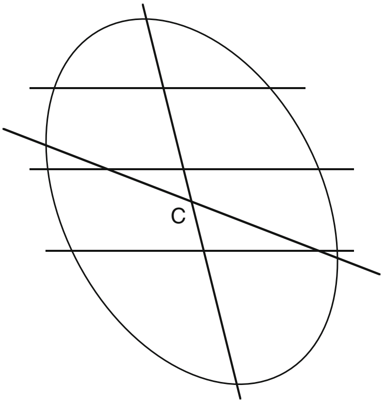


Fig. 18.11
Eight-chord algorithm. Here three parallel chords are bisected, and it is confirmed that a single line passes through the three bisectors. The centre C is located as the crossing point of two *lines* obtained from parallel chords of two different orientations. For clarity only one set of parallel chords is shown

In this method three parallel chords are used and it is checked that their midpoints lie on a straight line. Then another three parallel chords of different orientation are used and a similar check is made. Finally, the crossing point of the two bisecting straight lines is taken to be the centre of the ellipse (➲ *Fig. 18.11*). However, if the midpoints for either orientation do not lie on a straight line, further parallel chords are taken until sufficient midpoints do identify a unique line. (For this algorithm there is a build-up of voting evidence from the bisecting chords, so it may be classed as a genuine, albeit minimal, Hough transform.)

Both of these algorithms rely on the bisectors of sets of parallel chords lying on straight lines which pass through the centre of the ellipse. We have already seen that this property is valid for a circle. The proof that the property is valid for an ellipse follows the same type of projective proof as for the chord–tangent method (a suitable orthographic projection of the circular case can be selected to demonstrate the situation for any ellipse).

18.8 More Advanced Aspects of the Hough Transform

In the foregoing sections, it may seem that the Hough transform is suited to the location of a very restricted number of shapes, specifically circles, ellipses and those such as rectangles that are bounded by straight lines. However, this is by no means the case. Methods have been devised for detecting other specific shapes such as parabolas, and also for more general shapes which may be described either analytically or as lookup tables. This leads to the first of three advanced aspects of the Hough transform.

18.8.1 The Generalised Hough Transform

The generalised Hough transform [16] is in principle a straightforward generalisation of the circle Hough transform. Instead of proceeding a distance *R* directly along the edge normal at a boundary point *E*, we move an adjustable distance *R* along an adjustable direction ψ, and cast a vote at that location (➋ *Fig. 18.12*); more simply, we can describe this location as lying at a vector distance **R** from *E*. In fact, **R** is taken as a function $\mathbf{R}(\theta)$ of the local edge orientation θ. In many cases the function $\mathbf{R}(\theta)$ would be specified analytically, but in complicated cases, or when the computer is trained to cope with arbitrary shapes, it is specified as a lookup table. In either case it is commonly referred to in the literature as the *R*-table.

It is unfortunate that circles are the only isotropic shapes, so the *R*-table for more general shapes will vary with object orientation φ. Hence in reality the *R*-table will have to contain all

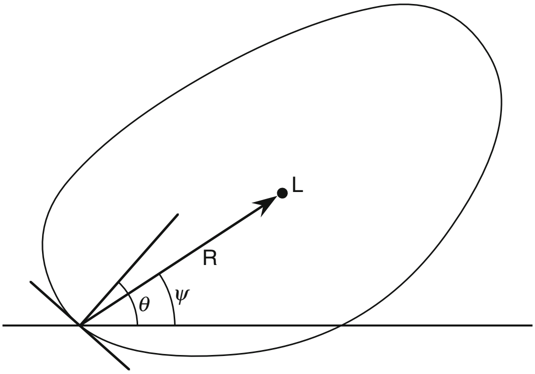


Fig. 18.12
The generalised Hough transform. The point L is taken as a reference point for the basic shape. To detect the shape, a vote is accumulated at the position of L that is estimated for each boundary point: L is assumed to lie at a vector location R relative to the boundary point, where R is a function R(θ) of the local edge orientation θ

the information needed to cope with variable object orientation. However, this is not an insurmountable problem. Indeed, the first method given in ➔ Sect. 18.7 for locating ellipses was a form of generalised Hough transform; and the Hough transform used for point pattern matching (➔ Sect. 18.9.1) can also be classified as an appropriately adapted form of generalised Hough transform.

18.8.2 Relation to Spatial Matched Filtering and Gradient Weighting

The idea of voting that the Hough transform embodies may seem a little strange from a signal processing point of view. However, it has its origins in matched filtering, which is known to be an optimal means of detection in a signal-to-noise sense [17]. In fact, matched filtering is of supreme importance in radar, where it is crucial to optimise the detection of aircraft, because, with the inverse square law operating twice, the amount of reflected energy arriving back at the radar system will typically be 10^{12} times smaller than the energy originally transmitted. In image analysis applications the signal is 2D and we should be talking about spatial matched filtering rather than matched filtering [1, 16].

The way of designing an optimal matched detector is to convolve the received signal with an idealised but reversed form of the expected signal [17]. However, image analysis differs from the situation for radar, not merely in replacing the time variable by two spatial dimensions, but also in that the level of background brightness tends to be arbitrary, so a zero-mean template is required for matching. (This is illustrated by the template masks given in ➔ Sect. 18.3). In addition, the brightness will tend to vary gradually but unpredictably over the whole image, so it is better to examine all parts of the image separately for evidence of the existence of objects, and then to accumulate the signal in sections (one contribution for each part of the image examined). This leads us to a basic understanding of why votes have to be accumulated for all the image features found.

However, there is another consequence of this analysis. The number of votes, or their amplitudes, ought to be proportional to the response received at each feature point. Specifically, features should not be regarded as either present or absent at different locations in an image – as a result of an arbitrary threshold being set: they should be regarded as present to a specific degree, and the votes cast in proportion to the strength of the evidence they embody. It has been shown that this model does produce a significant improvement in performance of the Hough transform when applied to noisy images, and is generally worth the additional computation that it involves [18]. On the other hand, in many cases the amount of clutter in images can be the real problem that limits object detection, and elimination of outliers will then be more important than correct weighting of inliers to obtain optimal signals.

In fact, there is a final twist in the story of how matched filtering gave rise to the Hough transform. In a matched filter the weight given to the final signal is in proportion not only, as stated above, to the received signal but also to the size of the expected signal [18]. This means that votes should be accumulated in proportion to the magnitude of the signal that is expected at a particular point in any object. While this criterion is somewhat difficult to apply in general (as it is not known what part of an object a feature may come from), if it can be applied it is beneficial to do so. Likely cases when it could be applied are when some edges are low contrast and some high contrast, and can be identified as such: in that case an additional weighting proportional to the expected contrast should be applied.

18.8.3 The Particular Value of the Hough Transform

At this point there may be some confusion over the real value and purpose of the Hough transform. It may seem that its value resides in shape analysis and recognition. However, this is one aspect of its value. More properly, it should be viewed as a method for searching for required objects amongst a clutter of other objects and noise. (Note also that other examples of the required type of object amount to clutter when a particular object is under consideration.)

In general, search is a highly computation intensive task, and special methods are needed to perform it efficiently. However, if an image contained only a single object, search would still be required to find it, but this type of search could probably be carried out by a rather basic method such as thresholding; in such a case, shape recognition could then be carried out using the centroidal profile approach. However, when the situation is not so straightforward, and neither thresholding nor centroidal profiles will work well or even at all, the Hough transform comes into its own, both searching for and recognising the shapes it is tuned to. It is best used when both of these tasks have to proceed in tandem: when either has to be done on its own, there may be other methods that work better. A further point will illuminate the situation. If the location of an object is known approximately, there will be little point in using the Hough transform to locate it accurately: it will be better to employ a method such as least squares analysis to refine the accuracy. Least squares analysis is unable to eliminate outliers but is excellent at averaging to reduce the error from inliers: the Hough transform is excellent at eliminating outliers (including irrelevant objects) but is relatively poor at averaging to reduce the error from inliers.

18.9 Point Pattern Matching Techniques

As pointed out in ➲ Sect. 18.3, there are four particularly important types of feature by which objects may be recognised and located, namely edge segments, line segments, small holes and corners. The first two of these are extended features and are commonly used for locating objects by their boundaries or internal divisions; on the other hand the last two are point features which are accurately locatable in two dimensions, being characterised by a specific pair of (x, y) coordinates. We have already seen how the Hough transform may be used with good effect with the extended type of feature. We shall now see how point pattern matching techniques may be used to locate objects from their point features.

At first sight point pattern matching would appear to be especially simple: first locate the object from a single point feature; then use a second feature to uniquely orientate it. Unfortunately, in many cases features may be missing from the image, and extra features may appear that are not due to the type of object being sought. In detail, the problems are (Essentially the same list applies whatever types of features are used to recognise objects, or whether Hough transforms or point pattern matching are used for analysing the feature point placements):

1. Noise or image clutter may give rise to additional features.
2. Noise may eliminate some features.
3. Poor lighting or shadows may reduce contrast so much that some features may not be visible.
4. Partial occlusion by other objects may cause some features to disappear.
5. Slight perspective distortions may shift features from their ideal locations.

6. Lens distortions (including barrel distortion) may shift features from their ideal locations.
7. Camera to object distances may be shifted so that features may not be at their ideal locations.
8. The objects themselves may be defective or distorted, again shifting features or preventing them from being found.

In case 8, it may be part of an inspection task to identify objects in spite of their being defective or distorted, or to identify the non-defective or non-distorted sections so that the defective or distorted sections may be highlighted.

All these effects mean that the object location software may not receive complete, ideal data, so it will have to do the best it can with non-ideal data. Thus it must embody robust algorithms. Note also that it will have to be capable of distinguishing features on one object from those on another. To achieve all this, the algorithms used must be able to cope successfully with the redundancy arising from a number of features.

To proceed, we consider the input image as a graph G, and the template of an ideal object as another graph T (A *graph* is defined as a diagram containing a set of points (sometimes called *vertices*), some of which may be joined by lines (sometimes called *edges*)). We now need an algorithm which searches for instances of possibly incomplete parts of T within G. We can re-express the requirement as the identification of subgraphs of T in subgraphs of G: this is often called the subgraph–subgraph isomorphism problem. Note that, to be optimally useful, the method needs to identify maximal subgraphs of T which appear in G.

To carry out this task, we need some criterion to assess the match between the template and image subgraphs. Perhaps the simplest criterion is when the distances between corresponding point features are the same (within a suitable practical tolerance limit) in the two subgraphs. First take separation distance as D_{ij} in the image and D_{pq} in the template, for some i, j, p, q; then if $D_{ij} = D_{pq}$, we can deduce a possible compatibility between i, j in the image and p, q in the template, such that $i = p$ and $j = q$, or else $i = q$ and $j = p$: the alternative arises because the proper ordering of the points in the image will not be known before the analysis, and separation distance will not immediately distinguish between the two possibilities.

We now take a five-point image (1, 2, 3, 4, 5) and a four-point template (A, B, C, D) as example (➋ *Fig. 18.13*): in this case the valid feature compatibilities are given by 1 = A, 2 = B, 3 = C, 4 = D. However, this needs to be inferred by a computer algorithm. The base information that the computer algorithm will receive as input is that $D_{12} = D_{\mathrm{AB}}$, etc. We next get the computer to build a "match graph" showing the compatibilities between pairs of feature assignments. Thus $D_{12} = D_{\mathrm{AB}}$ means that A1 (the assertion that feature 1 is in class (A) is compatible with B2, though A2 must also be compatible with B1). Hence these two compatibility lines are included in the match graph (➋ *Fig. 18.13c*). For an object with four points, all of which appear in the image, there will be a total of $\binom{4}{2} = 6$ pairs of features, each of which will produce two compatibility lines. In addition, the image shown in ➋ *Fig. 18.13c* contains feature 5, which is in such a position that $D_{35} = D_{\mathrm{CD}}$ and $D_{45} = D_{\mathrm{BD}}$. If there had been no such equalities, the match graph would not have needed to use the label 5 at all, and the feature point could have been totally ignored (It is easy to say this with hindsight from our percipient human perspective. However, in the computer algorithm there is no option other than to include label 5 in the initial match graph); but with these equalities, four additional compatibility lines are included in the match graph. At this point all 16 compatibility lines are marked in the match graph, and it is possible to start inferring the presence of objects in the image.

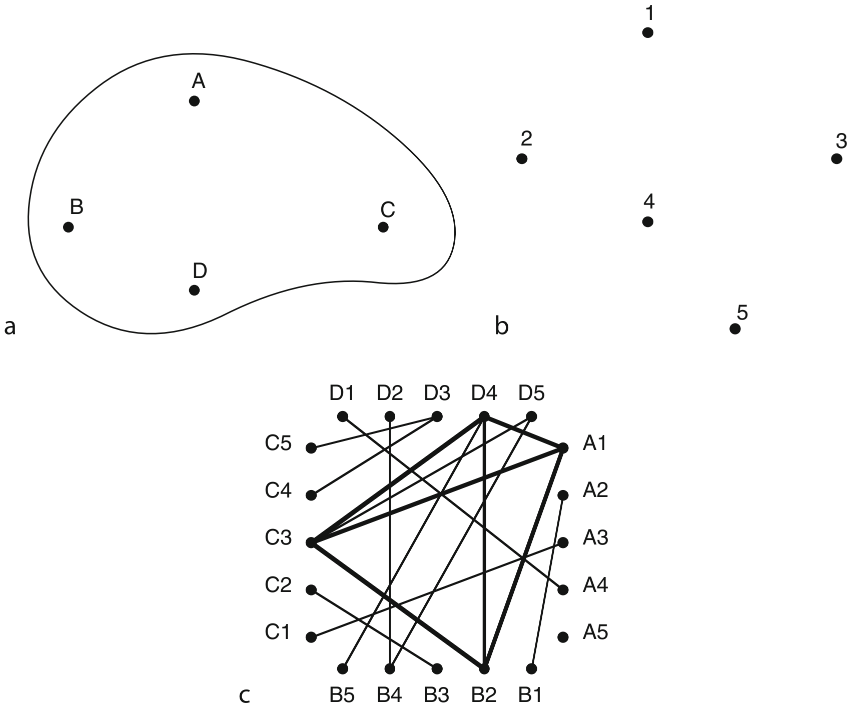


Fig. 18.13
The maximal clique approach to object location. (a) shows the template of an idealized object with point features A, B, C, D; (b) shows the point features 1,2,3,4,5 actually observed in an image; (c) shows the match graph for this image, the most likely interpretation of the data being represented by the largest maximal clique, in this case the quadrilateral A1–B2–C3–D4, which is indicated by *thicker compatibility lines*

To do this, the initial discussion showed that we need to look for maximal subgraphs of *T* within *G*. To achieve this we can search for maximal completely connected subgraphs (or "maximal cliques") within the match graph, as these represent complete agreement between the subgraphs (Completely connected subgraphs are called cliques: this reflects the fact that people in a "clique" all know each other, and thus in the present sense are compatible with each other. Here, however, we take the concept of a clique further and look for maximal cliques – which are those cliques that are not completely contained in any larger clique). In the match graph of *Fig. 18.13c*, there are 11 maximal cliques, A1–B2–C3–D4, A2–B1, B3–C2, C4–D3, D1–A4, B4–D2, C1–A3, B4–D5, B5–D4, C3–D5, C5–D3. Compatibility lines such as B4–D5 form two-element maximal cliques: these often (as here) have to be interpreted as due to chance distance equalities arising from noise features. However, two-element maximal cliques can also mean that there is an instance of an object which is oppositely orientated and for which there are only two support features. Clearly, it is far more likely that the four-element maximal clique A1–B2–C3–D4 (shown in thicker lines in *Fig. 18.13c*), which incorporates these same image features, is the most likely interpretation.

We now see that we are not just looking for maximal cliques: we are looking for the *largest* maximal cliques as these represent the *most likely* interpretations of the image features [19]. Of course, just stating that the largest maximal cliques are the most likely solutions could be invalid if the same set of features appeared in two large maximal cliques. However, such circumstances are rare. Specifically, it is unlikely that noise or image clutter will give rise to large maximal cliques. Nevertheless, to eliminate such problems, the match graph should probably be interpreted sequentially, by looking for the largest maximal clique; then eliminating from the match graph all the compatibilities between its features; then proceeding to the largest remaining maximal clique; and so on. In *Fig. 18.13*, eliminating clique A1–B2–C3–D4 and its features will mean also eliminating all the two-element cliques. Thus feature 5 will be seen to have no support from any other features and may be disregarded as due to noise. It is as well to note that these procedures for making sense of the match graph are somewhat ad hoc, and the solutions obtained should be regarded as hypotheses to be checked by further analysis of the image data. In fact, the same cautions apply as for the Hough transform, where the highest peaks in the parameter space are most likely, but not certain, to represent correct solutions. In both cases, the number of votes or the number of compatibility lines in a clique represent the amount of support for the solution in question, and thereby increase the probability that it represents the correct solution.

Unfortunately, graph matching algorithms based on finding maximal cliques take a long time to run because of the need to check all the possible pairs of elements that might form part of a clique. Indeed, this type of algorithm is "NP-complete," in that it runs in exponential time rather than polynomial time, relative to the number of image features involved. While the maximal clique approach has itself been speeded up [20], it has also proved possible to perform the graph matching task described above in an analogous way using the generalised Hough transform, as we shall see below.

18.9.1 Using the Hough Transform to Save Computation

The generalised Hough transform approach was described earlier in the context of ellipse detection, and again later in its full generality. In fact, it has a particularly neat form when applied to the location of objects from point features. All that is needed is to define a reference point within an object, and use pairs of features a specific distance apart to vote for the existence of the object at its reference point [21]. *Figure 18.14a* shows the same object as in *Fig. 18.13a*, with the reference point indicated as a small circle. If the topmost and leftmost feature points are used to vote for this reference point, the algorithm will not (as in the graph matching case) know which is which of the two feature points, so two votes will have to be cast – as indicated by the two small circles in *Fig. 18.14b*. When all the voting has been carried out for the image of *Fig. 18.13c*, the result is as shown in *Fig. 18.14c*: here, the small circles correspond to single votes, and the bold circle at the reference point corresponds to six votes. Interestingly, the total number of votes is the same as the number of compatibility lines in *Fig. 18.13c*, and the breakdown 16 = 6 + 10 is also the same. Thus there is a great synergy between the methods: the important difference is that the analysis of the parameter space is far easier and less computation intensive than for the match graph, and takes place in polynomial time: the execution time is actually proportional to the square of the number of feature points in the image, because all pairs of feature points have to be considered [21].

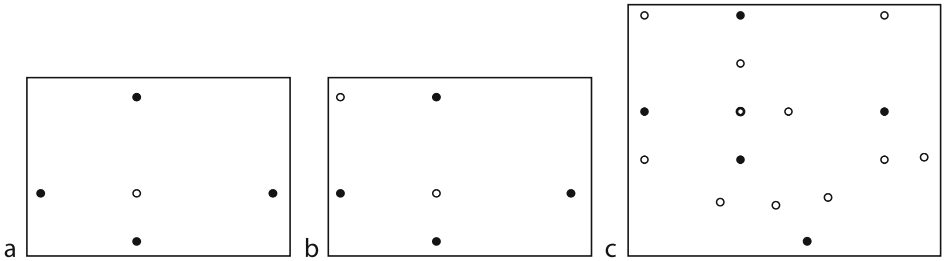


Fig. 18.14

Use of the generalised Hough transform for locating objects from point features. (a) shows the template of an idealized object, as in *Fig. 18.13a*, but with the reference point marked by a *small circle*. (b) shows the voting positions for the reference point when the topmost and leftmost point features are examined. (c) shows all the voting positions for the image of *Fig. 18.13b*: six votes are accumulated at the true position of the reference point, and are marked with a *small bold circle*

18.10 Perspective on the Development and Use of the Hough Transform

The Hough transform (HT) was developed in 1962 as an aid to the location of straight line high-energy particle tracks [2], and was introduced to the wider image processing literature only in 1969 [22]. Duda and Hart (1972) were the first to use it for circle detection [23]. In the mid-1970s, it was shown how to use edge orientation information to make line and circle detection more efficient [3, 7]. Dudani and Luk's classic (1978) paper was the first to give a definitive account of line localisation using the HT [6]. Gerig and Klein's key (1986) paper on circle detection used a two-stage approach to achieve greater robustness in the presence of clutter [24]. Subsequently, a series of papers demonstrated that the HT was formally equivalent to both template matching [25] and spatial matched filtering [26], and extended these results to the generalised Hough transform (GHT) [16]. One of the most important ideas since then has been the development of the randomised HT, which involves casting votes only until specific peaks in parameter space become evident, thereby saving unnecessary computation [27]. The diameter bisection, chord–tangent and GHT speedup techniques for ellipse detection, described in this chapter, originate respectively from papers by Tsuji and Matsumoto [12], Tsukune and Goto [28], and Davies [11]. The idea of saving dimensionality in the implementation of the GHT was an increasingly important theme in the late 1980s [8, 13]. Gradient weighting of votes for improving detection sensitivity also became an important topic around this time [18], and is still of decided interest [29].

Graph matching and clique finding algorithms started to appear in the literature around 1970. Bolles (1979) applied the maximal clique technique to engineering applications [30], and Bolles and Cain [19] extended the approach in a number of ways, including use of the symmetry-reduced match graph. Kasif et al. [31] showed how a modified GHT (the "relational HT") could be used for graph matching, although their paper gave few practical details. Subsequently, Davies [21, 32, 33] applied the GHT to point pattern matching to improve speed and accuracy.

Remarkably, considering the simplicity and importance of ellipses, new ellipse detection schemes are being developed even in the 2000s. Xie and Ji [34] developed an efficient ellipse detection technique, while Lei and Wong [35] employed a method which was based on symmetry: the latter was found to be able to detect parabolas and hyperbolas as well as ellipses.

Something of the power of the HT can be seen from the range of applications for which it is nowadays being used. These include the following:

- Toennies et al. [36] showed how the HT can be used to localise the irises of human eyes for real-time applications, one advantage of the transform being its capability for coping with partial occlusion – a factor that is often quite serious with irises.
- Wang and Sung [37] applied the HT not only to iris location but also to eye gaze determination: they found that the method had to be refined sufficiently to detect the irises as ellipses, not only because of iris orientation but also because the shape of the eye is far from spherical and the horizontal diameter is larger than the vertical diameter.
- Wu et al. [38] extended the 3-D possibilities further by using a 3-D HT to find glasses: sets of features that lie on the same plane are interpreted as the glasses rim plane. The technique allows the glasses to be separated from the face, and then they can thus be identified and located in their entirety.
- Schaffalitsky and Zisserman [39] extended earlier ideas on vanishing lines by considering the case of repeated lines such as those occurring on fences and brick buildings.
- Tuytelaars et al. [40] described how invariant-based matching and HTs can be used to identify periodicities in planes appearing within 3D scenes in spite of perspective skew.

Finally, a useful set of papers on projection-based transforms (as some of the developments of the HT are called) appears in [41].

18.11 Concluding Remarks

In this chapter we have seen something of the power of the Hough transform in performing the inference tasks required to locate objects efficiently and robustly in digital images. Remarkably, the method is able to overcome a great variety of potentially lethal interfering factors, including noise and background clutter, and problems with the objects being detected – such as distortions, breakages or partial occlusions. The method is very readily applied to the detection and location of circles, ellipses and objects bounded by straight edges, and there are a number of variants of the technique that may be used in these cases, offering differing degrees of specificity, robustness, accuracy and speed. In addition, the generalised Hough transform will recognise a good many other shapes, and indeed is completely general in this respect, though it may require more computation to cope with the additional shape parameters and complexity.

The need for the Hough transform arose because applying the obvious template matching technique is normally impracticable for objects of any substantial size – especially if their orientation is unconstrained. However, template matching is still suitable for locating small features, and the Hough transform extends this to whole-object detection by using an inference approach, which is also responsible for the method's robustness. The Hough transform has long been used when the features in question are edge and line features. However, when point features such as small holes and corners are chosen, it used to be usual for graph matching techniques such as the maximum clique approach to be employed. More recently, it has been demonstrated that a suitably adapted form of the generalised Hough transform gives essentially the same solutions, but is also much faster in operation.

The Hough transform has been seen to embody both search, recognition and location capabilities in a single approach. However, if there is only a single object in an image, simpler, more efficient search methods can often be applied to find it – as for example in the case of the triple bisection algorithm. Thus the real value of the Hough transform resides in its capability for effective search in a cluttered, noisy background. Similarly, it is less good at especially accurate object location, as it is optimised for outlier rejection rather than averaging inlier data. This means that it is often useful to apply a least squares fitting procedure once the Hough transform has completed its search for relevant data.

Acknowledgements

This research has been supported by Research Councils UK, under Basic Technology Grant GR/R87642/02.

References

1. Davies ER (2005) Machine vision: theory, algorithms, practicalities, 3rd edn. Morgan Kaufmann, San Francisco
2. Hough PVC (1962) Method and means for recognising complex patterns. US Patent 3069654
3. O'Gorman F, Clowes MB (1976) Finding picture edges through collinearity of feature points. IEEE Trans Comput 25:449–456
4. Davies ER (1986) Image space transforms for detecting straight edges in industrial images. Pattern Recogn Lett 4:185–192
5. Wallace RS (1985) A modified Hough transform for lines. IEEE international conference on computer vision and pattern recognition, San Francisco (11–13 June), pp 665–667
6. Dudani SA, Luk AL (1978) Locating straight-line edge segments on outdoor scenes. Pattern Recogn 10:145–157
7. Kimme C, Ballard D, Sklansky J (1975) Finding circles by an array of accumulators. Comm ACM 18:120–122
8. Davies ER (1988) A modified Hough scheme for general circle location. Pattern Recogn Lett 7:37–43
9. Davies ER (1987) A high speed algorithm for circular object location. Pattern Recogn Lett 6:323–333
10. Davies ER, Barker SP (1990) An analysis of hole detection schemes. In: Proceedings of the British machine vision association conference, Oxford (24–27 September), pp 285–290
11. Davies ER (1989) Finding ellipses using the generalised Hough transform. Pattern Recogn Lett 9:87–96
12. Tsuji S, Matsumoto F (1978) Detection of ellipses by a modified Hough transform. IEEE Trans Comput 27:777–781
13. Yuen HK, Illingworth J, Kittler J (1988) Ellipse detection using the Hough transform. In: Proceedings of the 4th Alvey Vision Conference, Manchester, 31 August–2 September, pp 265–271
14. Davies ER (1998) Rapid location of convex objects in digital images. In: Proceedings of the European Signal Processing Conference (EUSIPCO'98), Rhodes, Greece (8–11 Sept.), pp 589–592
15. Davies ER (1999) Algorithms for ultra-fast location of ellipses in digital images. In: Proceedings of the 7th IEE international conference on image processing and its applications, Manchester (13–15 July), IEE Conference Publication no. 465, pp 542–546
16. Ballard DH, Brown CM (1981) Generalizing the Hough transform to detect arbitrary shapes. Pattern Recogn 13:111–122
17. Davies ER (1993) Electronics, noise and signal recovery. Academic, London
18. Davies ER (1987) A new framework for analysing the properties of the generalised Hough transform. Pattern Recogn Lett 6:1–7
19. Bolles RC, Cain RA (1982) Recognizing and locating partially visible objects: the local-feature-focus method. Int J Robotics Res 1(3):57–82
20. Davies ER (1991) The minimal match graph and its use to speed identification of maximal cliques. Signal Process 22(3):329–343

21. Davies ER (1991) Alternative to abstract graph matching for locating objects from their salient features. Image Vision Comput 9(4):252–261
22. Rosenfeld A (1969) Picture processing by computer. Academic, New York
23. Duda RO, Hart PE (1972) Use of the Hough transformation to detect lines and curves in pictures. Comm ACM 15:11–15
24. Gerig G, Klein F (1986) Fast contour identification through efficient Hough transform and simplified interpretation strategy. In: Proceedings of the 8th international conference on pattern recognition, Paris (27–31 October), pp 498–500
25. Stockman GC, Agrawala AK (1977) Equivalence of Hough curve detection to template matching. Comm ACM 20:820–822
26. Sklansky J (1978) On the Hough technique for curve detection. IEEE Trans Comput 27:923–926
27. Xu L, Oja E (1993) Randomized Hough transform (RHT): basic mechanisms, algorithms, and computational complexities. CVGIP Imag Understanding 57(2):131–154
28. Tsukune H, Goto K (1983) Extracting elliptical figures from an edge vector field. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Washington, pp 138–141
29. Basalamah S, Bharath A, McRobbie D (2004) Contrast marginalised gradient template matching. In: Pajdla T, Matas J (eds) 8th European Conference on Computer Vision (ECCV 2004), Prague, Czech Republic. Springer, Berlin, pp 417–429
30. Bolles RC (1979) Robust feature matching via maximal cliques. SPIE, 182. In: Proceedings of the technical symposium on imaging applications for automated industrial inspection and assembly, Washington DC (April), pp 140–149
31. Kasif S, Kitchen L, Rosenfeld A (1983) A Hough transform technique for subgraph isomorphism. Pattern Recogn Lett 2:83–88
32. Davies ER (1988) An alternative to graph matching for locating objects from their salient features. In: Proceedings of the 4th Alvey Vision Conference, Manchester, 31 August–2 September, pp 281–286
33. Davies ER (1992) Locating objects from their point features using an optimised Hough-like accumulation technique. Pattern Recogn Lett 13(2):113–121
34. Xie Y, Ji Q (2002) A new efficient ellipse detection method. In: Proceedings of the 16th International Conference on Pattern Recognition, Québec, Canada, 11–15 August, vol II, pp 957–960
35. Lei Y, Wong KC (1999) Ellipse detection based on symmetry. Pattern Recogn Lett 20(1):41–47
36. Toennies K, Behrens F, Aurnhammer M (2002) Feasibility of Hough-transform-based iris localisation for real-time application. In: Proceedings of the 16th international conference on pattern recognition, Québec, Canada, 11–15 August, vol II, pp 1053–1056
37. Wang J-G, Sung E (2001) Gaze determination via images of irises. Image Vision Comput 19(12): 891–911
38. Wu H, Yoshikawa G, Shioyama T, Lao S, Kawade M (2002) Glasses frame detection with 3D Hough transform. In: Proceedings of the 16th international conference on pattern recognition. Québec, Canada, 11–15 August, vol II, pp 346–349
39. Schaffalitsky F, Zisserman A (2000) Planar grouping for automatic detection of vanishing lines and points. Image Vision Comput 18(9):647–658
40. Tuytelaars T, Turina A, van Gool L (2003) Noncombinatorial detection of regular repetitions under perspective skew. IEEE Trans Pattern Anal Mach Intell 25(4):418–432
41. Davies ER, Atiquzzaman M (eds) (1998) Special issue on projection-based transforms. Image Vision Comput 16:9–10

19 Morphological Image Processing

Bruce G. Batchelor[1] · *Frederick M. Waltz*[2]
[1]Cardiff University, Cardiff, Wales, UK
[2]University of Minnesota, Falcon Heights, MN, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_19,

Abstract: Morphology applies certain simple rules and shapes, such as squares, circles, diamonds, cubes and spheres, to process images. The objective is usually to identify features of interest in images as a prelude to performing high-level inspection, or machining, functions. Navigating around complex objects is facilitated using ‘anchor point’ features detected by morphology. The chapter begins by defining four basic morphology operations for binary images: erosion, dilation, opening, and closing. The application of these operations to analysing images from real-world vision tasks is discussed, as are techniques for fast implementation. Further operations akin to morphology are also described, including skeletonisation, hit-and-miss transform, and binary template matching. The extension of binary morphology into grey-scale morphology is also described and illustrated.

19.1 Introduction

The linguistic root *of morphology* is a classical Greek word (μορφή) meaning *form* or *shape*. The word is used in biology, to refer to the study of plant/animal shape and structure and is used in linguistics, to refer to the identification, analysis and description of units of meaning in a language. Our subject, is properly called *Mathematical Morphology* but we shall refer to it simply as *Morphology*. It too is concerned with recognising elemental shapes, this time in digital images. Morphology uses simple rules to process images and employs two- or three-dimensional templates, such as lines, (e.g. I, –, / and \), crosses (+ and X), composites (V, T, L and II), squares (open and closed), circles (open and closed), diamonds, etc. The templates may be derived instead from a ‘golden sample’ image. Erosion and dilation are two of the simplest morphology operations. The former makes binary objects (blobs) smaller, while dilation increases their size. The blobs may be shrunk or extended non-isotropically in certain directions, defined by the shape of the template. Identifying image features that match the given template is a particular benefit of morphology. Another possible function of morphology is preprocessing to enhance feature contrast, prior to local or global measurement of the image. While morphology sometimes improves the visual appearance of an image, such results are usually incidental to the primary goal, which is to facilitate recognition and/or location of features in images, or to measure them.

There are two principal ways to learn about morphology: by the study of mathematic theory, or experimentally. In this chapter, the authors strongly advocate the latter, as it provides a convenient way to gain intuitive insight into the functions that morphology can perform. In this essay, we shall draw on examples from several different application areas to explain the underlying concepts and to demonstrate the use of morphology in practice. It is important to realise that different template shapes, when used with the same processing rules, can produce widely differing effects. These can often be anticipated by an experienced vision engineer, without recourse to a lot of deep mathematics. Visual examination of an image will often suggest the appropriate means of processing it, at least in broad conceptual terms. However, this needs to be tried and tested thoroughly. An interactive image processor is ideal for this purpose. The engineer’s skill in working with morphological operators is also developed effectively in this way. For this reason, we have provided practical examples that can be repeated by the reader, using the QT system described in ❯ Chaps. 21 and ❯ 41. (Where appropriate, QT command sequences are given below, so that the reader can try these ideas for himself.)

Mathematical Morphology can be traced back to 1964, with the collaboration between Georges Matheron and Jean Serra in Paris, France. Morphological image processing techniques were formalised and named by Serra's book in 1982 [1]. Many of them were known earlier than that and were invented independently by other workers. One of the authors (BGB) independently 'discovered' several binary and grey-scale morphological operators in the 1970s, while he was experimenting with an early precursor of QT. This is mentioned here simply to emphasise the value of experimentation using an interactive processing system.

Morphology is defined in formal mathematical terms using set-theoretic notation. For example, the erosion of a binary image may be expressed thus:

$$A \ \Theta \ B = \left\{ z \in E | B_{\approx} \subseteq A \right\}$$

The notation is defined in the appendix to this chapter [2]. The precise details need not concern us for the moment. The point is that an equation like this does not yield the immediate intuitive insight that other less formal methods of expression can do. To demonstrate this, let us now describe erosion verbally: *it is the result of trying to fit a defined binary template to a given binary image. The output image consists of the set of pixels indicating where this is possible.*

There are four obvious ways to describe erosion, dilation and the other functions derived from them:

1. Mathematically
2. Verbally
3. Graphically
4. Program listing

The concept of erosion can be explained in a few words or a simple diagram. The authors of this chapter contend that morphology consists of fundamentally *very* simple concepts and that the powerful machinery of set theory is unnecessary for our present purposes. Indeed, they maintain that set-theoretic notation actually obscures the underlying simplicity. For this reason, they have chosen to rely on a graphical/pictorial approach to define and describe the various morphological operations. Notwithstanding this, formal definitions are given in the appendix to this chapter.

By definition, morphological operations are *neighbourhood operations*. That is, they base their calculations on more than one image pixel at a time. In keeping with the established terminology of this field, we will refer to the shapes used in these neighbourhood operations as *Structuring Elements*, often abbreviated to *SEs*. However, this term contributes very little to understanding what they actually do. Alternative names for an SE are *template,* or *mask.* However, in keeping with customary practice, the term structuring element will be used throughout the remainder of this chapter.

19.2 Definitions: Images and Structuring Elements

This chapter will consider primarily two types of images: binary and grey-scale. Initially, we will concentrate on the former and will extend those ideas later to describe grey-scale morphology.

Binary images have two intensity levels, usually called *black* and *white.* Black pixels will be represented here by the numerical value 0 when arithmetic is being performed, or the Boolean value 0 when logic operations are being performed. White pixels will be represented by Boolean 1 when logic operations are being performed. Sometimes, arithmetic is to be performed, in

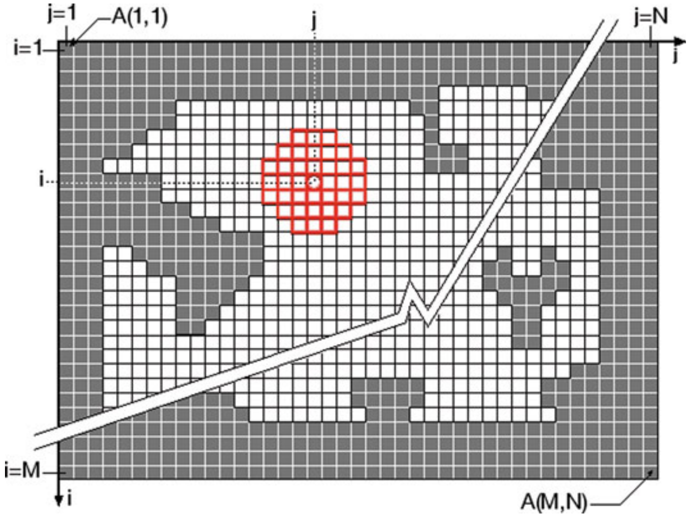


Fig. 19.1

Notation for binary images. The region outlined in red represents a 'circle' structuring element centred on the point (*i,j*), which is called the reference (or centre) pixel for that SE. The result of the calculation based on his neighborhood defines the new value for the centre pixel

which case, the numerical value 1 may be used to represent white. (In some image processing systems, any non-zero value denotes 'binary white'. In 8-bit systems, 255 is sometimes used instead. The context should make clear which representation is being discussed.) For convenience of discussion, the black pixels will usually be called 'background' pixels and the white pixels will usually be called 'object' pixels. Binary images will be denoted as arrays of binary numbers of the form *A(i,j), i* = 1,2,. . .,*M*; *j* = 1,2,. . .,*N* (*Fig. 19.1*). Each square is 1 pixel.

Grey-scale images, typically with 256 intensity levels (8-bit images). In these, the level 0 will be referred to as *black* and the level 255 as *white*. All other levels will be referred to as *grey*, although subdivisions of the grey range (e.g. light grey, dark grey) are sometimes useful. In addition, there is sometimes a need for 16-bit images (to avoid rounding errors when summations over regions with many pixels are involved, for example). When the need for such images arises, this will be noted. Grey-scale images will be denoted as arrays of *positive integers* of the form *A(i,j), i* = 1,2,. . .,*M*; *j* = 1,2,. . .,*N* (*Fig. 19.2*).

A binary SE *specifies a neighbourhood* relative to a *reference* pixel, usually called the *centre pixel*. This is usually, but not necessarily, located close to the centre of the neighbourhood. However, it is not necessary for the centre pixel to be included in the SE. The SEs in *Figs. 19.3* and *19.4* have various numbers of pixels, ranging from 1 to 69. These diagrams illustrate one of the conventions for representing SEs: small white squares represent pixels that are part of the SE and small grey or black squares represent pixels that are *not* part of the SE. The SEs in *Fig. 19.4* are shown using a different convention: non-SE pixels are omitted completely and the reference pixel is marked with a circle. Many SEs define neighbourhoods in which the pixels are arranged contiguously, symmetrically, and forming a convex shape. However, none of these is necessary; the SE may be disconnected, asymmetric, concave, and have

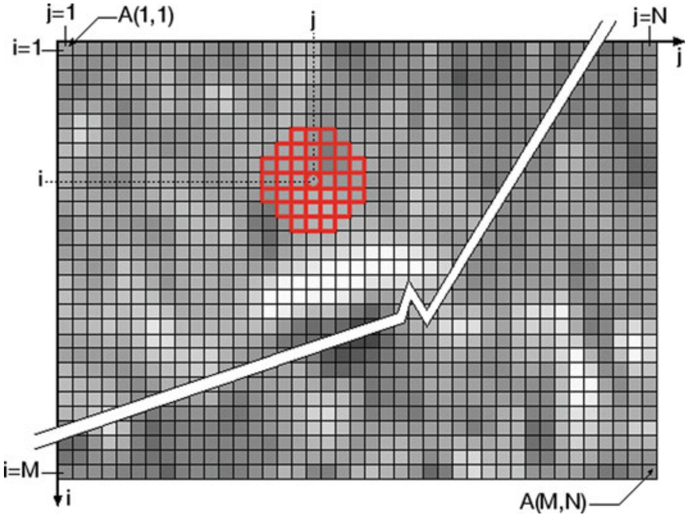


Fig. 19.2

Notation for grey-scale images. The region outlined in red represents a ‘circle’ structuring element centred on the point (*i,j*), which is called the reference (or *centre*) pixel for that SE. The result of the calculation based on his neighborhood defines the new value for the centre pixel

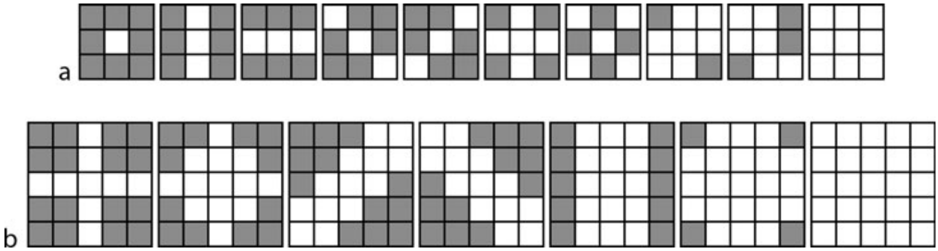


Fig. 19.3

Sample structuring elements (a) 3 × 3 pixel SEs. The left-most SE, the *identity operator*. Its sole use is for testing erosion or dilation programs; the input image remains unchanged. (b) 5 × 5 pixel SEs

‘holes’ such as a disc (O), or 8. *Figure 19.4* shows the rather limited range of SEs available in Photoshop™.

Another important concept is that of *connectedness* in digital images (see *Fig. 19.5*). Apart from pixels around the border of the image, each one has four *4-neighbours,* adjacent to it and arranged around it in the four principal directions: above, below, left, and right. All points within a blob that can be traversed by moving from pixel to pixel by visiting only 4-neighbours are said to be *4-connected.* Similarly, we can define *8-neighbours* and *8-connectivity.* Notice that the term 8-neighbour refers to all 8 pixels surrounding the central pixel’s 3 × 3 pixel neighbourhood. Notice further that an 8-connected arc does not divide the image into two distinct parts, in the way that a line normally segments a plane. Two 8-connected arcs can cross one another, creating the so-called *Connectivity Paradox.*

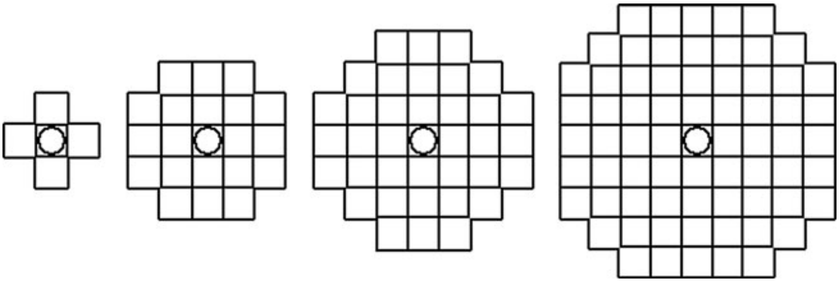


Fig. 19.4

SEs used by Photoshop® (CS3 and CS4) for erosion ('Contraction. . .') or dilation (*Expand*. . .). Photoshop® provides the four structuring elements shown here

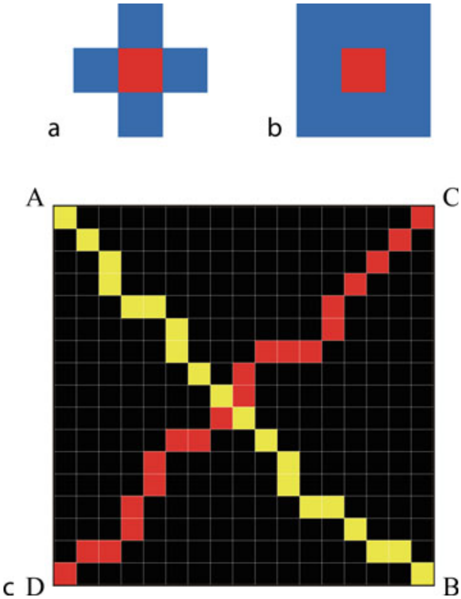


Fig. 19.5

Connectedness in digital images. (a) 4-neighbours (*blue*) of the red pixel and are said to be 4-connected to it. (b) 8-neighbours (*blue*) of the red pixel are 8-connected to it. (c) Connectivity paradox. A and B are 8-connected (*yellow line*), while C and D are also 8-connected to each other (*red line*). Neither of these lines divides the grid into separate regions (i.e., that are not 8-connected to each other)

19.2.1 General Morphological Procedure

Morphological image processing (both binary and grey-scale) proceeds as follows:

1. Place the reference pixel of the SE at some position on the input image. Computationally, the process usually begins with the pixel in the first row and first column of the input image, and proceeds in raster-scan order. However, any convenient scan sequence could be used.

2. Let the number of pixels in the SE be denoted by K. (For example, in the third SE shown in ➋ *Fig. 19.4*, $K = 37$.) Extract from the input image the K values of the pixels in the neighbourhood defined by the SE. Denote these by $\{a_1, a_2, \ldots, a_K\}$.
3. From these K values and the rules to be defined in later sections, calculate the value to be written into the output image buffer at that location. In general terms, this may be represented thus:

$$\text{Output\_Image}(i, j) = \mathbf{F}\{a_1, a_2, \ldots, a_K\}$$

   where $\mathbf{F}$ is a logical function specifying the operation being performed.
4. Repeat for all other positions in the input image (Depending on the application requirements, some of the input image pixel positions might be skipped, thereby saving computation time).

19.3 Binary Erosion

Consider a binary image containing both black pixels ('background') and white pixels ('object'). Binary erosion is a process by which some of the white pixels in the image are 'removed' i.e., converted to black pixels. The choice of which white pixels to keep and which white pixels to erode is determined by the SE. In terms of a physical analogy, the binary erosion operation finds all the places where the SE 'fits' completely inside a blob in the image (➋ *Fig. 19.6*). The SE is said to 'fit' at a certain position if the input image has 'white' pixels at all positions in the neighbourhood defined by the SE. An image may have one or many blobs. The erosion operation is position-invariant, in that it treats all blobs in the same way, irrespective of where they are.

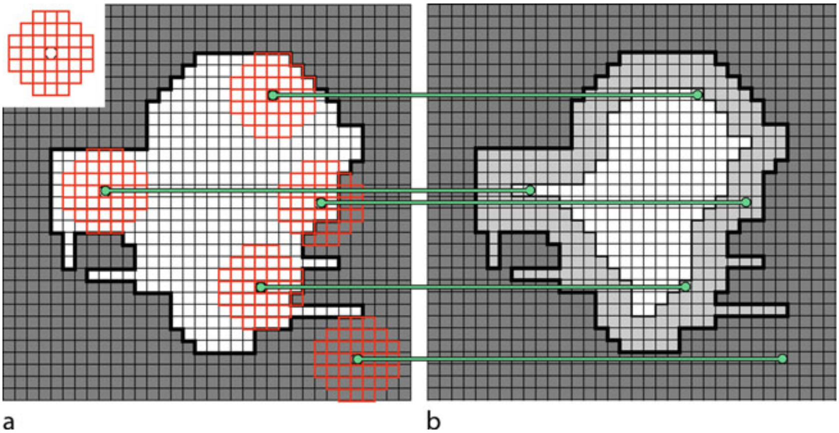


Fig. 19.6

Erosion using a 'circle' SE. (a) Original image. Inset: SE consisting of a 7 × 7 pixel representation of a circle. (b) Result of erosion (*white blob*). The grey pixels were originally white and are set to black by erosion. The SE is shown in several positions and the green lines indicate the corresponding pixels affected

19.3.1 Mathematical Definition

For binary erosion, the function $\mathbf{F}$ is the *logical* AND (&) of the K binary pixel values, which are, by definition, either Boolean 0 (black) or Boolean 1 (white). Thus, for binary erosion,

$$\text{Output\_Image}(i, j) = \{a_1 \& a_2 \& \ldots \& a_K\}.$$

This is equivalent to saying that the centre pixel is white if and only if all of the pixels in the neighbourhood are white.

19.3.2 Examples of Erosion

Erosion is illustrated with reference to a circular SE of diameter 7 pixels applied to a simple blob (➲ *Fig. 19.6*). This shows two of the many positions at which the SE 'fits'. That is, all image pixels within the neighbourhood of (*i,j*) defined by the SE are white. In addition, three SE neighbourhoods (shown in red) at which the SE does *not* fit, are also shown. At these points the output value is Boolean 0.

Notice the following points which are generally valid for erosion:

- Thin 'hairs' are removed.
- Very small blobs are removed (not shown in ➲ *Fig. 19.6*).
- Larger blobs become smaller.

We have established the general concept of erosion and an understanding of its actions without recourse to abstruse mathematics. The precise function of erosion depends upon the size and shape of the SE, as well as the particular blob features: size, shape, and orientation. This will be illustrated several times later.

19.4 Binary Dilation

Binary dilation is a process by which white regions in the input image are augmented by adding white pixels around the edges, or in holes ('lakes') in the interiors of blobs. That is, some of the black background pixels are converted to white object pixels. The choice of which black pixels to leave unchanged and which to change to white pixels is determined by the particular SE being applied. ➲ *Figure 19.7* shows dilation applied to the same simple blob that we previously used for erosion.

19.4.1 Mathematical Definition

The function $\mathbf{F}$ used to define binary dilation is the logical OR of the K binary pixel values covered by the SE. Thus, for binary dilation,

$$\text{Output\_Image}(i, j) = \{a_1 \textbf{ OR } a_2 \textbf{ OR} \ldots \textbf{OR } a_K\}$$

This is equivalent to saying that the output is *white* if at least one of the pixels in the neighbourhood covered by the SE is *white*.

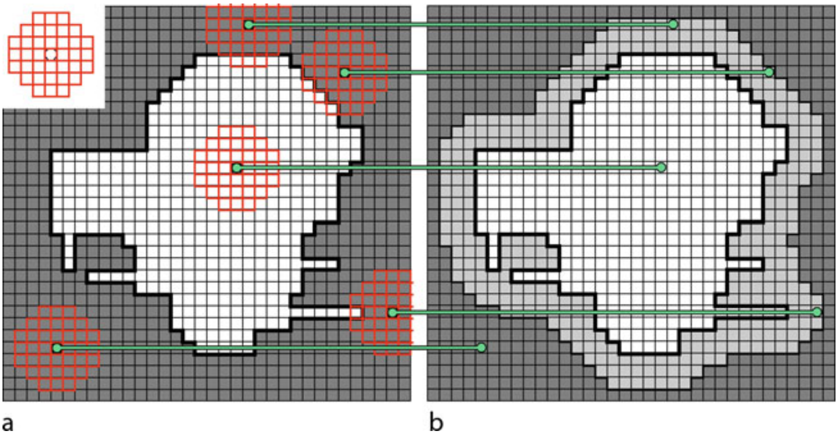


Fig. 19.7

Dilation using a 'circle' SE. (a) Original image. Inset: SE consisting of a 7 × 7 pixel representation of a circle. (b) Result of dilation. The grey pixels were originally black and are set to white by dilation. In two instances shown here, the SE overlaps the border of the image. This often gives rise to anomalies, called edge *effects*, which will be described later

19.4.2 Duality

Dilation is, in some ways, the 'opposite' of erosion, in that it *expands* white areas, while erosion reduces them. However, dilation is *not* the true *inverse* of erosion; dilation followed by erosion does not restore the image to its original state and vice versa. Morphological operations, like most other neighbourhood operations, in general, do not have inverses. This lack of an inverse operation is, in fact, a particular benefit of morphological operations and we shall see later how we can take advantage of this.

Ignoring 'edge effects'(defined below), *dilating* the white pixels of the *object* is identical to *eroding* the black pixels of the *background*, and vice versa. This relationship is referred to as the *duality* between erosion and dilation (see Supplement). Hence, duality is a particular case of *de Morgan's Rule.* (Recall that erosion is defined in terms of ANDing and dilation using ORing.) Duality allows us to define an *indirect* method of performing dilation:

1. Negate the image.
2. Erode the now-white pixels (which were originally the background pixels) with the appropriate SE.
3. Negate the output image, so that the original black-white relationship between object pixels and background pixels is restored.

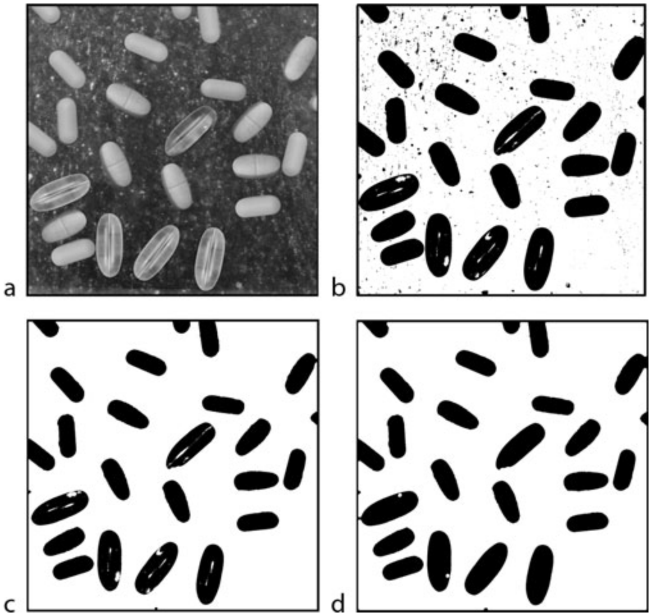
By simply interchanging the representations of black and white (equating black with Boolean 1 and Boolean 0 with white), we see that there is no need to incur any time penalty for performing dilation by the inverse method. In QT, *gbd(SE)* produces exactly the same result as [*neg, gbe(SE), neg*], for any structuring element, (*SE*), which of course is slower.

Dilation is useful for removing black streaks and spots. It also makes large blobs even larger. This is to be expected from duality. (Recall that erosion removes white streaks and spots and makes large white blobs smaller.)

The definition of duality given above holds for structuring elements that are rotationally symmetrical about some line. If this is not the case, the SE must be flipped, by rotating it through 180°. That is, dilation with structuring element $S = \{[i_1,j_1],[i_2,j_2], [i_3,j_3],\ldots\}$ is the dual of erosion with structuring element S', where $S' = \{[-i_1,-j_1], [-i_2,-j_2], [-i_3,-j_3], \ldots\}$ and vice versa.

19.4.3 Examples of Erosion and Dilation

➲ *Figure 19.8* shows a grey-scale image of tablets and capsules lying on a mottled background. Suppose the goal is to isolate and count the pills of the various types. Since all of the pills are larger than the speckles of the background, dilation with an SE *smaller* than the pills but *larger* than the speckles can be used to discriminate between them. It is not immediately apparent from ➲ *Fig. 19.8c* that the pills have become slightly smaller. Shrinkage is not important if our task is to count, rather than measure pills. ➲ *Figure 19.8d* shows that dilation can have a beneficial effect by removing white spots within the pills. Erosion and dilation are illustrated again in ➲ *Fig. 19.9*, where the change of blob size is more obvious.



◘ Fig. 19.8

Erosion and dilation. (a) Original image, pills lying on a mottled background. (b) Thresholded and inverted binary image. (c) Dilation result. Notice the absence of small black spots, except around the edges of the image. (d) Erosion of the previous image. Notice that there are fewer white blobs within the pill borders

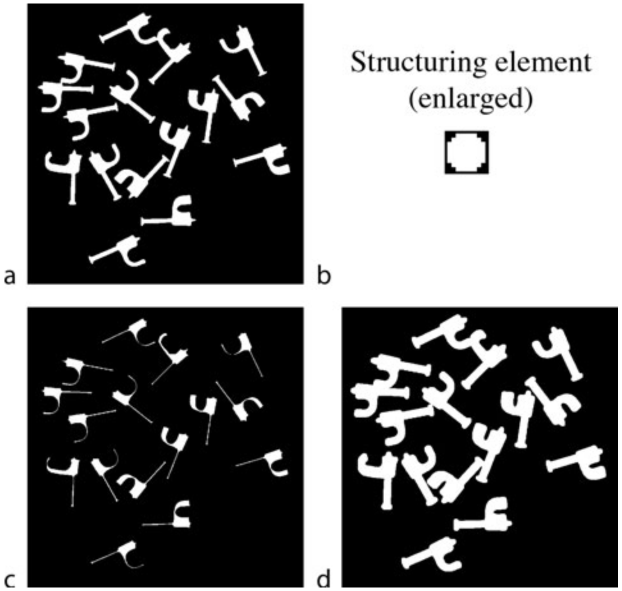


Fig. 19.9
Another example of erosion and dilation. (a) Original image, cable fasteners. (b) Structuring element, 9 × 9 'circle'. (c) Erosion. (d) Dilation

19.5 Opening and Closing

Opening and closing are combinations of erosion and dilation. The reader will already appreciate that erosion *reduces* the sizes of white regions in a binary image, while dilation does the opposite. With this in mind, it is reasonable to ask whether eroding an image and the dilating the result returns it to its original state. The answer is '*No!*' but the effects of combining them can be very useful.

The process of eroding an image, and then dilating the result, with the same SE, is called *opening*, and is defined using fairly obvious terminology:

$$\text{Output\_Image}(i,j) = \text{OPEN}_{\text{SE}}[\text{A}(i,j)] = \text{DILATE}_{\text{SE}}[\text{ERODE}_{\text{SE}}\{\text{A}(i,j)\}]$$

A*(i,j)* is the input image and the subscript SE denotes the particular structuring element being used, which is the same for both erosion and dilation. Opening is a very useful operation, particularly for eliminating noise in the form of small white spots, or narrow streaks, while leaving the rest of the image almost unchanged. *Figure 19.10* shows the result of opening on a simple test images. More practical results will be seen from the examples presented later.

Similarly, dilating an image and then eroding the result with the same SE, is called *closing*, and is defined thus:

$$\text{Output\_Image}(i,j) = \text{CLOSE}_{\text{SE}}[\text{A}(i,j)] = \text{ERODE}_{\text{SE}}[\text{DILATE}_{\text{SE}}\{\text{A}(i,j)\}]$$

Closing is useful for filling small lakes and eliminating thin black streaks. Again, the remainder of the image is changed very little.

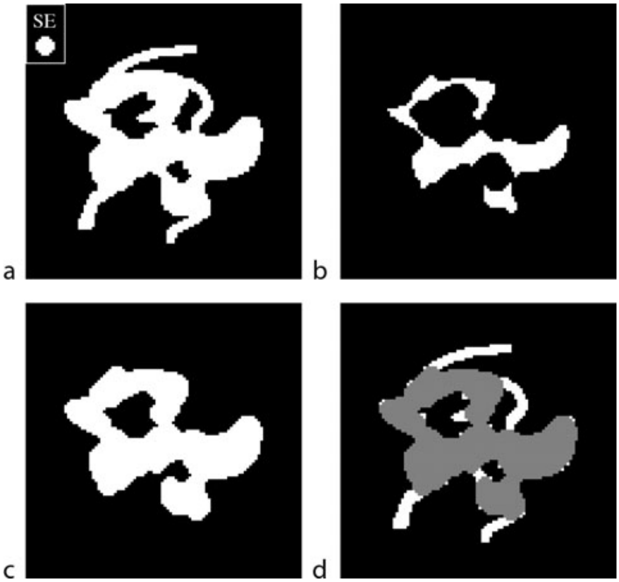


Fig. 19.10

Opening. (a) Original image. Inset: structuring element. (b) Erosion. (c) Opening. This is equivalent to the dilation of the previous image with the same structuring element. (d) Differences (*white*) between opening and the original image (*grey*). Notice that thin white streaks have been eliminated, convex corners have been smoothed slightly and a narrow 'neck' has been broken. The last mentioned effect gives this operation its name

Figure 19.11 shows the result of closing on an image containing a simple hand-drawn blob.

Figure 19.12 shows the results of applying all four binary morphological operators discussed so far, to a rather more sophisticated test image, using three different structuring elements. This test image consists of a wedge and several solid circles of different sizes, in positive and negative contrast. The reader is urged to study these results carefully, as they provide the key to understanding binary morphology. This *Fig. 19.12* also includes illustrations of the effects of two further functions:

- *Opening*. That is, erosion followed by dilation, using the same SE.
- *Closing*. That is, dilation followed by erosion, using the same SE.

19.6 Edge Effects

Blobs touching the edge of the image can result in unexpected outcomes when neighbourhood operations are performed. (This is true for other neighbourhood operations, including convolution, as well. See Chap. 15.) This follows from the fact that, when the SE is centered on a pixel on the edge of an image, part of it extends beyond the image, leaving the operation undefined. Assumptions are always built into the hardware or software implementing morphological functions that determine what will happen at the edges. Different assumptions give different results – and are always totally arbitrary!

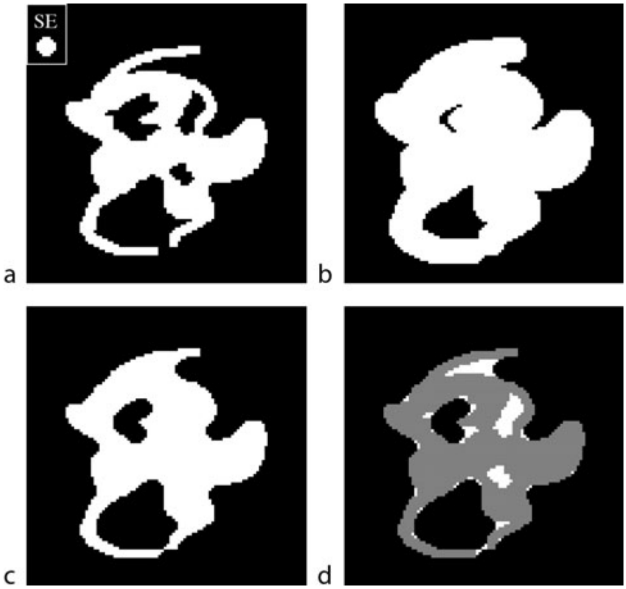


Fig. 19.11
Closing. (a) Original image. Inset: structuring element. (b) Dilation. (c) Closing. This is equivalent to the erosion of the previous image with the same structuring element. (d) Differences (white) between closing and the original image (*grey*). Notice that small black lakes have been filled, concave corners have been smoothed slightly and a narrow black 'channel' has been bridged. The last mentioned effect gives this operation its name

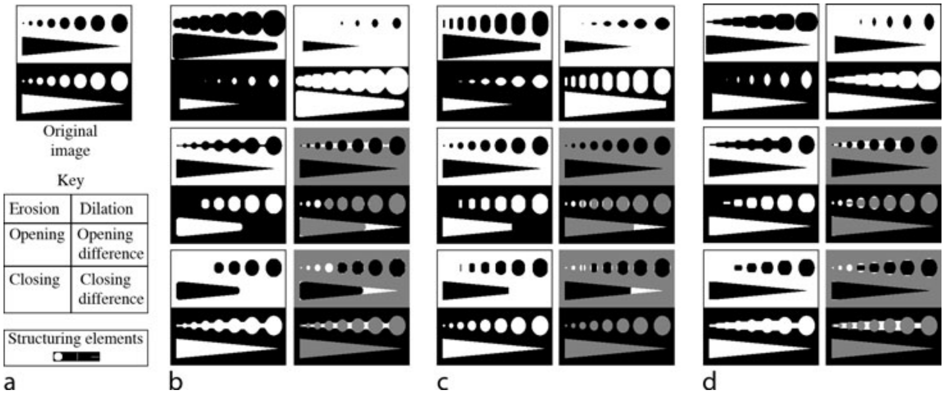


Fig. 19.12
Erosion, dilation, opening and closing applied to a simple test image. (a) Original image, the key to interpreting the results and the structuring elements. (b) Structuring element was a circle. (c) Structuring element was a vertical line. (d) Structuring element was a horizontal line

Algorithms for binary erosion, dilation, opening and closing usually handle this by assuming that there are additional 'phantom' rows and columns of pixels surrounding the image. This allows the '*inner loop*' code to run faster because it does not have to check for and handle special cases each time it moves to a new pixel. For erosion, these phantom pixels are usually set to black and therefore not part of any blobs. However, it means that all pixels are set to black, if their neighbours, as defined by the SE, lie outside the edge of the image. For a similar reason, it is customary when implementing dilation to set the phantom pixels to white. ➲ *Figure 19.13* illustrates this using erosion with a 3 × 3 4-connected SE on a very simple blob. ➲ *Figure 19.14* shows how the boundary of an object touching the image border can be distorted by edge effects.

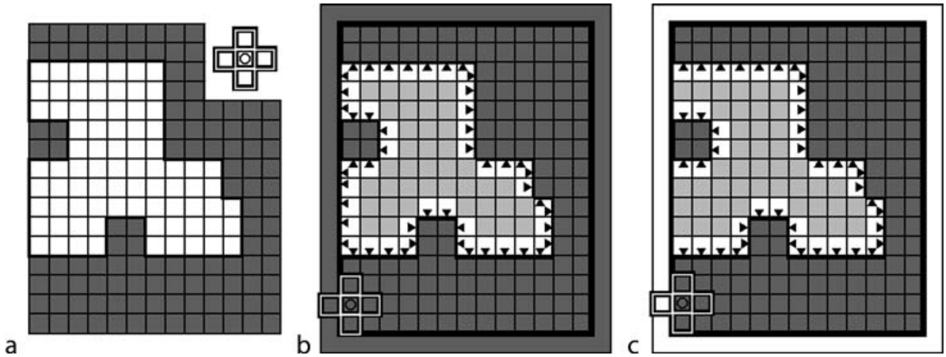


Fig. 19.13
Edge effects. (a) Original blob. (b) Erosion with black phantoms. The small arrow-heads indicate which pixels will be eroded to black, with the arrows pointing to the black pixel, or pixels, responsible for eroding them. (c) Erosion with white phantoms

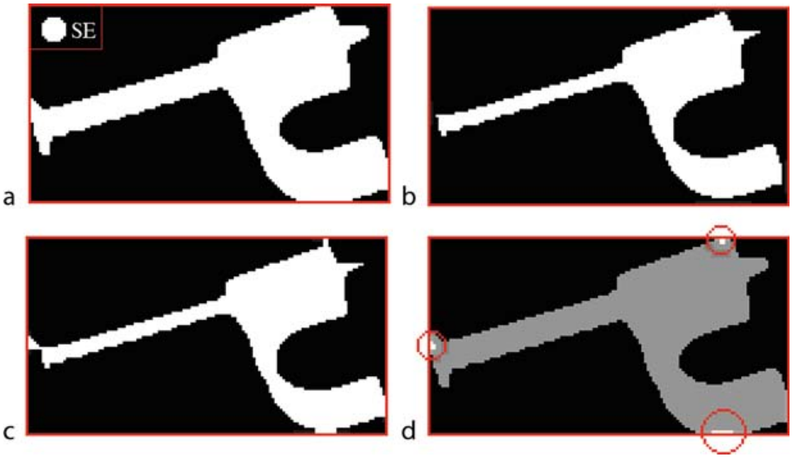


Fig. 19.14
Edge effects. (a) Original blob. (b) Erosion with black phantoms. (c) Erosion with white phantoms. (d) Differences due to change of phantoms. These are very small *white blobs circled* in *red*

Another approach to handling edge effects is simply to ignore all pixels close to the image border. This is really the only 'honest' approach, as all others are perfectly arbitrary. If the results at the edges of the image are not what is wanted, the assumption must be changed.

19.7 Extended Functions

Opening and closing are combinations of erosion and dilation. Since the last mentioned is the dual of erosion, we see that that all four of the morphological operators described so far can be expressed in terms of a single concept: erosion (or its dual, dilation). Clearly, other combinations are possible. For example, we could compare the images created by erosion (or dilation) with different structuring elements. Another possibility is to combine the four basic procedures with subtraction of the original image. All of these combinations can produce interesting and useful effects. In this section, we explore some of the options, wherever possible illustrating the operation using images drawn from practical applications. Other options are available, such as applying erosion recursively, with clever rules for deciding when to stop processing, locally and globally. Morphology is effectively a family of non-linear neighbourhood operations but it does not provide a complete tool-box for processing images. Combining morphological operations with members of the large range of non-morphological functions cannot be described effectively for the general case. The best advice we can provide is learn about morphology first, look at the image carefully, decide what operation to perform, then apply it and evaluate the results by eye and/or computer. On its own, a single morphological operator is rarely sufficient. The sequences of image-processing operations required for most practical applications do not fit neatly into simple categories. For this reason, the following examples do not follow a simple pattern.

Selecting linear features at a given angle. *Figure 19.15* shows how a straight-line structuring element can be used to select linear features within a narrow range of angles. The precision obtained depends on the length of the SE.

Checking the periodicity of a grid. *Figure 19.16* shows an array of spots. This image was generated using a standard lens which introduces barrel distortion. Since the authors wanted to exhibit the power of morphology, this warping effect was deliberately produced. (To obtain a distortion-free image, a telecentric lens should be used.) The SE was derived by sampling the original mesh-like structure after a small amount of erosion had been applied. (Actually, thinning was used but this explanation will suffice until we meet that subject later.) Generating the SE by sampling the image, optionally applying some preprocessing first, is an important adjunct to morphology. (This point will be illustrated again later, when we consider optical character recognition.) The initial erosion (*Fig. 19.16c*) leaves an array of disconnected spots, which is difficult to measure. Dilation and closing were then applied to join them together. The resulting blobs indicate where the local periodicity is 'acceptable' (*Fig. 19.16e*).

Deriving a disjoint SE from the image. *Figure 19.17* shows an example of erosion using a structuring element that consists of four separate parts. There is no theoretical limit on the number of distinct parts in an SE.

Using the largest internal circle to choose the SE. The largest internal circle operator [QT: *lic*] draws the biggest circle that can be drawn within the white regions of a binary image (*Fig. 19.18b*). It can also be applied recursively (*Fig. 19.18c*) [QT: *mic*], or to the background to measure lake diameter (*Fig. 19.18d*). Clearly, these circles only give an approximate guide to where SEs of different sizes will fit.

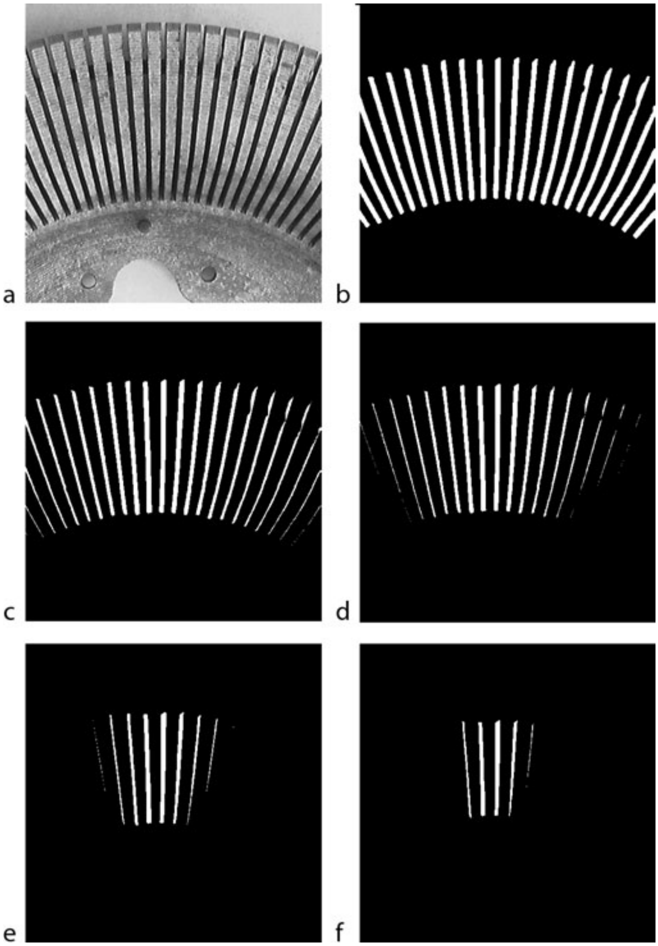


Fig. 19.15

Erosion with straight-line structuring elements of varying lengths selects linear features with variable precision. (a) Original grey-scale image (This is a die-cast alloy component, from a mechanical typewriter). (b) Binary image. Produced using thresholding and morphology, to reduce noise. (c) SE was a vertical straight line of length 15 pixels. (d) SE was a line of length 25 pixels. (e) SE was a line of length 50 pixels. (f) SE was a line of length 75 pixels

Noise reduction (➲ *Fig. 19.19*) Salt-and-pepper noise consists of white and black spots contaminating the background and foreground of a binary image, respectively. It results in bit reversal in a small proportion of the pixels. In general, noise arises in binary images as a result of a variety of physical effects, including camera noise, electrical noise, quantisation errors and actual dirt on the object being viewed. Morphology can selectively reduce the number of either white or black spots, or both. ➲ *Table 19.1* lists the general principles for selecting the method of noise reduction. For the sake of comparison, ➲ *Fig. 19.19* also shows results for other methods of filtering to remove noise. Simply removing all blobs that have a small area is often very effective but is rather slow compared to morphological operations. Taking a majority

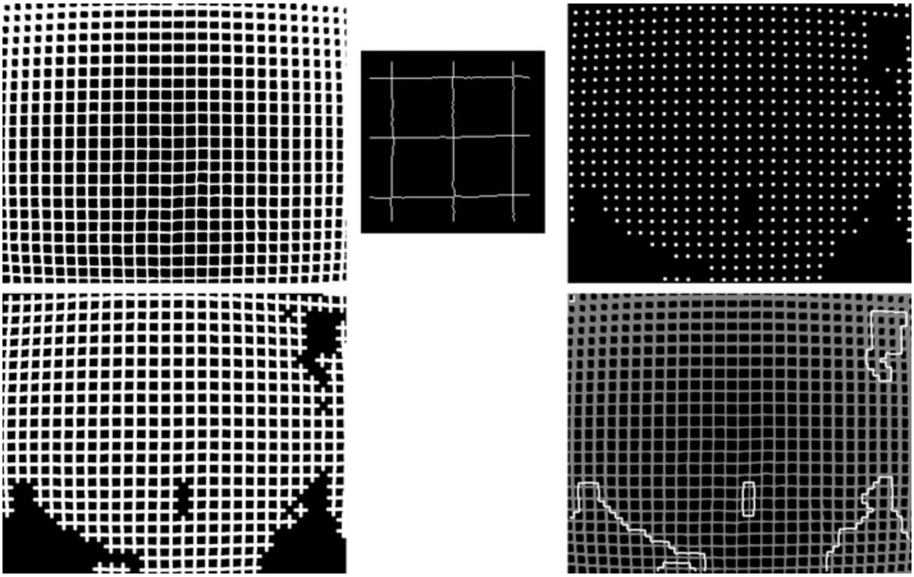


Fig. 19.16
Erosion with an SE derived from the image: checking repeated patterns. (a) Original binary image, substrate for an automobile catalytic converter. The barrel distortion is due to the lens. The warping of the geometry was deliberately tolerated here, so that we could better demonstrate the image filtering. (b) Structuring element, enlarged for clarity. This was derived from the central region of the image, after thinning. (c) Result of erosion. Each ‘hit’ is represented by a spot of uniform size. (d) Dilation using a vertical cross (+) SE ‘to join the dots’. (e) Closing of the previous image using a solid square SE produced a set of blobs, whose outlines have been superimposed on the original

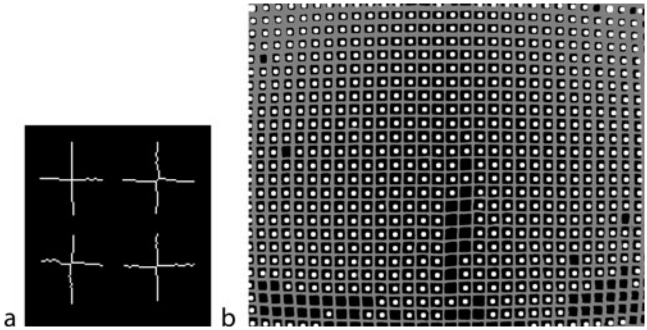


Fig. 19.17
Using a multipart SE. (a) Structuring element, selected manually from the thinned version of *Fig. 19.17*. **(b) Result of erosion. Notice the ‘holes’ are detected here, whereas the processing described in the previous figure detected the lattice-wall junctions**

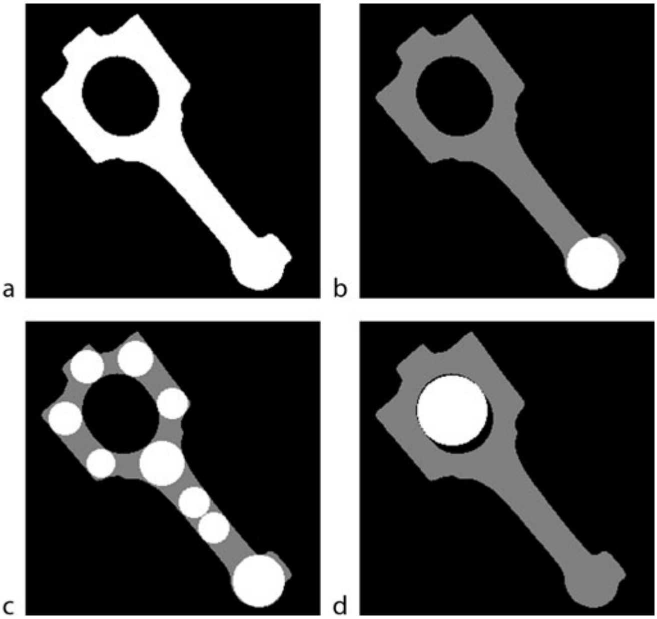


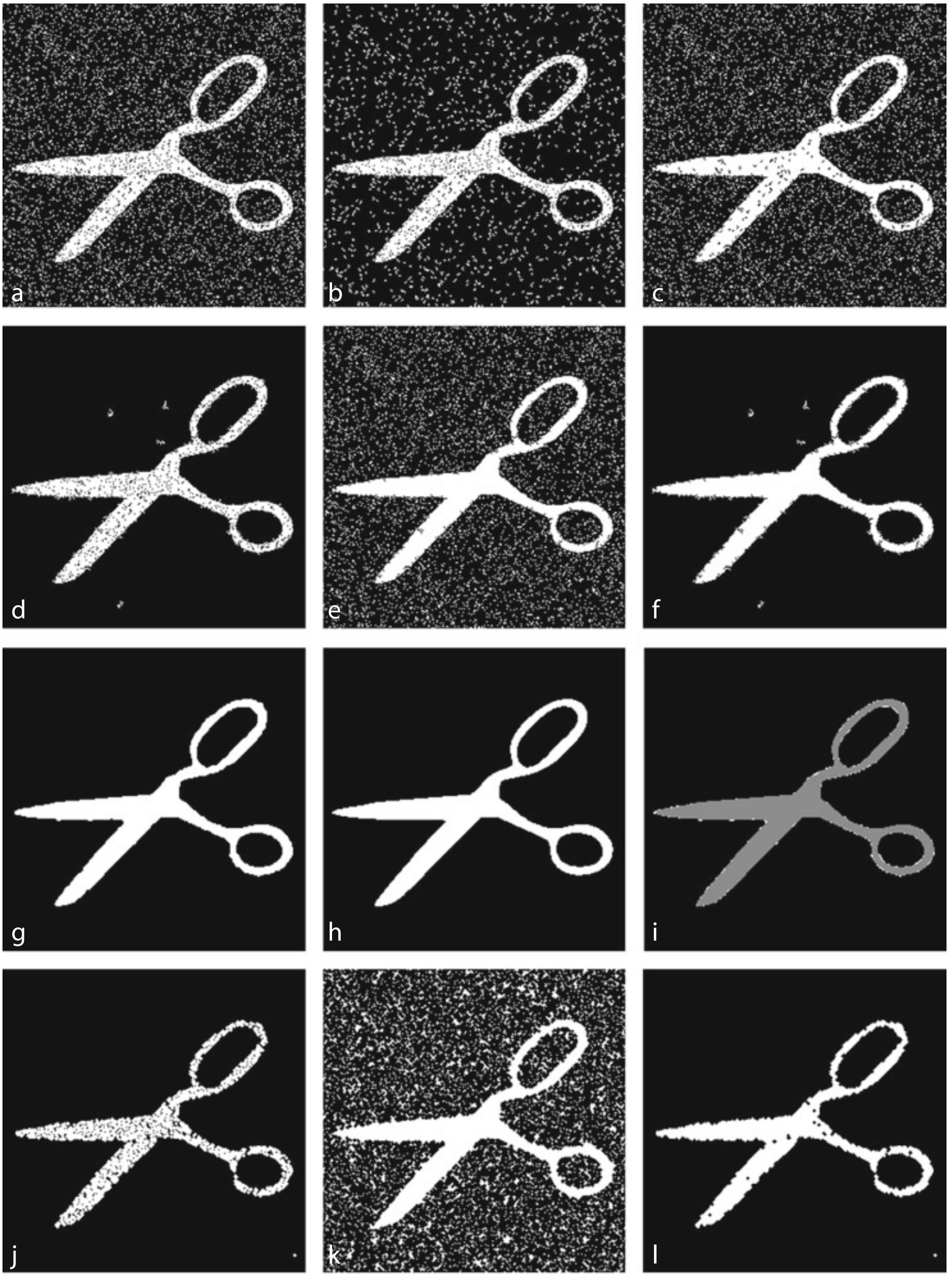
Fig. 19.18

The largest internal circle can help in choosing appropriate structuring elements for morphological operations. (a) Original binary image, automobile conrod (connecting rod). (b) Largest internal circle superimposed on the original. [QT: *lic, adi*] (c) The largest circle was found, the exclusive OR was applied. This process was repeated recursively a total of nine times. (d) Largest internal circle fitted to the lake. [QT: *blf, exr, lic*]

vote of the pixels within the neighbourhood covered by the SE provides an effective way to eliminate both black and white spots. [QT: *maj*] The majority voting function can be performed once, several times as the user chooses, or repeated recursively, until the image no longer changes. Of course, the execution time cannot be predicted for the last option, because it is data driven.

Binary edges. Erosion followed by exclusive OR with the original image, draws the edge contours (➋ *Fig. 19.20*). This operation is given the name *Top-hat.* The size and shape of the structuring element determines the exact form of the edge contour. Normally, it is desirable to produce a curve that is 1 pixel wide, in which case a 3 × 3 pixel SE is used. However, it is possible to highlight, say, top edges, or left edges; a linear SE is then used. The vision engineer must also decide whether a 4- or 8-connected curve is needed. Dilation followed by exclusive OR with the original image produces an edge contour. This is called the *Bottom-hat* operation. Top-hat produces contours that lie within the original white area, while bottom-hat places the curve in the background adjacent to the white pixels in the original image.

Searching for specific small-scale patterns. The top-hat and bottom-hat functions are, of course, able to detect any pattern that can be expressed in an SE, whether or not it is contiguous. An obvious extension of this is to look for pixels that have a given *number* of white neighbours (➋ *Fig. 19.21*). Strictly, this function does not look for a single pattern but any one of a number of patterns. Isolated white pixels (i.e. no white 8-neighbours) are detected



■ **Fig. 19.19**
(Continued)

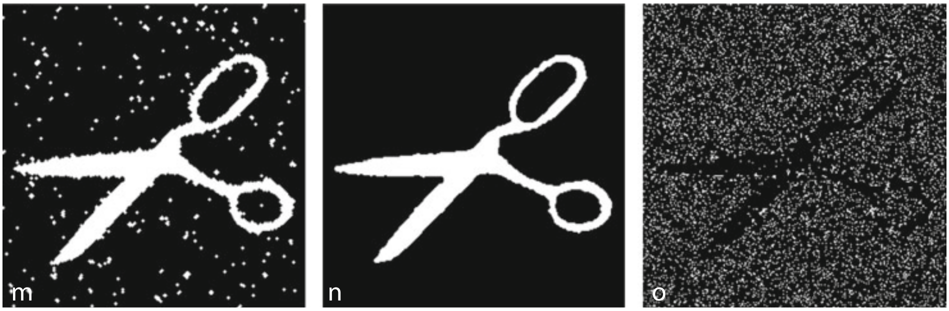


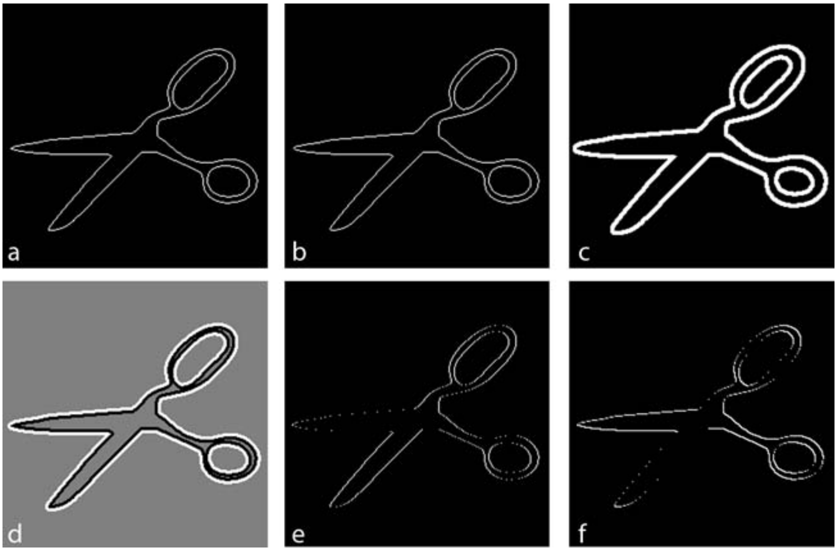
Fig. 19.19

Reducing noise using morphology. Some of the operators mentioned here are described later. They are included here so that the various options can be compared. (a) Original image image, created specially for this purpose by superimposing salt-and-pepper noise on the 'scissors' image. (b) Removing isolated white pixels. [QT: *cln*] (c) Removing isolated black pixels. [QT: *neg*, *cln*, *neg*] (d) Removing small white spots, fewer than 10 pixels. [QT: *kgr*(10)] (e) Removing small black spots, fewer than 10 pixels. [QT: *neg*, *kgr*(10), *neg*] (f) Removing both white and black spots. [QT: *kgr*(10), *neg*, *kgr*(10), *neg*] (g) Majority filter applied to (f) produces a smoother edge. [QT: *kgr*(10), *neg*, *kgr*(10), *neg*, *maj*] (h) Original scissors image, before the artificial addition of noise. (i) Differences between (g) and (h), shown in white. (j) Opening [QT: *x* = *cse*(14, 1), *gbe*(*x*), *gbd*(*x*)] (k) Closing. [QT: x = *cse*(14, 1), *gbd*(*x*), *gbe*(*x*)] (l) Opening followed by closing with the same SE. [QT: *x* = *cse*(14, 1), *gbe*(*x*), *gbd*(*x*), *gbd*(*x*), *gbe*(*x*)] (m) Closing followed by opening with the same SE. [QT: *x* = *cse*(14, 1), *gbd*(*x*), *gbe*(*x*), *gbe*(*x*), *gbd*(*x*)]. (n) Majority function with a 3 × 3 pixel SE applied recursively to the original image. (o) Top-hat. [QT: *x* = *cse*(14, 1), *gbe*(*x*), *gbd*(*x*), *rei*('original'), *exr*]

Table 19.1

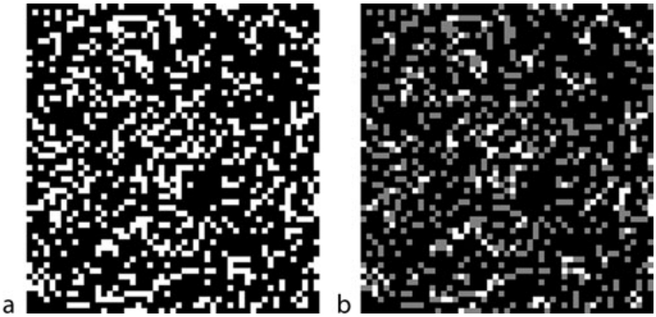
Reducing noise in binary images using morphology

Type of noise	Method	Comments
White spots	Erosion	Large blobs become smaller
White spots	Opening	Large blobs are same size
Black spots	Dilation	Large blobs become larger
Black spots	Closing	Large blobs are same size
White and black spots	Closing, dilation	Experiment
White and black spots	Dilation, closing	Large blobs are same size
White and black spots	Majority vote	See ➲ *Fig. 19.19*
White spots	Biggest blob	Non-morphological function [QT: *big*]
White spots	Retain only those blobs with a large area	Non-morphological function [QT: *kgr*]
Hair, spurs (white streaks with one 'free' end)	Erosion or opening	As white spots
Hair, spurs (white streaks with one 'free' end)	Selective erosion of the end of the spur	[QT: *spr(n), exr* repeated recursively]



■ Fig. 19.20

Binary edges. (a) Top-hat with a 3 × 3 pixel SE. [QT: *ero, exr*] (b) Bottom-hat with a 3 × 3 pixel SE. This contour is slightly larger than that in the previous image. [QT: *dil, exr*] (c) Bottom-hat with a 5 × 5 pixel circle SE. [QT: *rni(49), thr(1), x = cse(14,5), gbd(x), exr*] (d) Comparing edges found by bottom-hat (*white*) and top-hat (*black*). Both contours were generated using a 7 × 7 pixel circle. SE. (e) Top-hat with a horizontal-line SE. [QT: *x = cse(1,3),gbe(x), exr*] (f) Top-hat with a diagonal-line SE (/). [QT: *x = cse(4,3), gbe(x), exr*]



■ Fig. 19.21

Selecting white pixels that have a given number of white 8-neighbours. (a) Original image. (b) White pixels that have exactly three white 8-neighbours. White pixels that have any other number of white 8-neighbours are shaded grey. [QT: *sbc(3)*]

in this way (❯ *Fig. 19.19b*). Later, we shall see that counting white neighbours is used to identify important features on a 'match-stick' figure. In addition, note that the Euler number combines ten 8-neighbourhood counts (❯ Chap. 14) to calculate the number of blobs minus the number of lakes in the image.

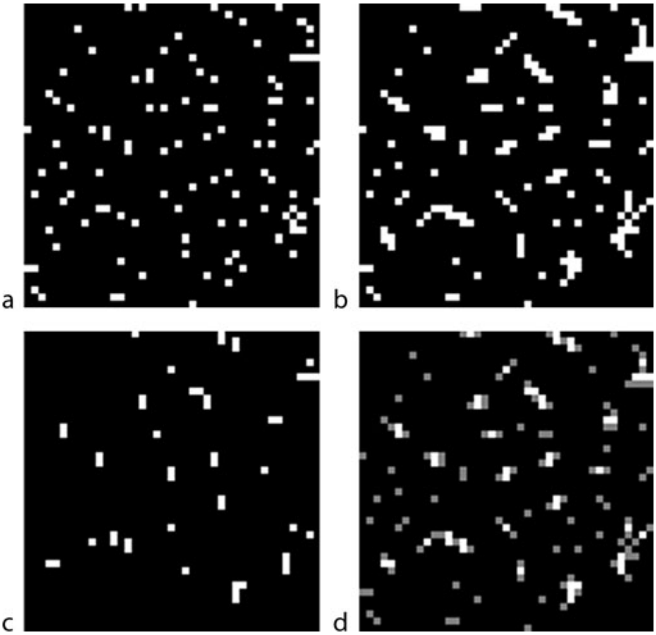


Fig. 19.22

Bridge operator. (a) Original image. (b) Result of bridge. [QT: *mbo*] (c) Exclusive OR of the original image and the result of bridge. [QT: *mbo, exr*] (d) White pixels were added by applying bridge. Original white pixels are shown here as mid-grey

Figure 19.22 shows the results of applying the so-called *Bridge function*. This detects black pixels that have two 8-neighbours that are not 8-connected to each other. Such a pixel is set to white. For example:

$$\begin{matrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{matrix} \qquad \text{becomes} \qquad \begin{matrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{matrix}$$

Similarly

$$\begin{matrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix} \qquad \text{becomes} \qquad \begin{matrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{matrix}$$

The bridge function has limited value in filling or detecting small gaps between white regions.

The morphological operators described so far are all logical. Hence, *all* pixels covered by the SE must be white for erosion to produce a white output. This makes them rather sensitive to noise. For this reason, it is sometimes better to insist that *most* of the neighbours be white. Counting white neighbours can be achieved using linear convolution with unit weights. In QT/MATLAB, the command sequence

```
x = ones(m,n)                      % Create an m*n array of 1s
t = floor(255*p/(m*n))             % Rescale parameter p
```

```
glc(x)                               % General linear convolution with weight matrix x
thr(t)                               % Threshold at level t
```

produces the value *white* (255) for each pixel that has at least *p* white neighbours, including itself. Implementing a 'biased-voting' function like this can be convenient during experimentation but may not lend itself for use in a high-speed target (factory floor) system.

Grass-fire Transform (*Prairie Fire Transform* or *Distance Transform*) The grass-fire transform (of a binary image) is a grey-scale image, in which each white pixel is assigned an intensity indicating the distance from it to the nearest black pixel. Pixels that were black originally remain black (➲ *Fig. 19.23b* and ➲ *d*). To understand how this is related to morphology, consider the process of recursively eroding a blob by removing a 1-pixel wide layer of pixels on each cycle. Points that are far away from the blob boundary will be removed later than those close to it. The output image created by the grass-fire transform indicates how long (i.e., how many erosion steps) it takes to remove each pixel by this 'onion-peeling' process. Clearly, there must be some rules to prevent the erosion continuing when to do so would 'break' the object. These rules act locally, not globally. Local maxima in the grass-fire transform image can be useful indicators of object pose (➲ *Fig. 19.23d*).

Skeletons The skeleton of an object is easier to understand from a picture than a verbal explanation (➲ *Fig. 19.23e*). The recursive erosion process just described needs only a minor modification to generate the skeleton, rather than the grass-fire transform. Comparing these shows that the skeleton runs like a continental divide over the mountain range defined by the grass-fire transform. (Think of intensity as defining the height of each point in a 3D representation of the grass-fire transform image.) Note that the skeleton preserves the Euler number of the original image. Since they are morphological functions, the grass-fire transform and skeletonisation are applied to every blob in the image.

Two skeleton features are important: joints (➲ *Fig. 19.23f*) and limb ends (➲ *Fig. 19.23g*). These are both detected by counting white neighbours, as explained above. The tips of a skeleton's limbs, called 'spurs', may be removed by recursively removing the limb ends. Notice that all limb ends in the skeleton image are pruned pixel-by-pixel, so all spurs are removed at the same time.

Shrinking Imagine trying to place a robot gripper to pick up a horse-shoe. The centroid will not suffice to calculate the robot pose, since the gripper will try to grasp air, not metal. In this kind of situation, it is useful to know the address of just one pixel within each blob that is close to its centre. Shrinking is a useful function that does just this, provided there are no lakes (➲ *Fig. 19.23i*). A binary object with lakes shrinks to a series of inter-connected rings. Each ring is halfway between a lake and the outer boundary nearby (➲ *Fig. 19.23j*).

Thinning The skeletonisation process involves repeated erosion, with rules devised to stop the figure from breaking apart. Thinning is the same process but terminated earlier, usually by specifying beforehand how many erosion cycles are to be performed. Partial, or incomplete thinning, is shown in ➲ *Fig. 19.23k*.

Thinning, like the grass-fire transform and skeletonisation, depends upon the concept of distance. This can be measured in many ways, the usual one being Euclidean distance. To improve computation speed, it is sometimes expedient to use other distance measures and this can sometimes produce unexpected results (➲ *Fig. 19.23l*).

Removing 'Hairs' and 'Spurs' A 'hair' is thin white streak. It may be attached at one or both ends to a large blob, or not attached at all. A 'spur' is a special case of a hair; it is attached at one end only. Hairs can be removed from an image by opening. Similarly, thin black streaks

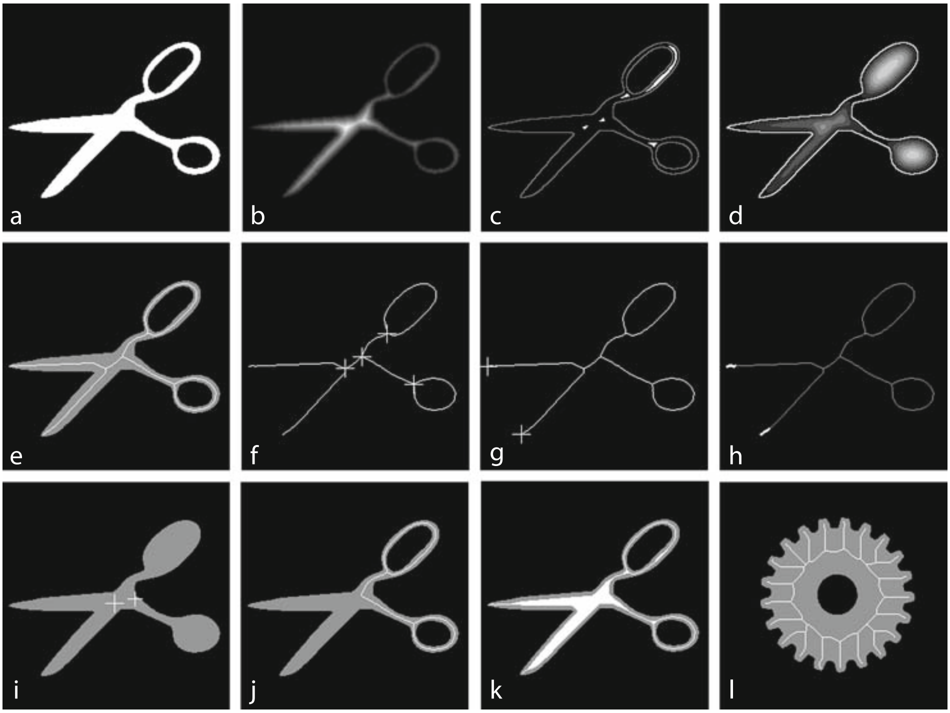


Fig. 19.23

Grass-fire, skeleton and thinning. (a) Original image. (b) Grass-fire transform after contrast enhancement. [QT: *gft, enc*] (c) Local maxima in the grassfire transform, detecte [QT: *gft, emx*] (d) The lakes have been filled in the original image. The grass-fire transform was then applied, followed by contrast enhancement and posterising. The outline after filling lakes is superimposed for illustrative purposes. [QT: *blf, bed, wri, gft, enc, sca(3), rei, mxi*] (e) Skeleton superimposed on the original scissors image. [QT: *thn*] (f) Skeletons joints [QT: *sbc(3)*. Function *jnt* incorporates skeletonisation.] (g) Skeleton limb ends. [QT: *sbc(1)* Function *lmb* incorporates skeletonisation.] (h) By repeatedly removing the limb end, the tip of the skeleton can be removed. The pixels that were removed by this process are shown in white here. This is the basis of the 'spur' removal operator. [QT: *spr*] (i) Shrinking when there are lakes present retains a single point for each blob that was part of the original blob. (*left-most* cross) [QT: *blf, shk*] The centroid (*right-hand* cross) may not lie within the original blob. (j) Shrinking when there are lakes present. A binary object with lakes shrinks to a series of inter-connected rings. Each ring is halfway between a lake and the outer boundary nearby. The Euler number remains unchanged. [QT: *shk*]. (k) Partial thinning. Skeletonisation is performed by thinning recursively, with rules to prevent 'breaking' the figure (i.e., altering the Euler number). Notice that the skeleton is already visible in places. [QT: *thin(5)*] (l) As it is currently implemented skeletonisation (MATLAB: *bwmorph* with the *'thin'* option; QT: *thn*) does not preserve Euclidean distance. Hence, the skeleton of the gear is not circularly symmetrical

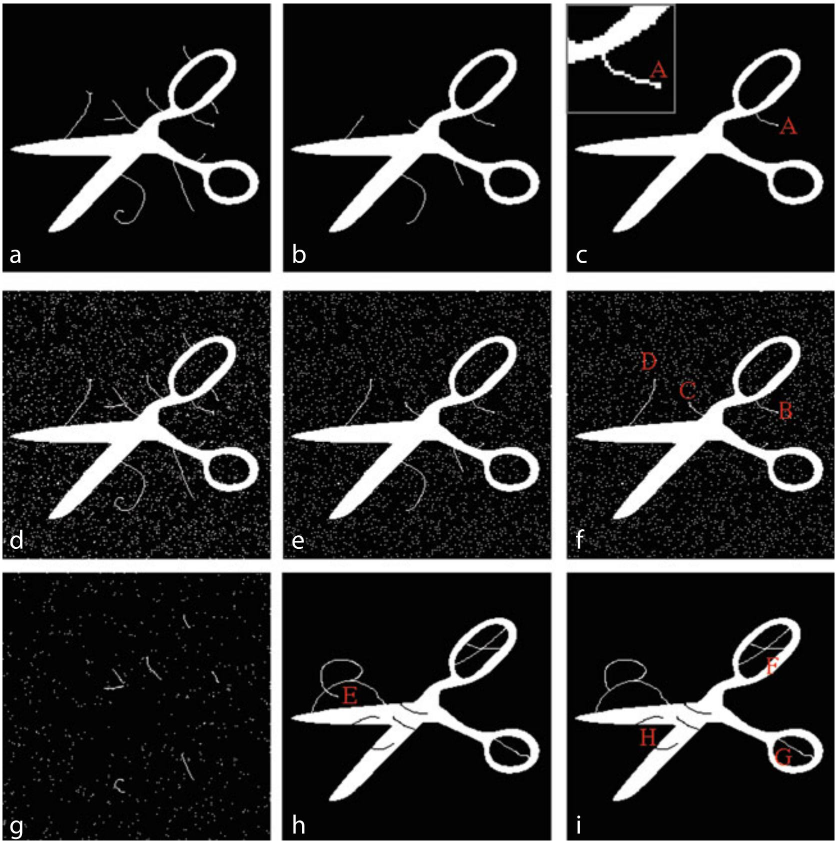


Fig. 19.24

Removing 'hair' and 'spurs'. A hair is a set of pixels forming a 1-pixel wide arc or line. (a) Original image (Hairs were drawn manually, using Photoshop®.) (b) Removing 25 pixels is insufficient to eliminate all hairs entirely. (c) All hairs have been removed. The algorithm will not remove any more pixels from this point on. Notice that one strand (A) remains because its end is not 1 pixel wide. (See inset) (d) Another original image, with noise superimposed. (e) After removing 25 pixels, some hairs remain. (f) The algorithm has terminated as in (c). Notice that B remains for the same reason as A. Removing the hair end pixels stopped prematurely at C and D because noise increased the strand width locally. (g) Differences between (d) and (f). Noise pixels have been removed where they constituted part of short hairs in the background. (h) A third original image. Notice the 'cracks'. (Hand drawn) (i) The algorithm does not affect the cracks (H) nor the loops (F, G), once the tail E has been removed

('cracks') can be eliminated by closing. Hair removal is illustrated in *Fig. 19.24*. Elimination of spurs is possible either by opening or recursive removal of limb ends.

Thickening Thickening is a recursive dilation process, with rules to stop locally when doing so would remove all evidence of a black region. Thickening is, of course, related to dilation, as thinning is to erosion. *Figure 19.25* illustrates thickening in the presence of noise.

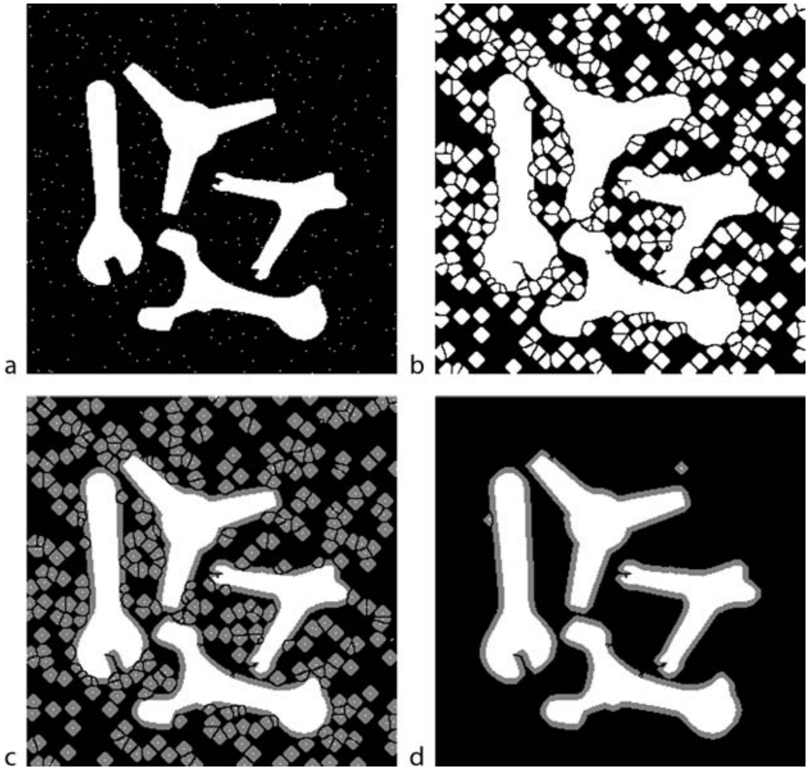


Fig. 19.25
Thickening. (a) Original image. (b) Result of thickening. (c) Thickened image superimposed on the original. (d) Most of the noise spots have been removed, by eliminating single isolated white pixels. [QT: *cln*] The result is improved but still imperfect, showing that thickening is very sensitive to background noise

Watershed Imagine a set of small white spots in a digital image. The watershed function places a 1-pixel wide mesh in the output image surrounding each blob. *Figure 19.26* illustrates this and compares the watershed with ultimate thickening, applying the function recursively until no further changes take place.

19.8 Applying Morphology to Practical Tasks

So far, we have described the binary morphological functions without discussing possible applications. It was necessary to do this because practical inspection task are rarely solved with just one key function. Now that we are properly equipped with a set of tools we can begin to consider the important question of how we use these techniques to solve useful problems. One important point emerges repeatedly: algorithms involving morphological operators cannot be designed without effective experimentation. Often, we find that the higher-level functions are best regarded as combinations of low-level functions that are selected separately. This even applies to opening and closing. If we start by thinking that we should use opening, we might

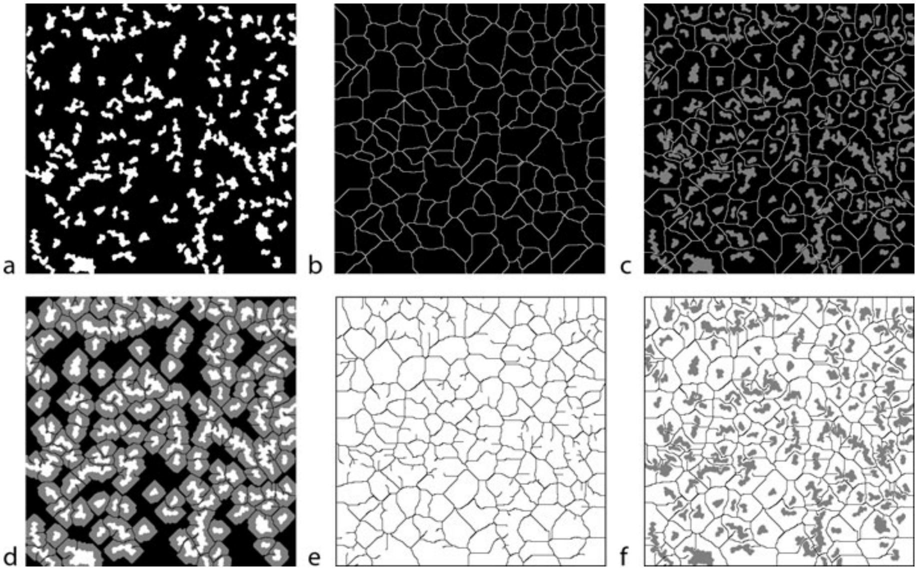


Fig. 19.26
Watershed and thickening. (a) Original image, generated by smoothing and thresholding an image created by a pseudo-random number source. (b) Watershed. (c) Watershed superimposed into the original. (d) Thickening by a limited number of pixels. (e) Thickening until there are no further changes. Notice the presence of 'spurs', which are not generated by the watershed operator. Also, notice that the 'cells' are not quite the same shape by watershed. All pixels within a given cell surrounding an input blob B see a member of B as closer than any white pixel not in B. (f) Thickened image superimposed on the original

miss this opportunity of using SEs of the same shape but slightly different sizes for erosion and dilation. It is important that a vision engineer remain mentally free to improvise, so that he can devise novel combinations. Some of these, such as erosion with a circle SE and dilation with a cross SE might be entirely novel.

19.8.1 Optical Character Recognition (OCR)

OCR cannot be solved by a simple algorithm! There are many different and interacting design aspects that must all be considered when building a practical machine to read text. It is not possible in the space available here to do more than outline a few of the concepts. The obvious way to read printed text, that is *self-evident* to every lay person, is to compare each printed symbol to a series of masks exemplifying every possible member of the character set (letter, digit, punctuation mark, arithmetic symbol, etc.). Finding the 'best fit' mask effectively identifies the character. Easy? Wrong! Practical experience, by many researchers, over many years, has show repeatedly that this 'obvious' technique for OCR does not work with sufficient accuracy. It is far too sensitive to print imperfections.

Figure 19.27 illustrates one possible alternative, which is clearly akin to the 'obvious' method just dismissed. A thinned version (skeleton) of a 'good' character defines the

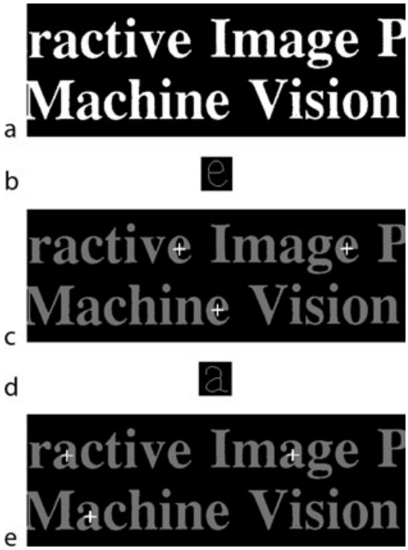


Fig. 19.27

Using erosion to detect characters in printed text. This technique is not normally robust enough for use in practice, since it is very sensitive to character size, orientation, font and print quality. (a) Original binary image. The upper case '*M*' is 84 × 67 pixels. Notice how 'clean' the printed characters are. This would not normally be the case in a practical OCR application. (b) The structuring element for detecting 'e' is the complete skeleton of that letter. (c) Erosion results in an image containing just three small white blobs. The centre of each of these blobs is indicated here by a cross. (d) The structuring element for detecting 'a' is the skeleton of that letter. (e) Erosion correctly identifies each occurrence of 'a'

structuring element, which is then used for erosion. Clearly, the skeleton retains the overall shape of the character but is not so demanding in requiring an exact fit around the edges, which always appear ragged when viewed under high magnification. While it looked promising initially, this second approach to OCR is still not reliable enough.

Figure 19.28 suggests that using a simpler SE might be effective. In this case, features such as skeleton limb ends, joints, simple curves, ɜ and (, etc. could be identified. Of course, the smaller SE is not as selective as we might like. So, it will probably be necessary to combine the decisions of two or more smaller erosion processes using different SEs. We might, for example, combine the results from five different SEs to identify an upper-case letter 'A': apex, left-T, right-T, left-limb-end, right-limb-end. If we obtain a 'hit' for each (or most) of these on the same character, then we can reasonably deduce that the character is an upper case letter 'A'. This approach is likely to be more tolerant than using the complete skeleton as an SE, because two small SEs used separately might fit in slightly different relative positions on different occasions. When they form part of a larger SE their relative positions are fixed. *Figure 19.28* clearly represents nothing more than a promising start. The complex process of reading English text requires our working with a character set containing 70 members, or more. We may be able to 'reuse' some SEs to identify different characters but it is clear that OCR is far from straightforward for even a single type font, style (*italics*, **bold** and regular) and size.

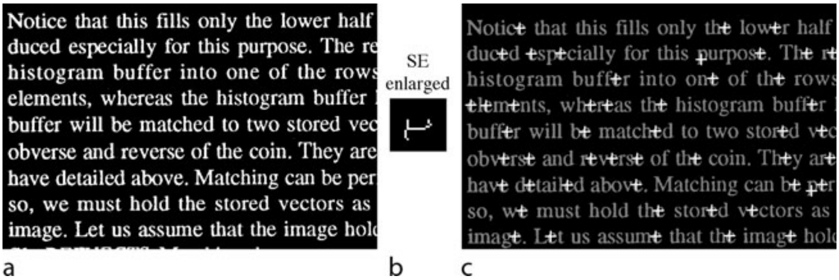


Fig. 19.28

Using erosion to detect specific features rather than whole characters. (a) Original binary image. The upper case '*M*' is 27 × 23 pixels. (b) The structuring element is shown enlarged (4x) for clarity. (c) The centre of each blob remaining after erosion is indicated by a cross. Erosion has identified each occurrence of 'e' but mistakenly detected '*p*' twice

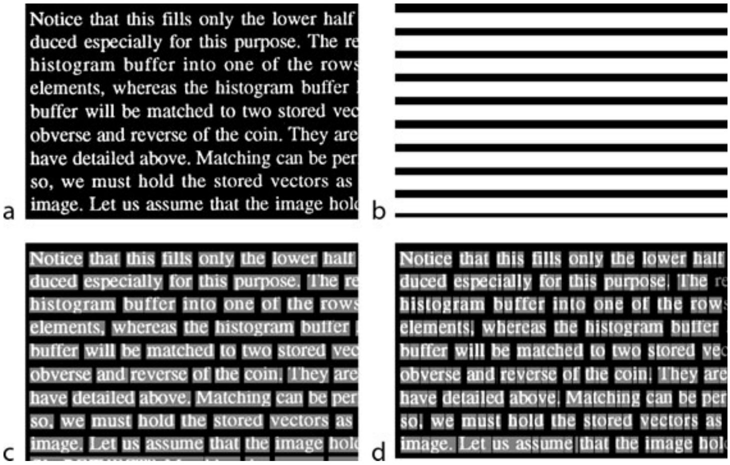


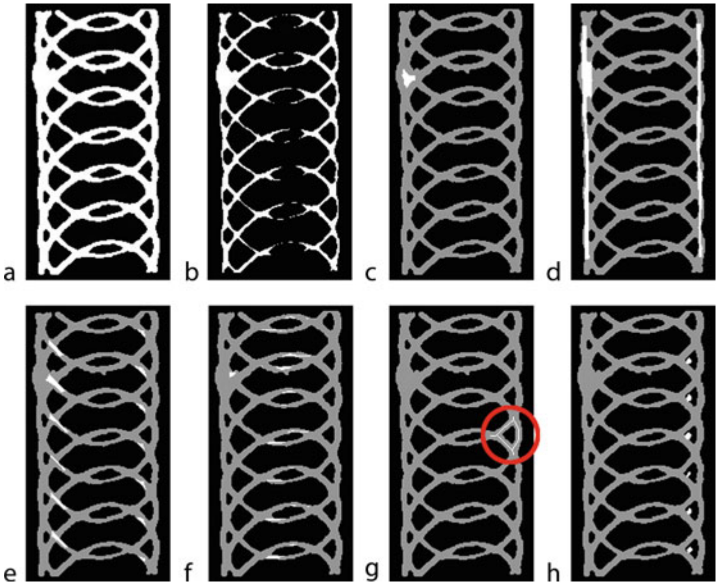
Fig. 19.29

Segmenting text. (a) Original image. (b) Identifying lines using row integration. Row maximum is more sensitive to isolated white pixels originating from noise, or dirt on the printed page. [QT: *rin, csh, thr, dil(7), wri(1)*] Another possibility it to use a procedure based on erosion, with an SE in the form of a long horizontal straight line. This will work better than row integration [QT: *rin*], if the printed text is not aligned exactly with the image axes (i.e., camera axes). (c) Isolating words, using dilation. [QT: *rei, x = cse(1,5); gbd(x); y = cse(2,30); gbd(y), rei(1), mni; gbd(y), rei(1), mni*] Image file (1) is the image saved in (b). (d) Isolating letters, again using dilation. [QT: *rei, y = cse(2,30); gbd(y), rei(1), mni; gbd(y),rei(1), mni*]

Figure 19.29 shows that lines of text, words and individual characters can all be recognised. These are useful prerequisites for successful OCR, as is the early identification of the font, size and style. Although character separation is not perfect in *Fig. 19.29d*, the 'errors' seem to be reasonable, as closer examination shows that they occur where adjacent letters are connected. (Errors in the bottom line of text are due to edge effects.)

19.8.2 Inspecting Cake Decoration Patterns

The cakes illustrated in ➋ *Figs. 19.30* and ➋ *19.31* are sections of indefinitely-long strips that are cut transversely into fingers, and fed in large quantities to lucky little monsters, called children. Clearly, cake inspection does not require the level of precision needed for machine parts; general trends in the constantly-varying appearance are all that is needed. Thinning and thickening of the icing (frosting) pattern are important issues, as are major discontinuities and the misplacement of the pattern relative to the cake body. These two figures show that useful information about factors such as these can be obtained through the use of simple morphological preprocessing. Remember that, at any given moment, the nature of the particular product being made is known, so the ideal form of the decoration pattern is too. The ability to look for specific features, such as the 'ellipses' in ➋ *Fig. 19.30* or the 'tear-drops' in ➋ *Fig. 19.31*, is valuable. Simply counting 'tear-drops' or 'cross-overs' (X) (➋ *Fig. 19.31*) in a given length of cake strip is a useful inspection function but would probably need to be augmented by imposing other criteria.



◘ Fig. 19.30

Erosion applied to a cake decoration pattern. Erosion, plus basic measurements, such as the Euler number [QT: *eul*], or the number of distinct white objects [QT: cbl], may suffice as the basis for a simple inspection algorithm. (a) Original image. Euler number = -30. Blob count = 1. (b) SE is a square of 5 × 5 pixels. [QT: *ero(2)*] Euler number = 0. Blob count = 17. (c) SE is a circular disc of diameter 13. [QT: *rei, x = cse(14,7); gbe(x), adi*] (d) SE is a vertical line of length 30. [QT: *rei, x = cse (2,30); gbe(x), adi*] (e) SE is a diagonal line of length 25. [QT: *rei, x = cse(3,25); gbe(x), adi*] (f) SE is a horizontal line of length 30. [QT: *rei, x = cse(1,30); gbe(x), adi*] (g) SE (white object, circled in red) defined by manually selecting part of the 1-pixel wide 'skeleton'. [QT: *thn*] (h) Erosion using the SE in (g)

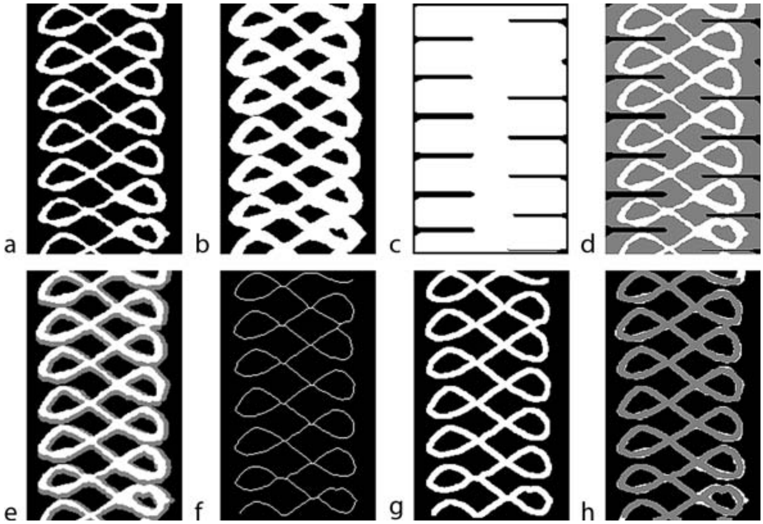


Fig. 19.31
Dilation applied to another cake decoration pattern. (a) Original image. (b) The SE was a solid disc. Notice that the number of lakes and the Euler number have changed. (c) The SE was a long horizontal line. (d) Original image superimposed on the previous image. Notice that this can be used to detect the closure of the gap near the top-right corner. (e) SE was an inverted L. The output consists of the union of the grey and white regions. Notice that dilation with this SE is non-isotropic; it extends the white regions only to the right and downwards. This has no obvious use in this particular application. (f) Thinned version of (a) with spurs removed. (g) Dilation of the previous image using an SE in the form of a circular disc. (h) The previous image compared to the original. This emphasizes the regions where the pattern is too thick

19.8.3 Counting the Teeth on a Gear

Three methods are demonstrated in ➲ *Fig. 19.32*:

- Top-hat
- Erosion, followed by a second stage of erosion. Then apply exclusive OR
- Bottom-hat

On the evidence presented in ➲ *Fig. 19.32*, all three appear to offer viable solutions. Further experimental investigation is needed to select the best one. ➲ *Figure 19.32d*, *f* and ➲ *i* each contains an array of spots, which can be counted rapidly using the Euler number, since they do not contain any lakes.

The author developed these algorithms interactively using the QT system. During this process, the conceptual units coming into his mind were simple *erosion* and *dilation*, rather than the higher level functions, *opening*, *closing*, *top-hat* and *bottom-hat*. This experience has been observed repeatedly during the preparation of this chapter. It is certainly easier to estimate appropriate parameter values and assess the results for the simpler functions when they are considered one at a time.

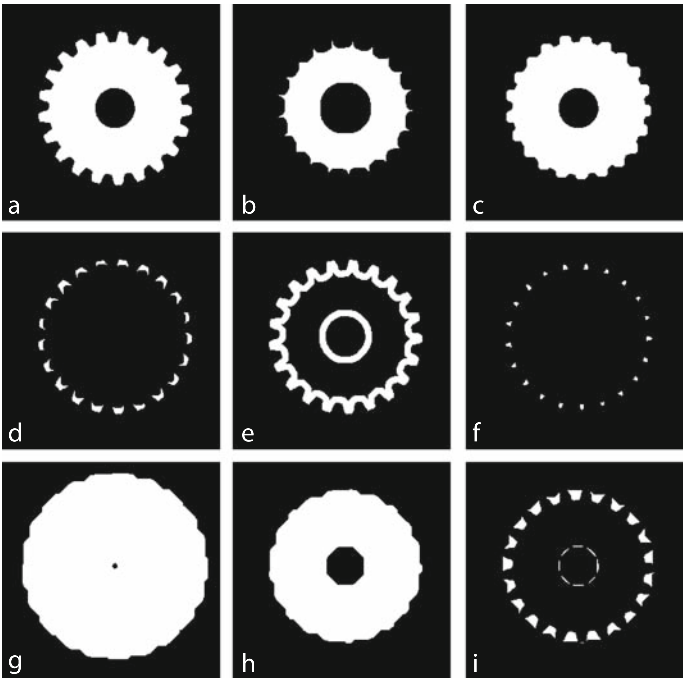


Fig. 19.32

Counting teeth on a gear. (a) Original image. (b) Erosion using an SE in the form of circular disc. (c) Dilation of the previous image using the same SE. (d) Exclusive OR of (c) and the original image. This implements the Top-hat function. To obtain a reliable result, it is probably necessary to dilate this image slightly before counting the blobs. (e) Another approach: Exclusive OR of (b) with the original. (f) Erosion of (e) with a small circular SE. (g) Dilation of the original image. (h) Erosion of the previous image using the same SE. (i) Exclusive OR of the previous image and the original. The large spots are the gaps between the teeth, not the teeth themselves. This implements the Bottom-hat function [QT: *wri, x = cse(14,19), gbd(x), wri, gbe(x), rei, exr*]

19.8.4 Inspecting Screw Threads

The principal features of a screw thread are:

- Shape of the crest
- Shape of the root
- Depth of thread
- Presence of debris
- Flank angle
- Pitch (number of threads/cm)

Morphology can be used for inspecting male (external) screw threads in several ways (➲ *Fig. 19.33*).

The crests and roots of the thread can be located using the top-hat and bottom-hat functions, respectively (➲ *Fig. 19.33b*). This task is similar to finding the tips of the teeth of

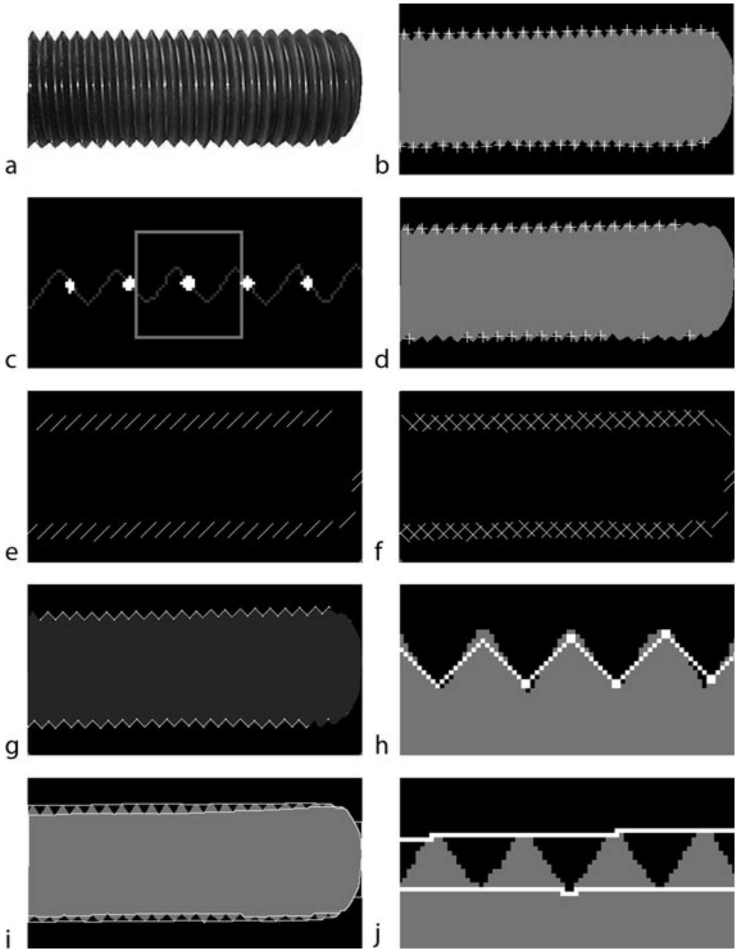


Fig. 19.33

Inspecting a male (external) screw thread. (a) Original grey-scale image. (b) Crests of the thread, detected using the top-hat function. (c) Checking thread form. (magnified view) A small part of the upper edge (1 pitch length) was selected to form the SE (grey rectangle). This was used for erosion on a slightly dilated edge. The white spots indicate the 'hits' over this small sample. [QT: *bed, dil(2), x = use; gbe(x)*] This test can be made more tolerant by increasing the parameter for the *dil* operation. (d) The centres of the spots retained after erosion are represented here by white crosses, which indicate where the thread form is *'acceptable'*. (e) Locating the '/' flanks of the thread. The QT command sequence to do this uses six different morphological operations. It was developed interactively and hence is impossible to fit it into one of the standard categories for morphological functions. [QT: *bed, u = cse(4,4), v = cse(4,5), w = cse(4,11), gbe(u), dil(2), gbd(v), shk, gbd(w)*] (f) Both flanks were detected, as just explained and the results merged. (g) The spurs in the previous image were removed and the result superimposed on the original. (h) Magnified view of the previous image. (i) The vertical positions of the crest tops and root bottoms were located using closing and opening respectively. The SE was a horizontal line whose length was just less than the pitch. (j) Magnified view of the previous image. The vertical separation of these white lines is a measure of the depth of the thread

a gear and can be accomplished using opening. Once the tips/roots have been located, it is a simple matter to count them over a known distance, to determine the pitch of the thread.

Checking the shape of the crest or root is easily accomplished using erosion with an SE derived from a ‘good’ thread. The SE was designed in the following way:

- The edge of the thread was computed. The result was a 4-connected curve 1-pixel wide.
- A small amount of dilation [QT: *dil(2)*] was applied to make the test a little more tolerant than it would be otherwise.

```
bed                   % Binary edge
dil(2)                % Dilate by a small amount. Call this image I.
x = use(0)            % Revert to a 1-pixel wide edge contour. Using the cursor, the user
                      % selects a ‘good’ section of the thread, to define the SE (x).
gbe(x)                % Erode image I (not the original) using SE defined by x.
```

The sensitivity of this method is reduced (i.e., made more tolerant of thread variations) by increasing the parameter given to the *dil* command.

Locating the flanks of the thread is performed in two stages. First, we locate the ‘/’ and ‘\’ flanks separately. The following algorithm was devised while working with the QT interactive image processing system.

```
bed             % Binary edge detector, produces a 4-connected edge contour
u = cse(4,4)    % First line SE (‘/’), length 4 pixels
v = cse(4,5)    % Second line SE (‘/’), length 5 pixels
w = cse(4,21)   % Third line SE (‘/’), length 21 pixels
gbe(u)          % Erosion, 1st SE. Detects the ‘/’ flank.
dil(2)          % Dilation, 3*3 square SE. Joins ‘broken’ flank edge contours, stage 1
gbd(v)          % Dilation, second SE. Joins ‘broken’ flank edge contours, stage 2
shk             % Shrink to single point
gbd(w)          % Dilation, third SE. Approximates flank with line at 45°. (Ideally, should
                % be the same as the nominal flank angle, which is known for each
                % thread type.)
```

The result is shown in *Fig. 19.33e–h*, while *Fig. 19.33i–j* demonstrates that closing with a horizontal line SE can be used to generate a polygonal contour spanning the tops of the crests. Similarly, opening can be used to draw an arc spanning the bottoms of the roots. The gap between these curves indicates the depth of thread.

19.8.5 Locating Holes and Other Features

Figure 19.34 shows how erosion with a ‘hollow’ circle SE can be used to locate holes of a certain size, while ignoring other features. Among the options for the initial processing are the grey-scale morphological edge detectors described later (Sect. 19.12). Notice that the binary image resulting from the edge detector plus thresholding is first dilated, to make the edges slightly thicker before erosion is applied. This makes the operation more robust in the presence of noise. Clearly, once the centres of the holes have been located, a robotic device can be

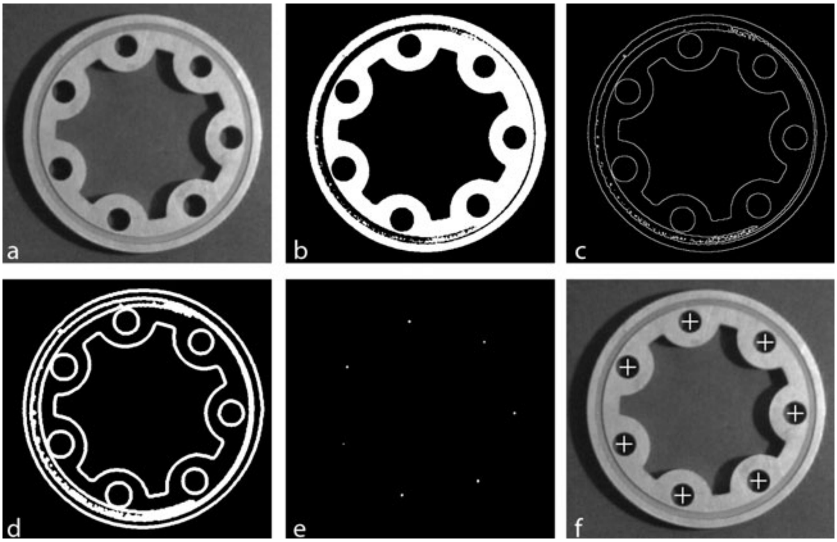


Fig. 19.34
Locating holes in a part of a Wankel engine. (a) Original image. (b) After thresholding. Deliberately, no attempt was made to make the lighting more uniform. (c) Binary edge detection, based on dilation and subtraction. (d) Small amount of dilation applied to make the edges thicker. This operation makes the procedure more robust. (e) Erosion using an open circle SE of about the same diameter as the round holes. (f) The spots in (e) were shrunk to single pixels and then dilated to form crosses. It is clear that these are close to the centers of the holes

directed to process the component in some way: pick up the component, tap its holes, fit an insert in each hole, etc. Here, as in many applications, the morphological operator is just one step amongst several others. Perhaps, it may be regarded the key *concept* but it must be seen as part of the overall solution, not as the whole solution. Hence, it is no more and no less important than the other parts of the algorithm.

➋ *Figure 19.35* is also concerned with locating holes, this time on an electrical power socket. In this instance, the solution involves a combination of operators and, once again, does not fit into any of the standard morphology categories.

As we have already seen, feature location is a key issue in many Machine Vision applications and we will pursue this topic a little further. ➋ *Figure 19.36* shows a thick-film circuit, which presents a complex binary pattern. In order to examine, or machine, a component like this, we must find some 'anchor points'. A frequently-used inspection technique involves comparing the test image to an image from a 'golden sample', or a model derived from one. These are then matched in some way, the details of which need not concern us here. The important task that we will address is that of feature location. As we have done before, the SE was derived from the image. A small sample may be taken directly from the 'golden sample' image, or as in this case, from a processed version of it. The vision engineer should use his best judgement to select a small region of interest with the cursor. In this application, it was necessary to apply a bit of judicious 'touching up', using PhotoShop™, to generate a 'clean' SE. This is totally appropriate, for example, when we know that the object under inspection is constructed with straight edges and 90° corners. In such a situation, it would not be justified to use a 'ragged' SE to search for

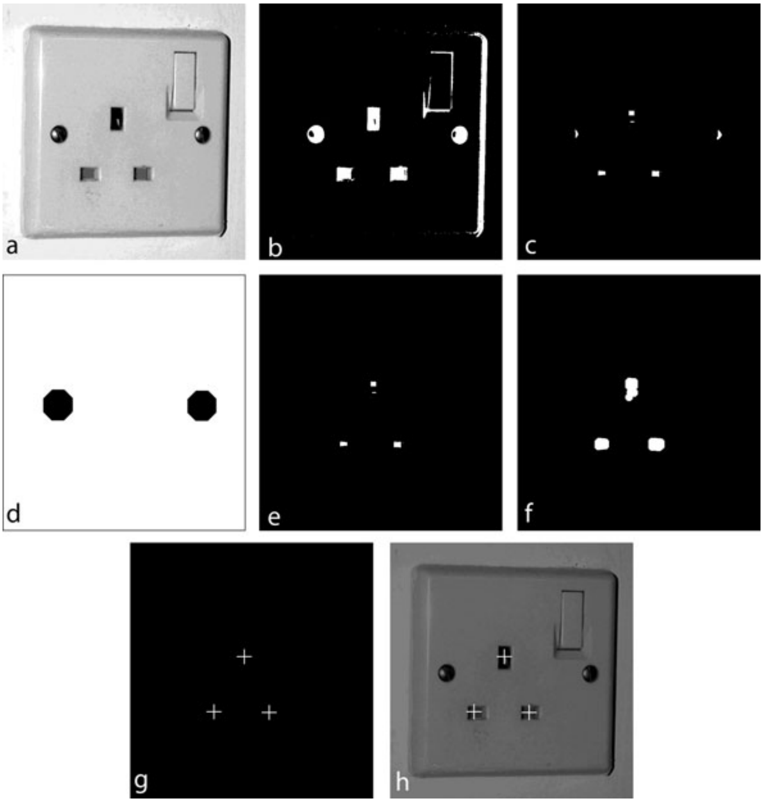


Fig. 19.35

Application requiring a combination of morphological operators. (a) Original image, British 240V/13A electrical power socket. (b) Threshold applied. (c) Erosion with solid square SE. (d) Erosion of (b) with open circle SE. The resulting image was negated, dilated using a solid circle SE and then negated. (e) Previous image was used to mask (c). (f) Dilation with solid square SE. (g) Shrunk to single pixel and then dilated with '+' shaped SE. (h) Result superimposed on the original. The following sequence of operations was disovered using QT. (There were a few short 'blind alleys' in the search tree and these have been removed for the sake of clarity.) [QT: *rni(original); enc; thr; type cse; neg; pxv; wri; x = cse(7,20), gbe(x); wri(1); x = cse(14,40), gbe(x); rei; x = cse(13,40), gbe(x); x = cse(14,40), gbd(x); neg; rei(1); mni; dil(3); swi; dil(7); swi; dil(11); shk; dil; crs(20); rni(original); swi; adi; mxi*]

features like these. The points indicated by crosses in *Fig. 19.36c* and *e* (not the crosses themselves) effectively form 'star maps'. The combined set of coordinate pairs representing points in these two images would form an ideal input for a Prolog program (Chap. 23). Such a program would align each test image with an image derived from a 'golden sample'. To do this, two 'star maps' should be matched. In some cases, when there is some noise or dirt, two such maps might contain different numbers of 'stars'. This is exactly the kind of situation in which Prolog excels. Feature detection using morphology and Prolog reasoning are ideal companions in the kind of application that we have just described.

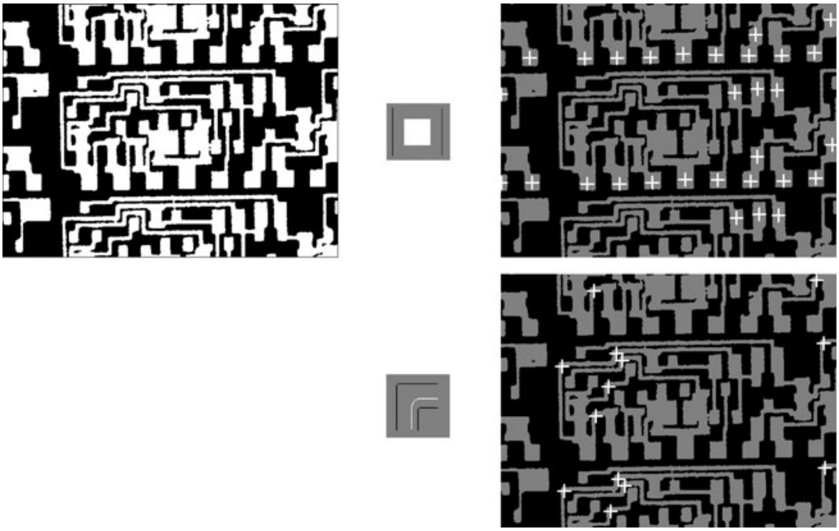


Fig. 19.36

Hit-or-miss. (a) Original image, thick film circuit. (b) Hit-or-miss requires two structuring elements: black: *se_x*; white: *se_y*. These refer to the SEs used in the QT command sequence given below and are designed to detect white rectangles that have no white pixels along two vertical lines beside it. (c) Hit-or-miss produced a set of small white blobs but these are not easily displayed at this resolution. For clarity, the centre of each of these blobs is represented by a cross. (d) Another pair of structuring elements, designed to detect the corners of thin vertical conductors as they 'turn right'. (e) Result. [QT: *rei('se_y'), y = strel(current); rei('se_x'); x = strel(current); rei, current = bwhitmiss(current,y,x), thr(1), shk, dil(2), crs(25), rei(original'), swi, adi, mxi*. Both *strel* and *bwhitormiss* are MATLAB functions]

Figure 19.37 demonstrates feature location on a bare printed circuit board. In 1978, J.A.G. Halle and P. Saraga [3] used morphology for this task, although this method of processing images did not yet have that name.

19.8.6 Colour Recognition

This topic cannot be covered in detail here and the reader is referred to Chap. 16 for further explanation and examples. *Figure 19.38a* shows the so-called *Colour Scattergram* and was derived from *Fig. 19.37a*. The details of how this is done are not needed here; *Figure 19.38a* is the starting point for what follows. Various morphological functions have been used in the preparation of the colour map:

- Dilation, to join disjointed spots that form a cluster
- Opening, to remove small white spots in the thresholded colour scattergram
- Closing, to join disjointed spots that form a cluster
- Majority filter, to eliminate very small spots
- Thickening, for limited colour generalisation
- Watershed, for extreme colour generalisation

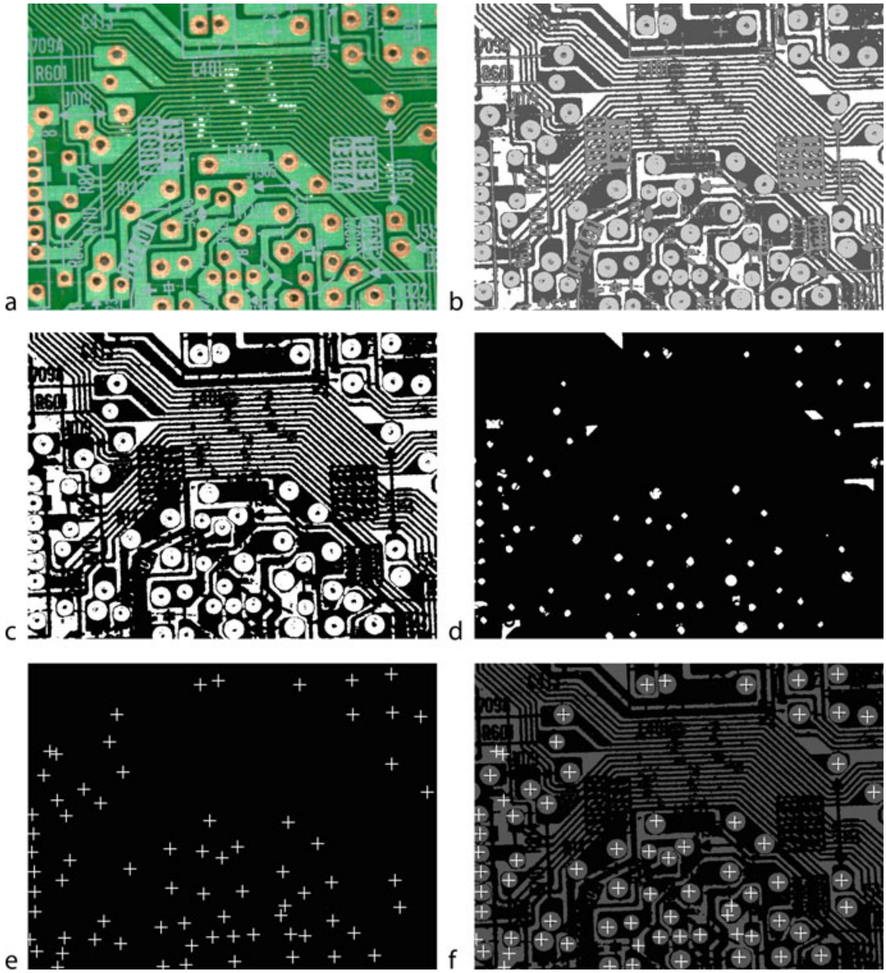


Fig. 19.37

Locating pads on a printed circuit board. (a) Original image. (b) Image segmented using a programmable color filter (*Fig. 19.38e*) and Chap. 16 (c) Thesholding of (b) distinguishes the copper conductors from the substrate. (d) Erosion with an open-circle SE was applied first. The size was about 80% of the diameter of the pads. The result was then ANDed with the negative of (c). Finally, a small amount of dilation was applied to merge small nearby blobs. (e) The blobs in (d) were shrunk to single pixels. Their positions are indicated here by the crosses. (f) Results superimposed on the original binary image, (c)

19.8.7 Texture Analysis of Binary Images

Texture analysis is a vexed subject, despite its importance in a wide range of applications, including automobile (paintwork), white goods (paintwork), ceramics (microstructure), timber (grain), food (cell matrix structure), etc. The reason is that *'acceptable'*, or *'normal'*, texture in a given application cannot be defined satisfactorily by rule, or formula. So, it is necessary to

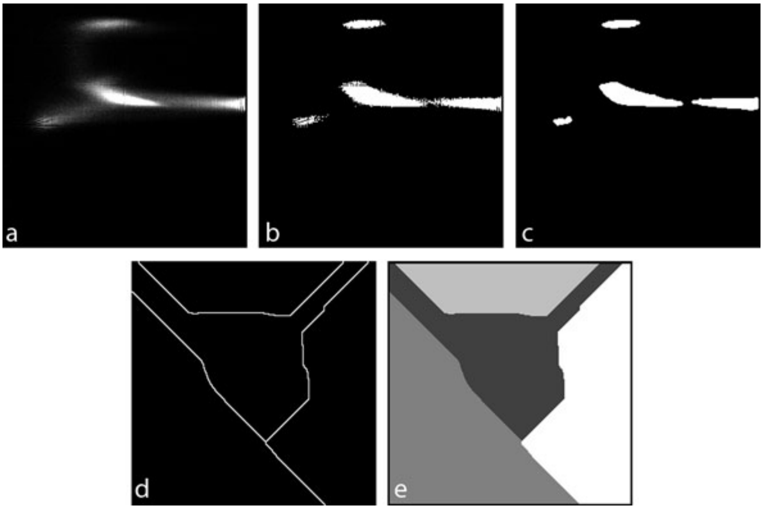


Fig. 19.38

Colour recognition using morphology for preprocessing. (a) Color scattergram of Fig. 19.38a. (See Chap. 16.) (b) Thesholding. Notice the ragged edges of the blobs and the small spots. (c) Majority voting filter. (d) Watershed. (e) Region labeling. This is the colour map which, when applied to Fig. 19.38a, produces the image in Fig. 19.37b (see Chap. 16 for details of the Programmabale Colour Filter)

learn from samples what a person labels as '*acceptable*', or otherwise. It may not be possible to collect a representative sample of '*not acceptable*' textures but it is usually possible to obtain a set of '*acceptable*' textures that is large enough to allow statistical, or Pattern Recognition, methods of analysis to be used (Chap. 24). In this brief section, we have a very specific objective: showing how morphological operators can be used to generate a multi-variate (vector) description of binary texture. Once such a data representation has been obtained, it can be applied to a Pattern Recognition system. *Figure 19.39* shows several examples of binary texture being transformed using different morphological operators. Simple measurements on the resulting images, such as blob count and number of white pixels, are affected in different ways by these operators (*Table 19.2*). The important points to note are

1. While it is easy to suggest a wide variety of texture measurements, the vision engineer should be prepared to invent novel combinations of erosion and/or dilation. Many morphological operators are quite fast in operation and several functions can often be performed together, with very little increase in computational effort. (See Chap. 19.)
2. Selecting which measurements are most useful is always difficult. So, aiming for a 'sufficient solution' is probably better than fretting about not being able to find the very best one.
3. Pattern Recognition, statistical and other data analysis programs can probably help in choosing a good set of texture descriptors but there is no point in ignoring human intuition and common sense.
4. If there is a strong directional component in the texture, then exploit it by using structuring elements that are aligned and others that are orthogonal to it.

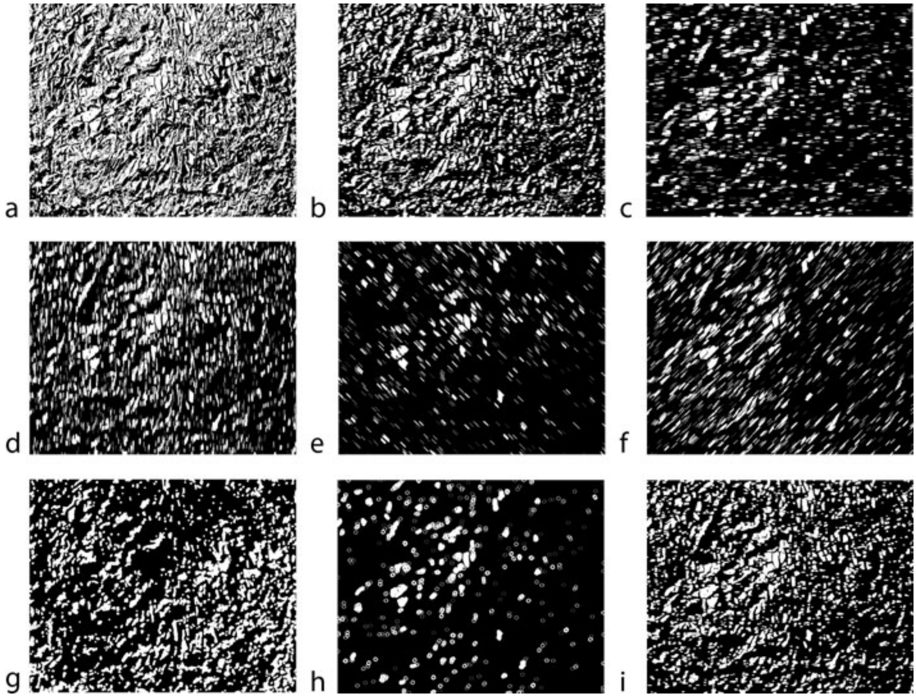


Fig. 19.39
Analyzing the texture of binary images using morphological operators. (a) Original image, thresholded so that 50% of the pixels are white. (b–i) Processing with eight different SEs. The details are given in *Table 19.2*

19.9 Recursion and Speed of Operation

In the discussion above, morphological functions have several times been described as recursive. We need to consider the implications of this, since the very mention of the word ‘recursion’ suggests that execution time is unpredictable. However, we can place some limits on the execution time. We will not, however, claim in absolute terms such as ‘*Function* $\mathbf{F}$ *with structuring element* S *can be performed in* T *seconds on an image containing* P *pixels*’. Instead, we shall show, in functional form, how T depends upon P and the size of S. First, however, we shall show why recursion is necessary.

Consider the task of thinning to generate a skeleton. It is clear throughout this chapter, beginning with *Fig. 19.12*, that erosion with a large SE eliminates thin white streaks, while retaining wide ones but leaves them slimmer than before. In order to generate a skeleton, fine control of the erosion process is needed. Let us now apply erosion repeatedly, using a small SE that on each cycle removes a 1-pixel wide object boundary (*Fig. 19.40*). Again, thin white streaks are lost, so we do not appear to be any closer to our objective. However, let us now modify erosion by adding a rule that acts locally to stop the process when the *next* erosion cycle would break the blob into pieces.

Table 19.2

Measuring texture using morphology. Notice that different sizes and shapes of structuring element produce varying blob count and white area values. We can combine these to generate a multi-variate description of the texture that can then be applied to a pattern classifier

Operation	Fig. 19.39	Structuring element (SE)	Size of SE (pixels)	No. of blobs	No. of white pixels
None (original)	(a)	Not applicable	–	2,614	7,40,341
Closing	–	Horizontal line (–)	5	2,467	6,64,893
Closing	(b)	Horizontal line (–)	10	2,557	4,85,489
Closing	–	Horizontal line (–)	15	1,952	3,24,627
Closing	(c)	Horizontal line (–)	20	1,365	2,07,498
Closing	(d)	Horizontal line (–)	25	8,59	1,29,194
Closing	(e)	Vertical line (–)	20	1,595	3,57,155
Closing	(f)	Diagonal line (\)	20	6,83	1,00,582
Closing	(g)	Diagonal line (/)	20	1,013	2,10,168
Closing + negation	(h)	Solid circle (●)	5	8,17	4,09,526
Closing + negation	(i)	Open circle (○)	20	1,465	1,22,150
Closing	–	Rectangle (✺)	10 × 5	965	4,16,043
Opening	–	Rectangle (✺)	10 × 5	1,364	38,580

Many papers have been written about skeletonisation. Here, we will state only that the desired result cannot be achieved using only the information from a 3 × 3 neighbourhood of the current pixel (➲ *Fig. 19.41*). The 3 × 3 neighborhood does not allow 2- and 3-pixel wide arcs to be distinguished, whereas the 4 × 4 neighbourhood does. Hence, they can be treated differently. Repeating this kind of analysis for stripes of all widths and orientations shows that the 4 × 4 neighbourhood, with the upper-left corner omitted, is sufficient to avoid fracturing, in all cases.

Floeder [4–6] showed that a single pass using the information from a 4 × 4 neighborhood is sufficient. ➲ *Figure 19.42* shows the seventy-eight 4 × 4 templates required by the Floeder algorithm, which performs one thinning stage in each pass. These templates distinguish between pixels that can and cannot be removed without 'breaking' the resulting skeleton. The Floeder thinning algorithm is simply stated thus:

- If the pixel is white and its neighbourhood matches one of the templates, then it is set to black.
- Otherwise, the pixel remains unchanged in the output image.

Various approaches and implementations are used for thinning and other high-level morphological functions by different researchers and equipment vendors. The SKIPSM implementation (➲ Chap. 20) [7–9] allows all seventy-eight of the Floeder templates to be tested in no more time than it takes to test one of them. (This remarkable fact is explained in ➲ Chap. 20.) For this reason, SKIPSM is likely to be one of the fastest implementations, when compared to others running on computers of comparable speed.

Now that we have established what each cycle of the thinning process requires, we can return to the calculation of timing the complete skeletonisation process. This can only be

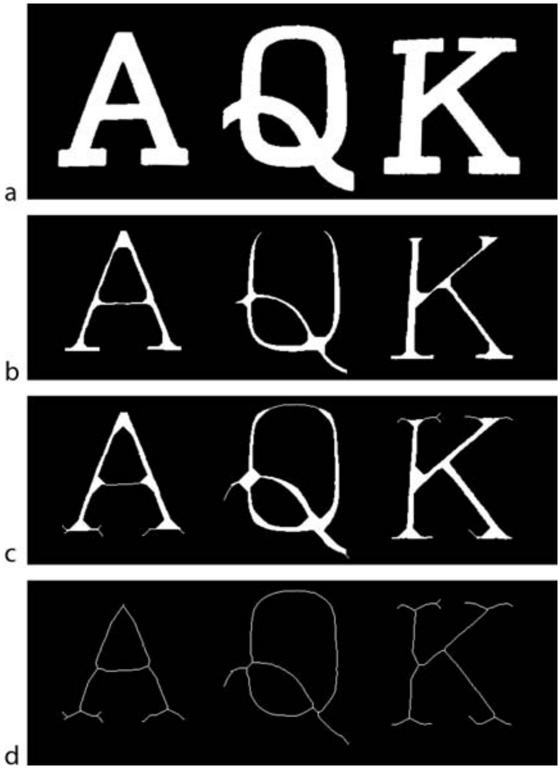


Fig. 19.40
Iterated erosion and thinning. (a) Original image. (b) Erosion after several cycles has already broken some of the white streaks. (c) Thinning has not broken any streaks. Thinning is not yet complete. (d) Skeleton is 1 pixel wide. Thinning this has no effect

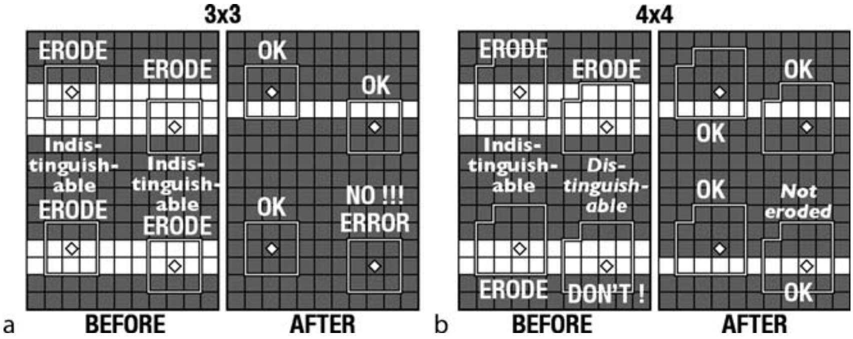


Fig. 19.41
Skeletonisation is impossible using a 3 × 3 pixel SE because there is no way to sense when to terminate recursion (a) SE: 3 × 3 pixels. (b) SE: 4 × 4 pixels with the top-left pixel omitted. Erosion is performed by scanning from left to right, top to bottom

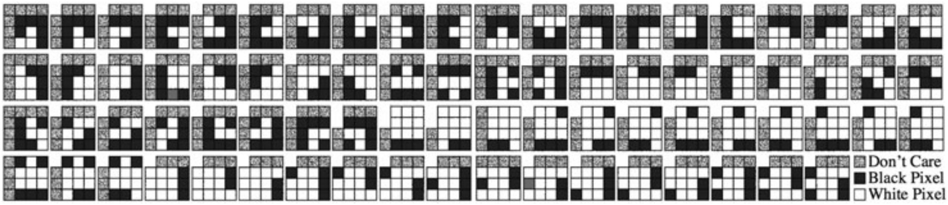


Fig. 19.42
The seventy-eight 4 × 4 templates used in the Floeder skeletonisation algorithm

answered by finding out how many thinning passes are required. On each cycle, a 1-pixel curve is removed all around the border of each blob. This will require *G* passes, where *G* is the maximum intensity in the grass-fire transform, to generate the skeleton. This is immediately apparent when we realise that these two functions are both built on erosion. (Notice that measuring G is a very simple task.) Let the image resolution be *M* × *N* pixels. For the SKIPSM implementation, the time to process each pixel is almost independent of the size of the structuring element. (Another remarkable feature of SKIPSM!) Denote this by L. Then, the time to reduce the image to its skeleton is G.M.N.L. For a square image, this simplifies to G.N^2. L. Notice that, for the SKIPSM implementation, thinning takes almost exactly the same time as erosion to process each pixel. So we are able to conclude that skeletonisation takes *G* times as long as a single erosion cycle. This parameter is measurable, using a simple test: the grass-fire transform. Notice that 'fat' blobs take longer to thin than 'thin' ones. The execution time depends on the square of the number of image rows/columns but is linearly dependent on G. If we study the same skeletonisation task at different resolution levels we must take note of the fact that *G* is proportional to *N*. So, the time to perform skeletonisation varies as N^3. This cubic dependence on resolution can make thinning, skeletonisation, thickening and watershed quite slow, so any speed increase that can be achieved by any means is welcome. SKIPSM greatly reduces the time spent processing each pixel (L) but the execution time is still proportional to N^3.

19.10 Hit-and-Miss Function

Structuring elements defined for erosion and dilation require only that some pixels be *white.* Any pixels in the processing rectangle but not in the SE are referred to as *'don't care'* pixels. It is an obvious conceptual extension to binary morphology to use structuring elements that contain *white*, *don't care* and *black* elements. An appropriate name for this is *Binary Template Matching*, but it is usually called the *Hit-and-Miss* function. The SE 'fits' the image at a particular position if and only if all of the white pixels *and* all of the black pixels in the SE are matched by corresponding image pixels. In that case, the output image is set to white. Otherwise, it is set to black.

Mathematical Definition: Let the K elements of the SE be denoted by $S_1, S_2, \ldots, S_K$. The values of the S_i are restricted to be either Boolean 1 (*white*), or Boolean 0 (*black*). (*Don't care* elements are not allowed as values for any of the S_i. They have no role in the definition of the algorithm. They are only defined so that the SE can be represented graphically, which is

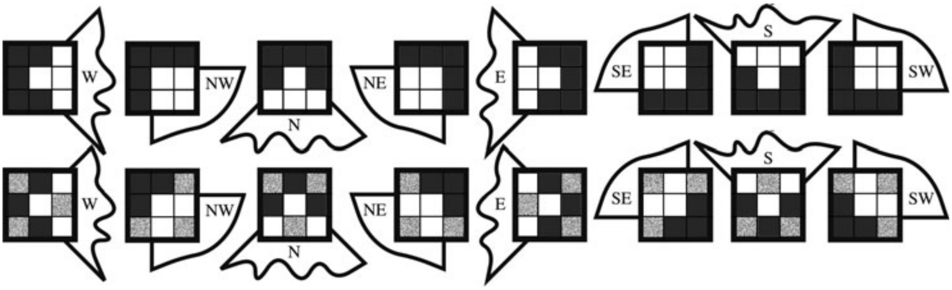


Fig. 19.43

The upper row shows 'exact' templates, and the lower row shows comparable templates with *don't care* pixels (speckled). The templates in the lower row are preferred as they produce a more robust corner detector when image noise is present. Structuring elements for detecting corners. (a) Allowing only white and black elements. (b) Allowing *'Don't care'* elements. These SEs produce a more robust corner detector in the presence of image noise

a significant conceptual aid.) As usual, let the K pixels of the input image covered by the SE be denoted by $a_1, a_2, \ldots, a_K$. Then, the Hit-or-Miss function is defined by

$$Output\_Image(i,j) = \{(a_1\ \ XNOR\, S_1)\&(a_2\ \ XNOR\, S_2)\&\ldots\&(a_K\ \ XNOR\, S_K)\}.$$

where XNOR (Exclusive NOR) is the logical operation with output Boolean 1 if its two arguments are the *same* and Boolean 0 if they are not the same. The above expression therefore gives a *white* output if all the image pixels match the corresponding SE elements, and *black* otherwise.

Noise template matching of any kind is susceptible to noise. As a result, some of the desired features may be missed. Performance in the presence of noise can often be improved by incorporating 'don't care' pixels. *Figures 19.43* shows 'exact' and 'don't care' versions of eight 3 × 3 SEs used to detect corners. *Figure 19.44* shows 'exact' and 'don't care' versions of four other larger (9 × 9 or 9 × 18) SEs that have successfully been implemented as fast, one-pass operations in SKIPSM [7].

19.11 Grey-Scale Morphology

Grey-scale morphology extends the ideas already discussed to grey-scale images. Familiarity with binary morphology will help in understanding grey-scale morphology. We will begin by looking at binary morphology in a different way. This far, our conceptual approach to processing binary images was as if we were looking at a map, in plan view. Images can also be considered as three-dimensional objects: the pixel locations define coordinates in the horizontal XY plane, while the image intensity determines the Z dimension. *Figure 19.45* shows a binary image and a grey-scale image represented in this way. The binary image has only two levels (0 and 1), so it looks like a 'mountain range' with sheer cliffs and flat plateau(s). A grey-scale image can take on intermediate values, normally with gently sloping 'hills and valleys'. In both cases, assume that the 'mountains' are hollow and that their surfaces are impenetrable by the SE, which is pushed up (for erosion) or down (for dilation). That is, the SEs can touch the surfaces but not pass through them.

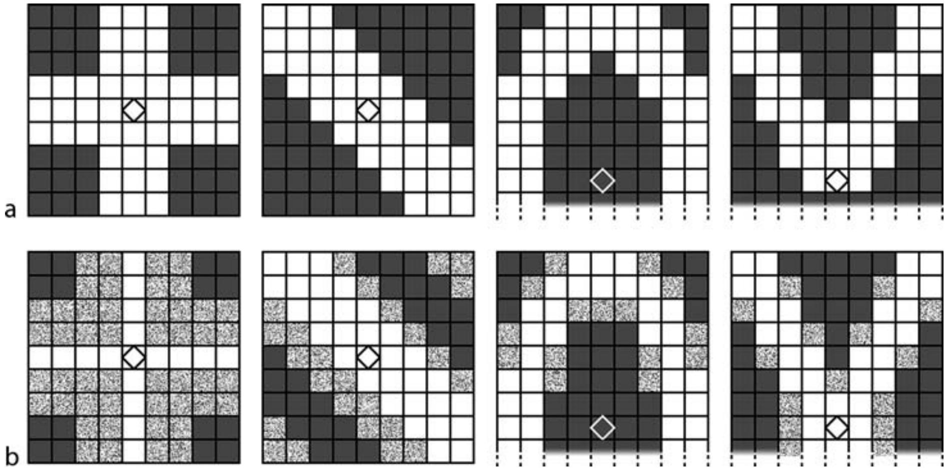


Fig. 19.44

Larger SEs. These have been implemented as fast one-pass operations in SKIPSM. (a) Allowing only white and black elements. (b) Allowing *'Don't care'* elements

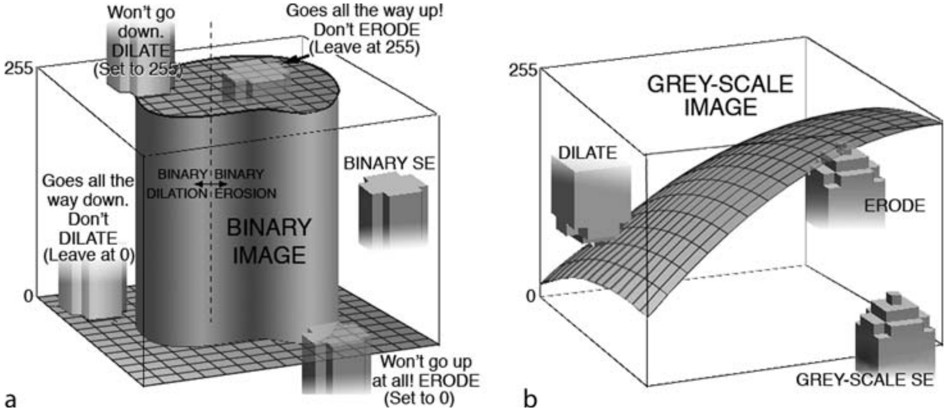
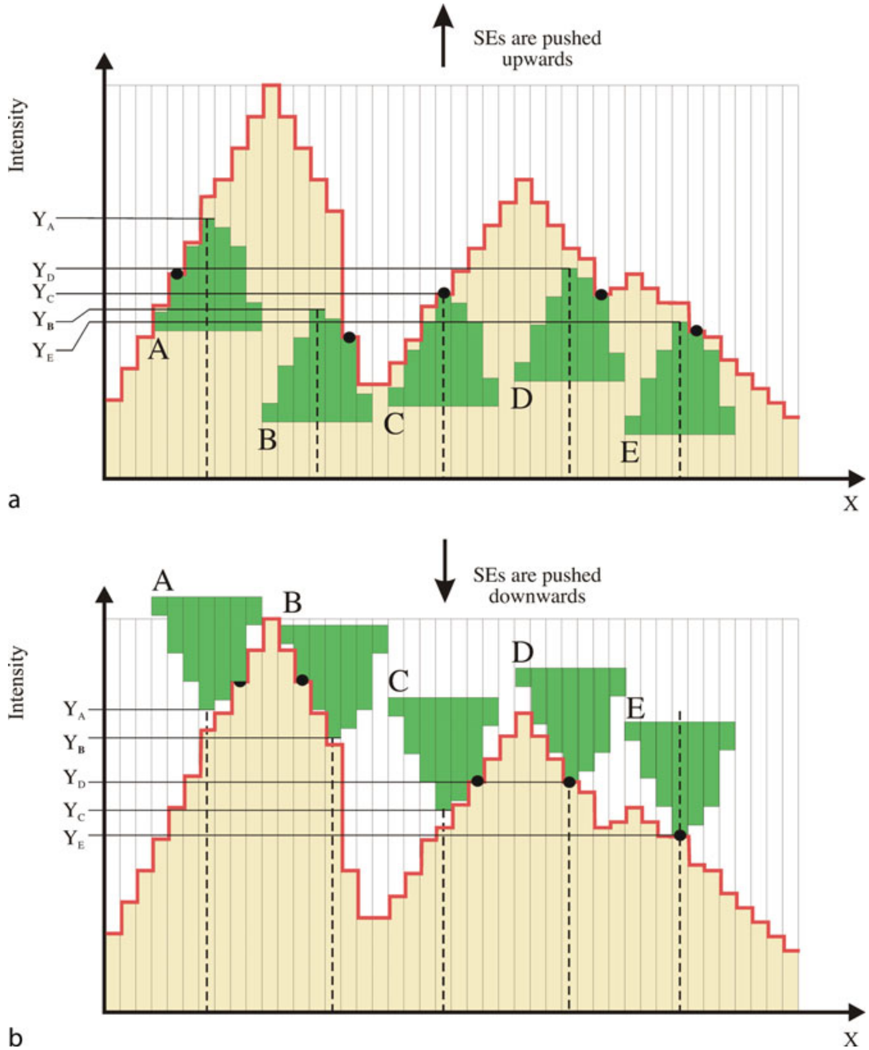

Won't go down. DILATE (Set to 255)
Goes all the way up! Don't ERODE (Leave at 255)
255
BINARY SE
Goes all the way down. Don't DILATE (Leave at 0)
BINARY DILATION
BINARY EROSION
BINARY IMAGE
0
Won't go up at all! ERODE (Set to 0)
a
255
GREY-SCALE IMAGE
DILATE
ERODE
0
GREY-SCALE SE
b

Fig. 19.45

Representing erosion and dilation as operations in three-dimensional space. For erosion the SE is pushed up. For dilation it is pushed down. (a) Binary morphology. The large grey cylinder (or mountain), representing a white blob on a black background, is hollow and has an impenetrable top for the SEs. (b) Grey-scale morphology. The sloping surface representing the intensity function of a grey-scale image is impenetrable for the SEs

Figure 19.46a shows that grey-scale erosion represented as pushing the SE up from the bottom, until it 'crashes' into the intensity surface, while *Fig. 19.46b* shows dilation as pushing the SE downwards. With this in mind, we see that grey-scale and binary erosion are similar, except that in the former case, the SE hits the intensity surface and usually stops somewhere between black and white. Notice, that grey-scale SEs may have flat or contoured tops. The output grey level of the morphological operation is defined by the height of the center pixel of the SE as it crashes into the intensity surface.



◘ Fig. 19.46

Grey-scale morphology represented in one dimension. (a) Erosion. At each position, X, the SE (green polygon) is pushed up as far as it will go, without any part of it crossing the intensity surface, represented here by the red line. At position A, the SE can be pushed up to height Y_A. Similarly, at position B, the SE can be pushed up to height Y_B and so on. The output is given by the sequence of Y values generated in this way. The small black circles indicate where the SE 'crashes' into the intensity profile. (b) Dilation. The SEs are pushed downwards

In regions of gradual intensity change and with these round-top SEs, the outputs for both erosion and dilation follow the input quite closely. Where the intensity gradient is larger, the differences between the outputs and the original are greater. Note that, for grey-scale erosion, the output value can never be higher than the intensity at the centre-pixel location in the original image but it can be lower. As we might expect from duality, the output from grey-scale dilation can never be lower than the original value.

As an alternative to this graphical way of visualising grey-scale erosion, we now present a mathematical expression for this operation and grey-scale dilation. It is similar in form to those for binary morphology, except that the function $\mathbf{F}$ is more complex.

As before, a_1, a_2, . . ., a_K are the values of the K input-image pixels in the neighbourhood defined by the SE at position (*i, j*). In the following definitions, g_1, g_2, . . ., g_K are positive offset values for the corresponding SE elements. These often span only a very small part of the available intensity range, [*black,white*]. Let

$$g_{max} = MAX(g_1, g_2, \ldots, g_K)$$

$$g_{min} = MIN(g_1, g_2, \ldots, g_K)$$

Then, erosion is defined thus

$$\text{Output\_Image}(i, j) = MIN((a_1 - g_1), (a_2 - g_2), \ldots, (a_K - g_K)) + g_{max}$$

and dilation as

$$\text{Output\_Image}(i, j) = MAX((a_1 + g_1), (a_2 + g_2), \ldots, (a_K + g_K)) - g_{min}$$

The higher-level functions for grey-scale morphology are defined in a similar way to binary morphology:

- Grey scale opening: dilation followed by erosion
- Grey-scale closing: erosion by dilation
- Grey scale top-hat: erosion followed by subtraction of the original
- Grey scale bottom-hat: dilated followed by subtraction of the original

We have not included grey scale hit-or-miss in this list because the concept of ‘don’t know’ has no meaning when dealing with grey-scale images. Instead, use the term ‘*ignored*’, which applies to all of the other grey-scale operations as well.

19.11.1 Building Grey-Scale SEs

Suppose we wish to ‘borrow’ a segment of a real image in order to construct a grey-scale SE for erosion (or dilation). One simple way to do this is to copy a small area of that image to define the SE’s values directly. To do this, it would normally be convenient to take a rectangular sample S of the image to define the SE. That is, we use the intensity values of pixels within S to define $\{a_i \, | i \in S\}$. However, we might be interested in using only a subset of *S* to define a smaller SE. We can adjust some of the offset terms, {g1, g2, . . .} so that they will be ignored by the MIN function. Let $S' \subset S$ denote the set of pixels we wish to use in the SE and $S'' = (S \sim S')$ those that we wish to ignore. An SE using all members of S will produce exactly the same result as one using only members of S' if we choose the values for the gi as follows:

```
for $i$ = 1:K
    if $i \in S'$
            $g_i = 0$    % Pixel $i$ is ‘wanted’ & will be not be ignored
    else
            $g_i = -W$    % W = white level. Pixel $i$ is ‘unwanted’ & will be ignored
end
```

Rather than using S' to define the SE, we use S and give the offset parameters for 'unwanted' pixels (i.e., $\{g_i \mid i \in S''\}$) some large negative value. Terms such as $(a_i - g_i)$ where $i \in S''$ will then be ignored by the MIN function.

Why should we deliberately make the SE larger? It is easy to select a rectangular region of an image to define the set S. We can then 'switch off' certain elements by setting the gi appropriately. This can all be done interactively, using only the computer mouse and hence readily lends itself to experimentation. Of course, more computational effort is involved using S rather than S', so the SE would have to be redesigned for a target (factory floor) system. While this is not difficult, it can be tedious.

One word of warning is required: designing an SE by sampling an image allows its parameter values to become 'frozen' noise. For this reason, it may be necessary to make some manual adjustments. For example, pixels within a region of nominally constant intensity should generate equal a_i and g_i values.

SE terms can be ignored in grey-scale dilation by giving the appropriate g_i offset parameters large positive values.

19.11.2 Examples and Uses of Grey-Scale Morphology

➲ *Figure 19.47* shows how grey-scale erosion can be used to detect objects of a given orientation.

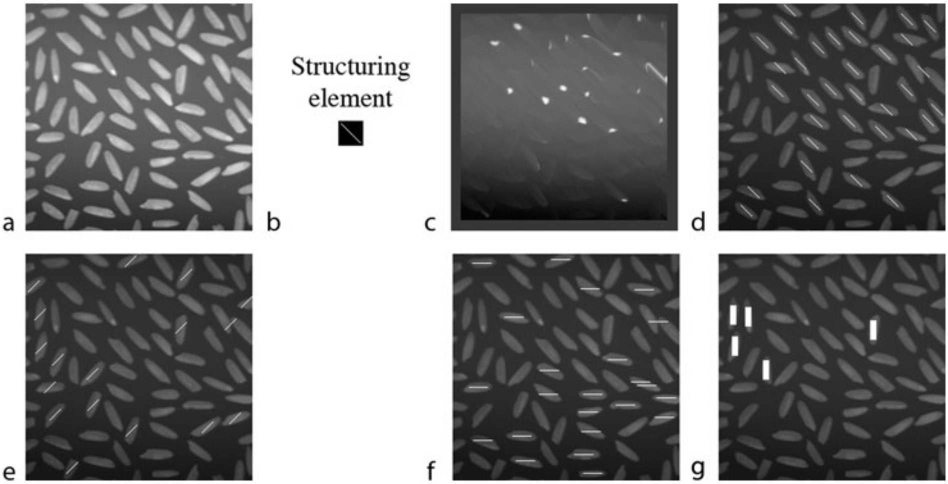


Fig. 19.47

Using grey-scale morphology to detect objects of a given orientation. (a) Original image, rice grains. (b) Diagonal line structuring element, 1 × 13 pixels. (c) Erosion. [QT: *x = gsk(3,13); gse(x)*] (d) The previous image was thresholded. The resulting white blobs were then shrunk to single pixels and (binary) dilated, using the same SE. Finally, the result was superimposed on the original. [QT: *thr(110), shk, gbd(se), rei(original), swi, adi, mxi*] (e) Same processing with the orthogonal diagonal line structuring element. (1 × 13 pixels) (f) Same processing, again. The structuring element was a longer horizontal line. (1 × 22 pixels) (g) Similar processing but with a rectangular structuring element (9 × 24 pixels)

➤ *Figure 19.48* shows that grey-scale erosion or dilation, followed by subtraction of the original, produces a very effective edge detector. The size of the SE determines the width of the edge produced. Line SEs can be used to detect edges at a given orientation. The point made in relation to ➤ *Fig. 19.20d* applies here: dilation produces an outer edge and erosion an inner edge.

The ability to detect cracks and other image features appearing as thin dark streaks using bottom-hat has been known for many years and predates the formalised approach now encompassed by Mathematical Morphology (➤ *Figs. 19.49* and ➤ *19.50*. Also see ➤ Chap. 27. The author discovered the 'crack detector' algorithm independently, using a precursor of QT.) Thin dark streaks that are known to arise in a given orientation can be detected most effectively using a line SE, rather than a rectangle or disc. Thin bright streaks can, of course, be detected by top-hat, or by negating the image before applying bottom-hat. The crack detector has been very successful in solving a wide variety of applications not involving cracks. One such application is 3D shape measurement by triangulation, using projected light stripes (➤ *Fig. 19.51*).

➤ *Figure 19.52* shows how a combination of grey-scale morphological operators can be used to locate simple patterns, such as hollow rings (pads on a printed circuit board). It is interesting to note that the SEs were successfully selected experimentally. It would be very difficult to design the SEs without the ability to try different ideas and see the results almost immediately.

Grey-scale morphology has a significant role to play in preprocessing. For example, textured images can often be enhanced in this way (➤ *Figs. 19.53* and ➤ *19.54*).

Noise in grey-scale images can be suppressed by opening (eliminates bright spots) and closing (eliminates dark spots). Use both to eliminate dark and bright spots (➤ *Fig. 19.19*).

The grass-fire and watershed functions can be used together to isolate objects that are just touching. ➤ *Figure 19.55* illustrates this on a binary image containing four objects. (The same four objects but isolated from one another are shown in ➤ *Fig. 19.25a*.) This is an important task since many objects are delivered for inspection, handling or machining in contact with one another. ➤ *Figure 19.56* shows how grass-fire and watershed have been used to segment a grey-scale image of cut cane ends.

19.11.3 'Frequency Response' of Morphological Filters

It is common practice to measure the frequency response of filters used for analysing time series. The same concept can also be useful when discussing image filters. However, morphological filters are grossly non-linear, so there can be no precise measurement of their frequency response (➤ *Fig. 19.57*). For this reason, we can only relate their behaviour to spatial frequency in an approximate way. Nevertheless, we can gain some useful insight by discussing frequency and wavelength. Remember that what follows in this section is merely an aid to understanding; it is not precise.

➤ *Figure 19.58a* shows an image in which the intensity varies as a sine-wave whose frequency varies along the horizontal axis. Such an image can generated using an equation of the following general form:

$$f(x,y) = 127.5 * (1 + \sin(2\pi k(a + bx))), \text{for all } y$$

Here, a and b are constants. This equation has been normalised so that the intensity $f(x, y)$ always lies in the range

$$0 \leq f(x,y) \leq 255$$

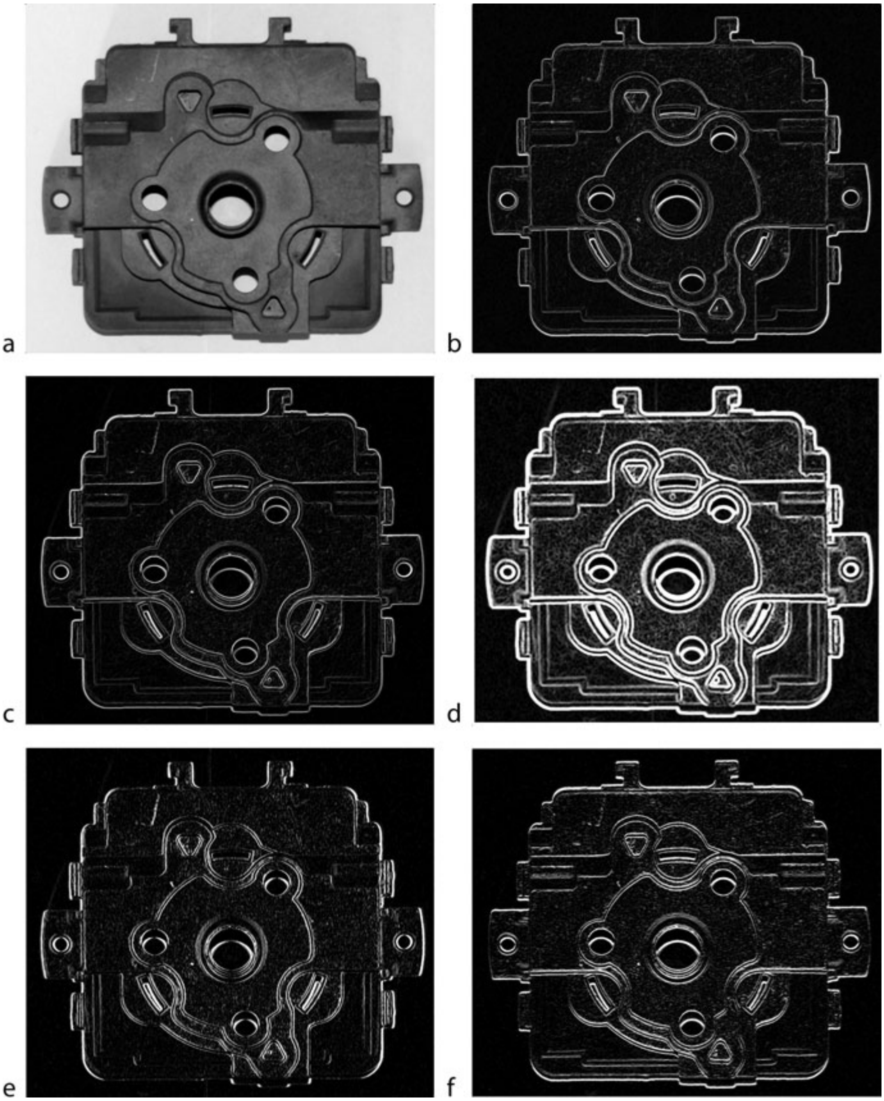


Fig. 19.48

Grey-scale edge detection. (a) Original. Zinc die-cast component. (b) Dilation, then subtract original. Circle SE. [QT: *x = gsk(14,2), gsd(x), sub, enc, neg, din, din*] (c) Erosion, then subtract original. Circle SE. [QT: *x = gsk(14,2), gse(x), sub, enc, din, din*] (d) Erosion, then subtract original. Large circle SE. [QT: *x = gsk(14,8), gse(x), sub, enc, din, din*] (e) Erosion with horizontal line SE, then subtract original. [QT: *x = gsk(1,5), gse(x), sub, enc, din, din, din*] (f) Erosion with vertical line SE, then subtract original. [QT: *x = gsk(2,5), gse(x), sub, enc, din, din, din*]

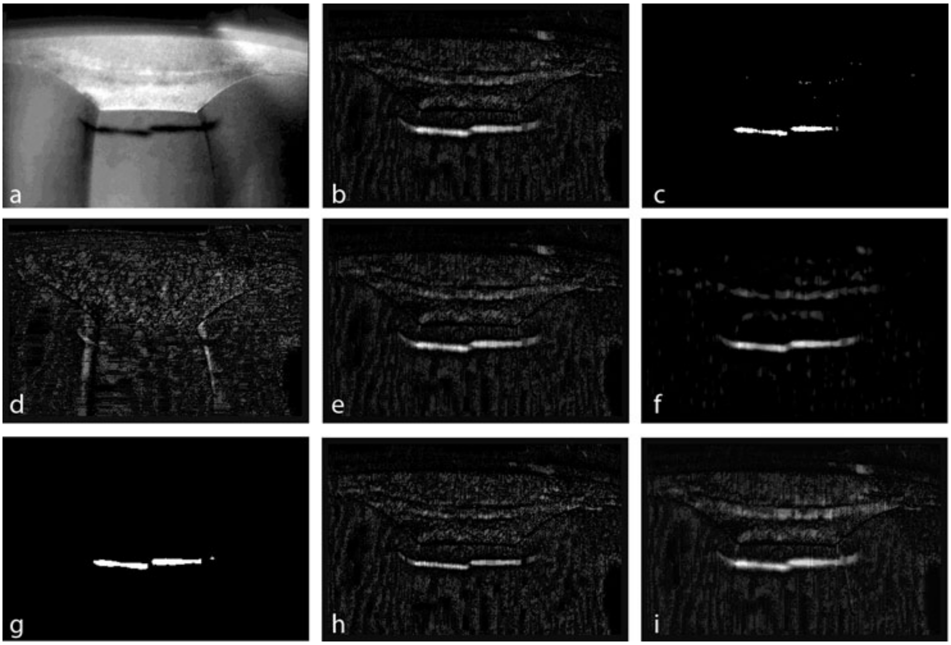


Fig. 19.49

Crack detector (Closing) (a) Original, cracked ferrous component. Crack highlighted with dye penetrant. Illumination UV plus visible. (b) Circle SE, diameter 13 pixels. [QT: *wri, x = gsk(14,13); gsd(x), gse(x), rei, sub, edg(8,avg), enc*] (c) Previous image thresholded. Parameter selected manually. (d) Horizontal line SE, length 13 pixels. (e) Vertical line SE, length 13 pixels. (f) Vertical line SE, length 13 pixels. [QT: *wri, x = gsk(2,13); gsd(x), gse(x), rei, sub, edg(8,avg), enc*] (g) Median filter applied before closing. Vertical line SE, length 13 pixels. [QT: *mdf(7), wri, x = gsk (2,13); gsd(x), gse(x), rei, sub, edg(8,avg), enc*] (h) Previous image thresholded. Parameter selected manually. (i) Circle SE, diameter 9 pixels. (j) Circle SE, diameter 19 pixels

Applying grey-scale dilation to such an image, with a horizontal line structuring element of appropriate length, causes the stripes on the right-hand side to merge into a bright block (➋ *Fig. 19.58b*). While the bright stripes on the left do not merge, they do become narrower as a result of dilation. Applying grey-scale closing instead is not particularly helpful, since the bright block on the right remains bright (➋ *Fig. 19.58c*). Erosion creates a dark block on the right (➋ *Fig. 19.58d*). Adding the results of dilation and erosion creates a filter that approximates a low-pass filter (➋ *Fig. 19.58e*). Varying the length of the line SE alters the 'cut-off' frequency (➋ *Fig. 19.58f*).

Another way to express the same idea is that narrow streaks can be eliminated by a 'low-pass' filter, thereby differentiating them from wider ones. By subtracting the output of a 'low-pass' filter from the original image, we can produce a 'high-pass' filter. An example of this is shown in ➋ *Fig. 19.51*, where a series of light stripes have been projected onto the top surface of a loaf. Tracing the light stripes is a prelude to measuring its 3D shape. (This technique is called *Triangulation by Light Sectioning* and is described in ➋ Chaps. 14 and ➋ 41.) Notice that median filtering and majority voting are also used, for eliminating the effects of noise.

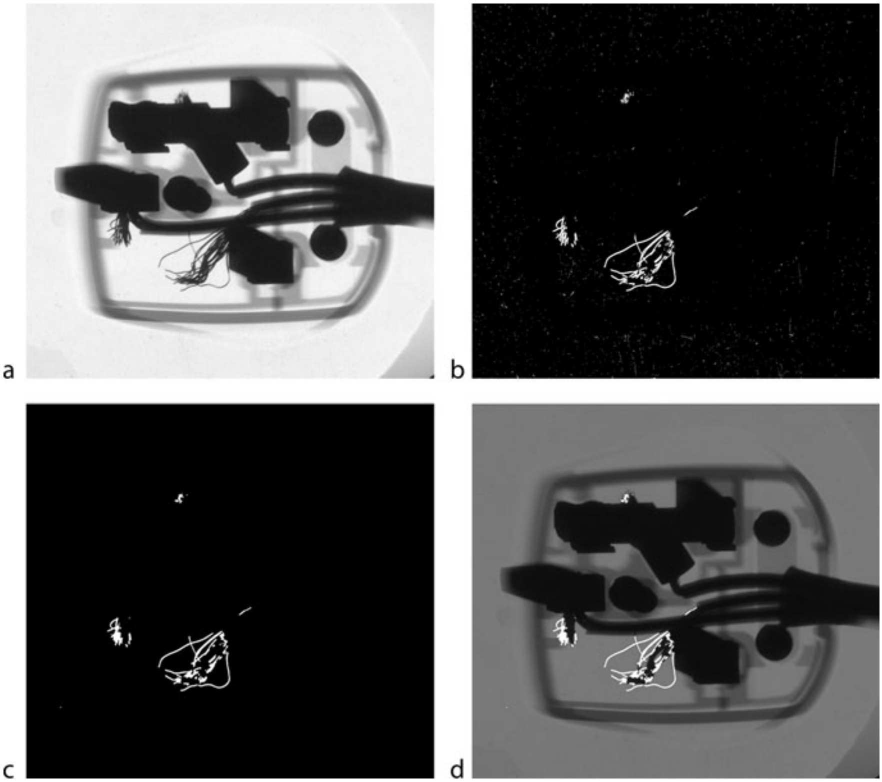


Fig. 19.50

Detecting thin dark streaks due to shadows of wires using grey-scale morphology. (a) Original image, x-ray of a domestic power plug (British design, 240V//13A). The loose strands of wire make this dangerous. Resolution: 1454 × 1336 pixels. (b) Grey-scale closing. [QT: *crk(5)*] (c) After removal of isolated white points [QT: *cln*] (d) Previous image superimposed on the original. It is not apparent in this printed image that the algorithm has correctly detected all individual loose strands and no other features

Interpreting the 1-pixel wide contours is not our concern here; let it suffice to say that some 'intelligence' is needed and may even require Prolog reasoning (Chap. 23).

So far, we have considered only 1D filtering. We can apply similar ideas to 2D patterns, perhaps filtering in quite different ways along the two image axes. We could, for example, apply a low-pass filter along one axis and high-pass filter along the orthogonal axis. To investigate the behaviour of a filter, it is helpful to devise a suitable test image. One possibility is shown in *Fig. 19.59a*. A variety of images of this general form, covering different frequency ranges along the image axes, can be generated; the reader must be prepared to exercise his imagination and judgement.

In *Fig. 19.59*, 'isotropic' 2D filters using simple rectangle and circle SEs are illustrated. *Figure 19.60* demonstrates non-isotropic filtering on the same test image. Notice the way that the various parts of the sine-wave pattern are blurred in different ways by these filters. The important lesson to be noted from these two figures is that test images can provide a valuable aid to understanding. When a rectangle SE is used (*Fig. 19.59b–h*), the two image axes can be processed independently and in either order, to obtain the same result.

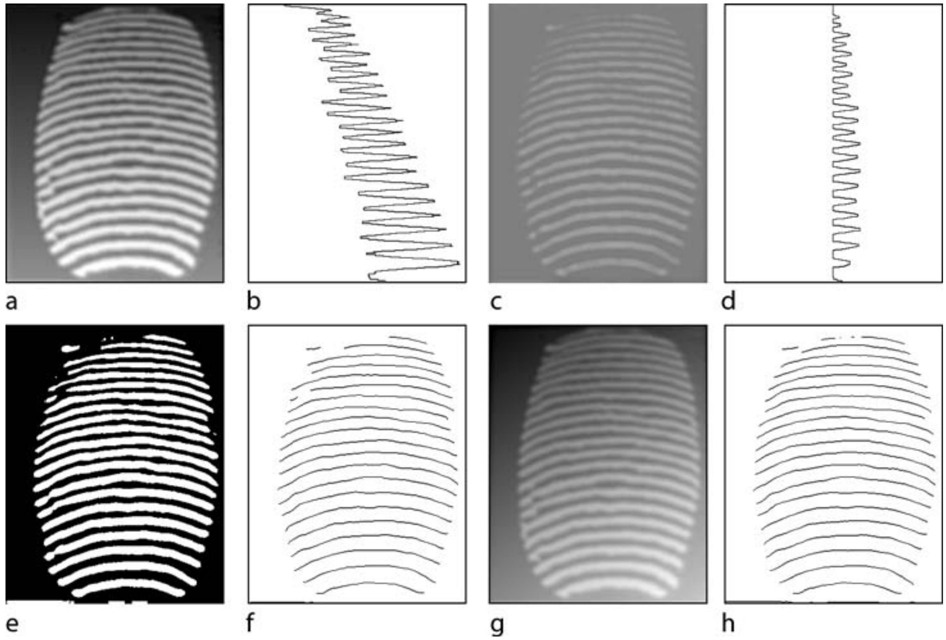


Fig. 19.51

Loaf shape analysis using grey-scale morphology. (a) Image contrived for this demonstration, using light sectioning and triangulation. A series of parallel light stripes was projected onto the top surface of a loaf. The camera viewed the loaf from a different angle. The resulting image had a high contrast and could be thresholded to produce a 'clean' binary image. However, this does not properly indicate the range of variation that can occur with other loaves that are over- or under-baked locally. In addition, variations of the intensity of the projected light stripes and ambient light make fixed-level thresholding unreliable. To demonstrate the effectiveness of the filtering better, an image whose intensity varies with vertical position was added. ('intensity wedge'; QT: *wgy*) (b) Intensity profile of (a). Notice that no fixed threshold could properly detect all of the peaks in this graph. (c) After median filtering and grey-scale closing with a vertical line structuring element of length 15 pixels. (The image has 325 × 415 pixels.) (d) Intensity profile of image (c). Although this has a low contrast, there is no variation of background intensity, so thresholding can be more effective. (e) After thresholding (c). The threshold parameter was selected manually but its value is not critical and does not depend on the light level. (f) Majority voting and thinning applied to the image in (e). (g) Another synthesised image, this time with vertical and horizontal wedges added. [QT: wgx, adi, wgy, adi]. (h) The processing just described was applied to (g). [QT: *wgx, adi, wgy, adi, mdf(7), wri; x= gsk(2,15), gse(x), gsd(x), rei, sub, neg, thr (a), maj, thn*]

This makes it easier to visualise what the filter does and to design one with characteristics that we desire. However, this is not the case for the circle and non-paraxial line SEs. Anticipating the results of some filters is less straightforward, making experimentation essential.

While it is easy to relate the behaviour of a simple morphological filter, albeit approximately, to the familiar concepts of time-series analysis, it also worthwhile remembering that

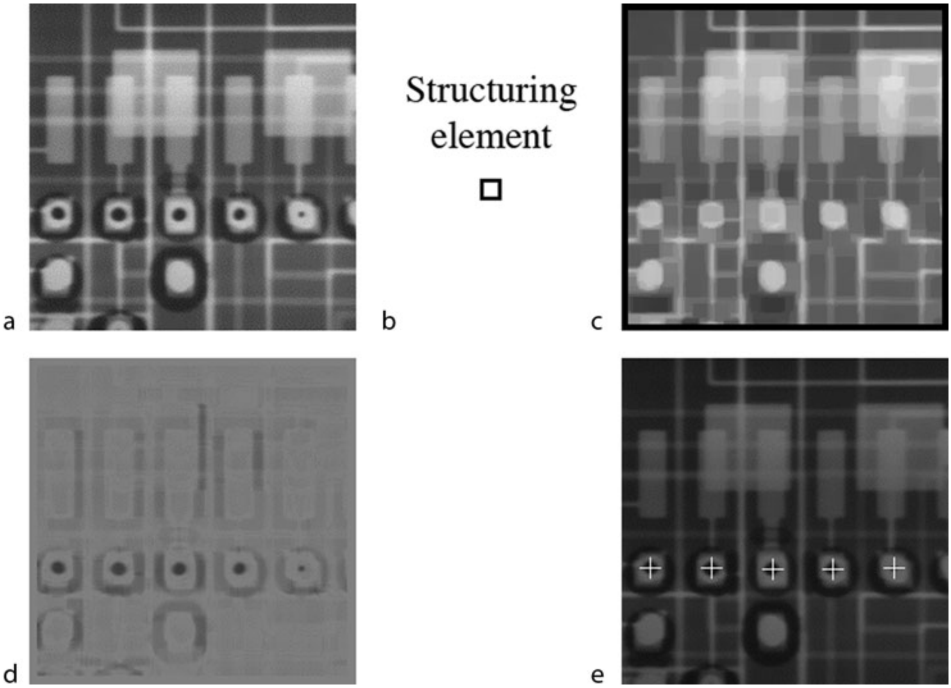


Fig. 19.52
Grey-scale closing (a) Original image, printed circuit board, imaged using x-rays. The grainy appearance is sensor noise in the x-ray system. (b) Structuring element: 11 × 11 pixel 'hollow' square. (c) Closing. (d) Subtracting (a) and (c). (e) Centres of the dark spots. The previous image was thresholded, then the resulting blobs were shrunk to single points and a cross was centred on each white pixel. [QT: *x = ones(11,11); gsc(x), sub, edg(6,128), thr(0,67), dil, shk, crs(8)*]

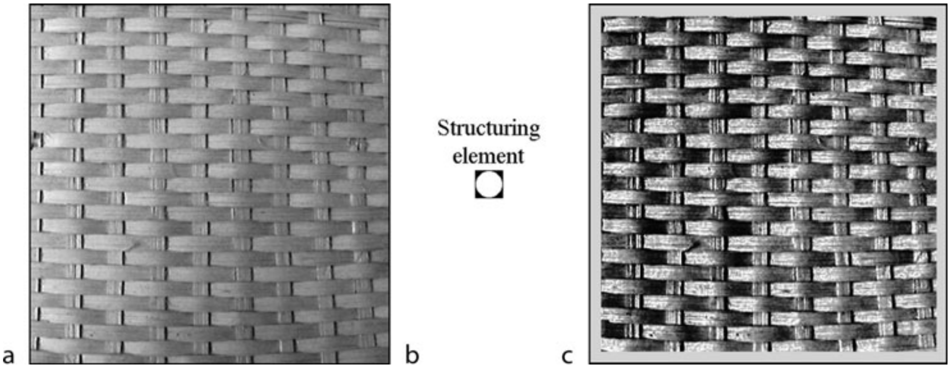


Fig. 19.53
Grey-scale morphology used to enhance contrast. (a) Original image. Rattan cane webbing for furniture. (b) Structuring element. (c) Median filtering within a 5 × 5 neighborhood was applied first, to reduce the noise level. Then, grey-scale closing was applied and the result subtracted from the original. Edge effects were eliminated by masking

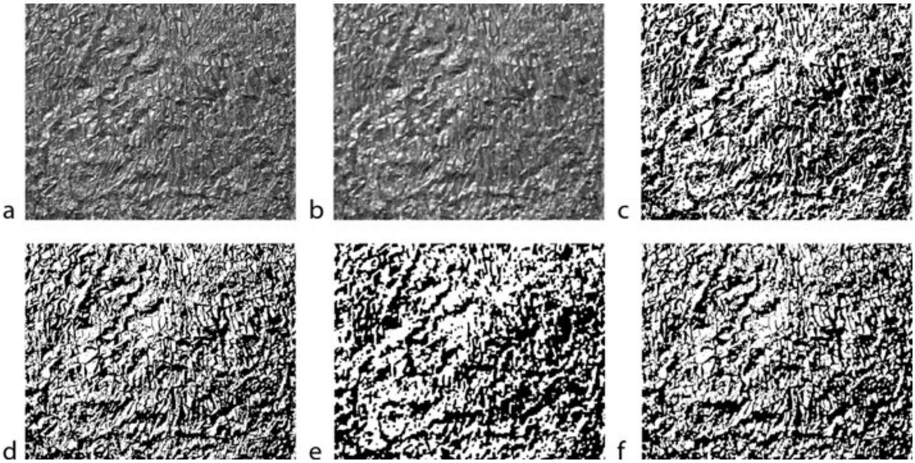


Fig. 19.54
Analysing grey-scale texture using morphology. (a) Original image. (b) Closing using a 5 × 5 flat rectangular structuring element, followed by erosion using the same SE. (c) Thresholding previous image so that 50% is black. (d) Opening using a 5 × 5 pixel flat rectangular structuring element, followed by thresholding. (e) Closing using a 7 × 7 pixel flat rectangular structuring element, followed by opening with the same SE, then thresholding. (f) Opening using a 7 × 7 pixel flat rectangular structuring element, followed by thresholding

wavelength is proportional to the reciprocal of frequency. We have seen that to eliminate a dark gap spanning N pixels, the SE used for dilation must be at least N/2 pixels wide. Hence, if the 'wavelength' of a periodic structure, along a line inclined at θ° to one of the image axes, is N pixels, dilation with an SE of width greater than N/2 pixels will eliminate it. (We measure the width of the SE along the line at θ°.) There may be another periodic structure within the image, at a different orientation. Often this is $(\theta^\circ + 90^\circ)$, in which case the task of designing a filter is simplified.

This discussion relating frequency and wavelength to morphological filters behaviour is approximate and breaks down completely for SEs with complicated shapes. We should not, for example, try to pursue this analogy for SEs with shapes such as V, T or X.

19.12 Concluding Remarks

Morphology provides a valuable class of image processing functions and can be used in a variety of ways. The following is an abbreviated list:

- Feature detection, as a prelude to
  - Location of features or objects
  - Image registration
  - Dimensional measurement
  - Existential inspection
  - Object identification
  - Robotic object handling

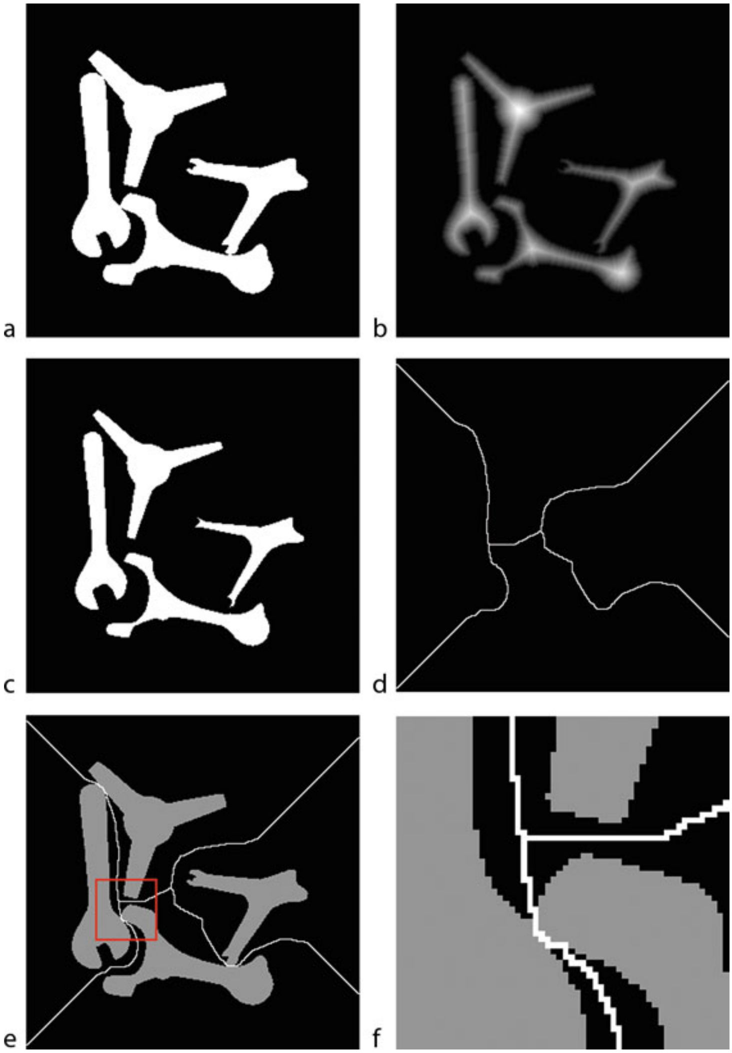


Fig. 19.55

Using the grass-fire and watershed functions to isolate objects that are just touching. (a) Original image. Notice that there are three points of minimal contact among the four objects. (The isolated objects are shown in *Fig. 19.25a*) (b) Grass-fire transform, after contrast enhancement. (c) Thresholded. (d) Watershed. (e) Watershed superimposed on the original. (f) Detail within the red square in (e)

  - Counting objects
  - Automated machining
- Detection of objects whose orientation falls within a limited range of angles
- Image filtering to perform
  - Contrast enhancement
  - Detection or measurement of periodic patterns

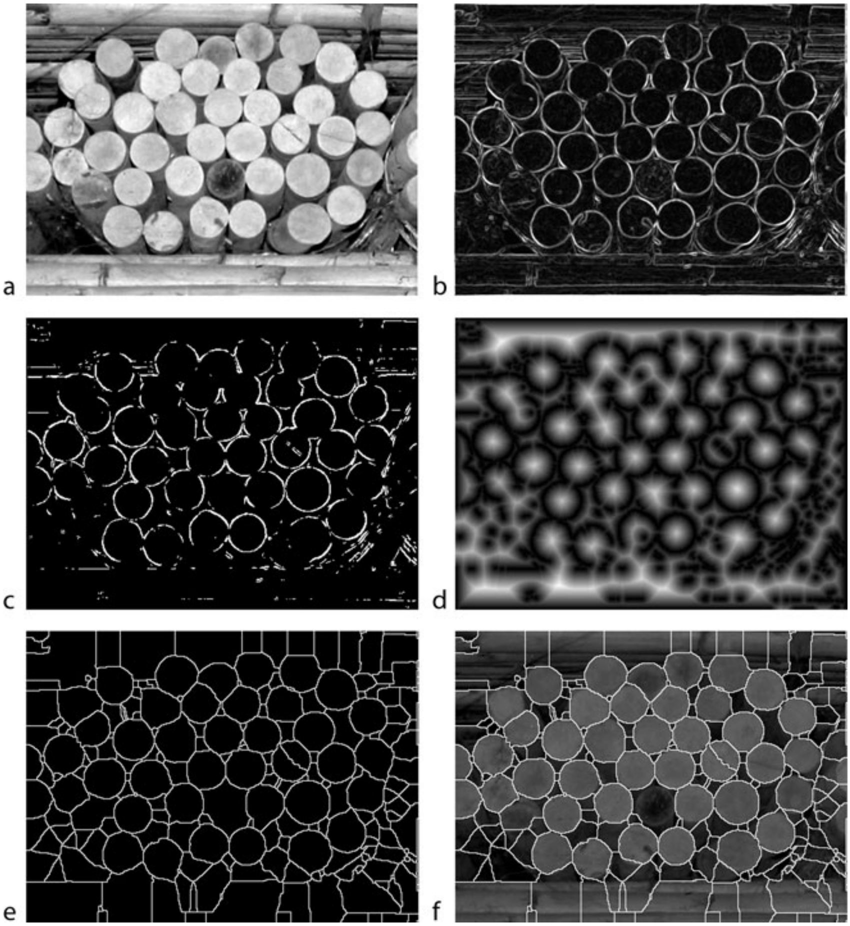


Fig. 19.56

Segmenting an image using the grass-fire and watershed functions. (a) Original image, a bundle of rattan canes viewed end-on. (b) Edge detector. (c) Thesholding. (d) Grass-fire transform. (e) Thresholding and watershed function. (f) Segmentation boundaries superimposed on the original. [QT: *rni(153), sed, thr(32), gft, enc, wsh, rni(original),swi,adi, mxi*]

  - Noise reduction
  - Texture analysis
  - Crack detection
  - Segmentation
  - Low-pass and high-pass filtering (approximate)
- Colour recognition
- Optical character recognition

Table 19.3 summarises the action of the more significant morphological functions and describes how they can be executed in MATLAB and QT.

The implementation of morphological functions to maximise their execution speed has been studied extensively over many years. Many important classes of SE lend themselves

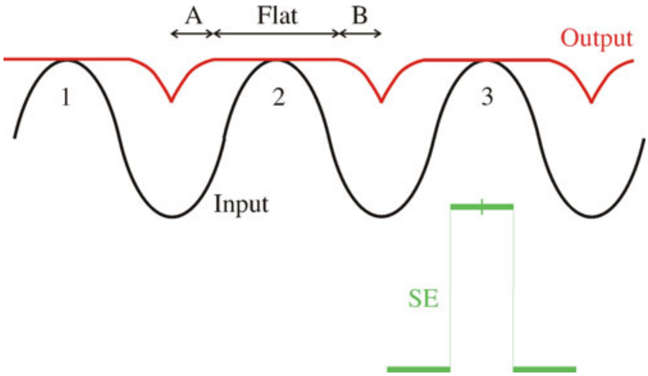


Fig. 19.57

Morphology is not a linear process. The sections of the output labelled ‘Flat, A, and B’ follow peak 2. The SE has a flat top. Grey scale dilation (or erosion) on a sine wave does not generate a sine wave. Since a multitude of high-order harmonics is generated by morphological filtering, the concept of ‘frequency response’ is strictly not valid

to the SKIPSM implementation described in ➲ Chap. 20. This includes shapes such as the following:

T L V O ∪ /— ∠ + X ▲ ▼ ♦ □

(These particular SEs can be implemented in other orientations.) Where direct SKIPSM implementation is not possible, it is usually possible to achieve the desired result easily, by breaking the SE into smaller units. The SKIPSM implementation is important because binary erosion and dilation morphological functions can be performed by accessing each pixel in the input image just once. This means that erosion and dilation can be performed at essentially the same speed as simple operations such as negation and thresholding. In many cases, it is possible to improve upon that, because several erosions and dilations can be performed at the same time.

Morphological structuring elements and convolution kernels are conceptually similar but they differ in the way that the combine the intensity values of the pixels they cover. Dilation uses the maximum (MAX) function; erosion uses minimum (MIN); convolution (local operators and linear N-tuple filters) uses summation. Consider the following equation

$$S_p(W_1, W_2, \ldots, W_N) = \sqrt[p]{\sum_{i=1}^{N} (W_i X_i)^p}$$

S_1 (i.e., $p = 1$) represents a simple summation of the terms $W_i X_i$ for $i = 1, 2, \ldots, N$. That is the type of calculation performed by convolution. As p approaches plus infinity and with all $W_i = 1$, S_p approximates $MAX(W_i X_i)$, which is the heart of the grey-scale dilation operation. To represent grey-scale erosion $MIN(W_i X_i)$, we let p approach minus infinity, again with all $W_i = 1$. The point is that in this one equation we can relate these three functions to one another. (Incidentally, it also suggests a range of other functions in which p is a finite number not equal to unity. Rank and range filters form another closely related family of operations, although they are not encompassed exactly by the above equation. However, we shall leave these ideas for the reader to explore.) The question arises as to why we should use any process other that

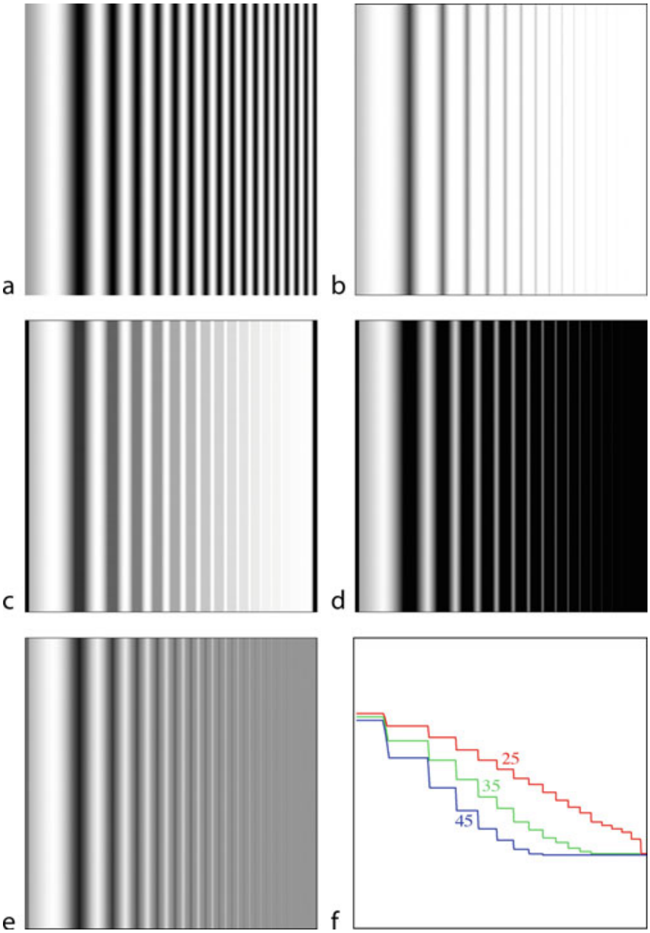
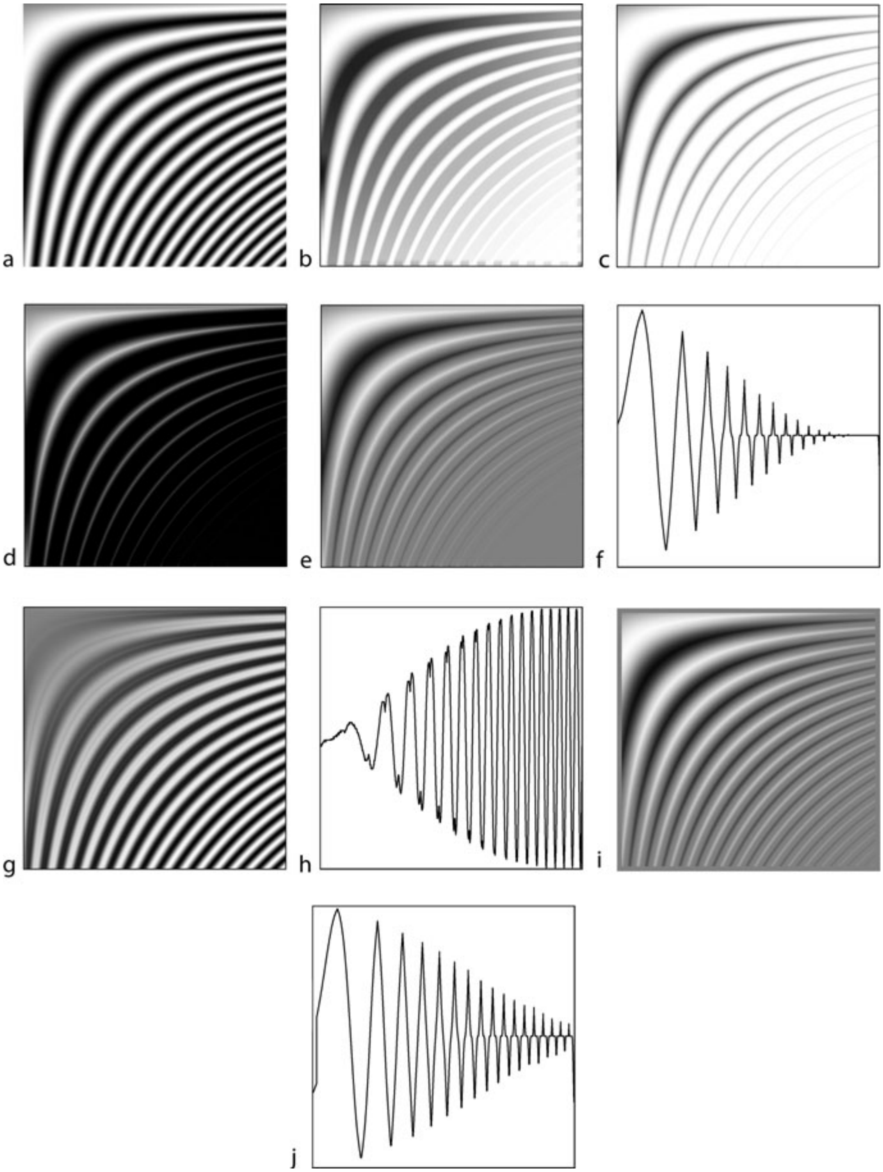


Fig. 19.58

Adjusting the frequency response of a morphological low-pass filter, by varying the size of the structuring element. (a) Variable frequency sinusoidal intensity variation. This image has 1024 × 1024 pixels. (b) Grey-scale dilation using a horizontal line SE of length 30 pixels. (c) Erosion of (b) using the same SE. (d) Erosion of the original image using the same SE. (e) Adding images (b) and (d). Summing the results of dilation and erosion approximates a low-pass filter. (f) The 'low-pass' filter was applied to image (a) and the local contrast measured as a function of horizontal position (i.e., spatial frequency). Three filter sizes were used to measure the strength of the filter output. Numbers indicate the lengths of three horizontal line SEs. As the SE length is increased, the filter attenuates ever lower frequencies. Both axes are linear

convolution, since it and morphology are simply variations of a single theme. Erosion is a *non-integrative* feature-detection process and is therefore highly sensitive to small 'spot' defects, whereas convolution is capable of ignoring them. Dilation is sensitive to isolated black pixels and erosion to isolated white pixels. For this reason, morphology is well suited to detecting small spots, or conversely to eliminating them. Dilation and erosion are capable of discriminating the polarity of 'spot' defects much better than convolution can. Simple unsharp masking



◘ Fig. 19.59

Grey-scale morphological low-pass and high-pass filtering in two dimensions. (a) Variable frequency sine waves. [QT: *vfs*] (b) Closing using a rectangular flat-top structuring element. [QT: *lnb(35), snb(35)*] (c) Dilation [QT: *lnb(35)*]. (d) Erosion. [QT: *snb(35)*] (e) Adding the results of dilation and erosion approximates low-pass filtering [QT: *lnb(35), wri(1), rei(original), snb(35); rei (1), adi*] (f) Intensity profile along the diagonal in (e), showing the reduction in amplitude (local intensity variation) as the frequeny rises. (g) Difference between images (e) and (a), produces an approximation to high-pass filtering. (h) Intensity profile along the diagonal in (g), showing the increase in amplitude as the frequeny rises. (i) Low-pass filter, using a circular disc SE. [QT: *x = gsk(14,18); gsd(x), wri(1), swi, gse(x),rei(1), adi*] (j) Intensity profile along the diagonal in (i)

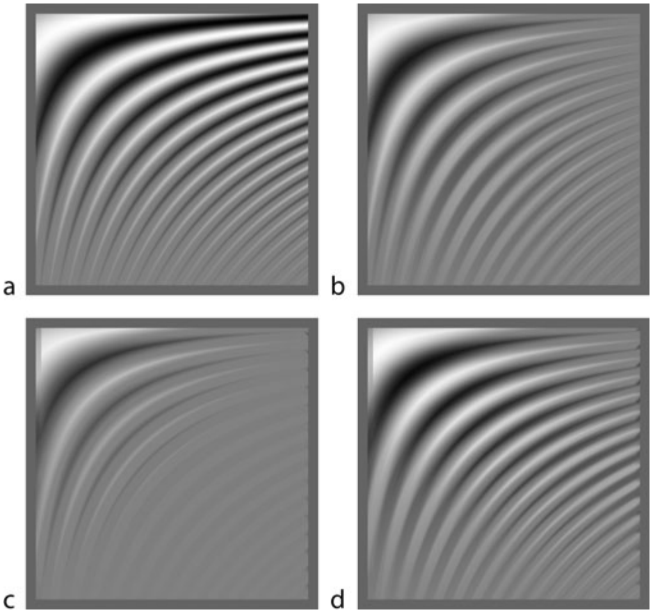


Fig. 19.60

Directional low-pass filtering using grey-scale morphology in two dimensions. In each case, the length of the structuring element is 55 pixels and the image has 1024 × 1024 pixels. Edge effects have been suppressed by masking. (a) SE is a horizontal line (b) SE is a vertical line. (b) SE is a diagonal line, top-left to bottom-right. (d) SE is a diagonal line, bottom-left to top-right

(convolution high-pass filter) produces a bright 'halo' around a dark spot or streak and is difficult to distinguish this from a dark spot. On the other hand, the morphological crack detector (bottom-hat) identifies dark spots and completely ignores well-spaced bright ones. Morphology is well suited to detecting features such as *parts* of printed characters, sharp corners, joints and limb ends of 'skeletons'. Convolution cannot do this with the same ease.

Programs for performing morphological image processing operations are available from many sources. Most commercial software libraries for image processing possess a collection of these functions. QT (➲ Chap. 21) and MATLAB® were used to generate many of the illustrations in this chapter. NeatVision (➲ Chap. 22) provides a comprehensive set of morphological functions, while Photoshop® also has basic erosion and dilation functions.

19.13 Supplement: Definitions of Morphological Operators

We shall use the following notation:

- E, an integer grid
- A, a binary image in E.
- A^C, the complement of A
- B, a structuring element, also in E
- B_Z, the translation of B by the two-element vector Z
- B^S, the rotation of B by 180°

Table 19.3

Common morphological functions, with their implementation in MATLAB and QT (➔ Chap. 21)

Binary edge detection	
Function MATLAB QT Figures	Removes interior pixels. Only object boundary pixels are set to white *new_image = bwperim(old_image,n), n = 4 or 8.* *bed* ➔ *19.20*, ➔ *19.34*
Bottom Hat	
Function MATLAB QT Figures	The output image is the difference between the original image and the result of closing it. *new_image = bwmorph(old_image,'bothat',n), n* is the iteration limit. *btt(n)* ➔ *19.20*, ➔ *19.32*, ➔ *19.49*, ➔ *19.50*, ➔ *19.51*
Bridge	
Function MATLAB *QT* Figure	Black pixels are set to white if they have exactly two white neighbours that are not connected. White pixels remain white. Example: 1 0 0 1 1 0 1 0 1 becomes 1 1 1 0 0 1 0 1 1 *new_image = bwmorph(old_image, 'bridge')* *mbo* ➔ *19.22*
Cleaning	
Function MATLAB QT Figure	Remove white isolated pixels *new_image = bwmorp(old_image, 'clean')* *cln* ➔ *19.19*
Closing	
Function MATLAB QT Figures	Performs morphological closing. That is *dilation* followed by *erosion*, using the same structuring element *new_image = imclose(old_image,se)* *cls(n)* Closing, n selects one of the pre-defined structuring elements *gsc(se)* Grey-scale closing, with structuring element *se*. ➔ *19.11*, ➔ *19.19*, ➔ *19.33*, ➔ *19.49*, ➔ *19.51*, ➔ *19.53*
Count white neighbours	
Function QT Figure	Count the number of white pixels within each 3 × 3 neighbourhood white. *cnw* ➔ *19.21*
Detect pattern	
Function MATLAB *QT* Figures	Retain each pixel at the centre of a given pattern of white eight-neighbours. This pattern is detected in whatever rotations/reflections are defined when constructing a look-up table, *lut*. *new_image = applylut(old_image,lut),* where *lu*t is a look-up table created by *makelut.* sbc, counts the number of white pixels in each 3 × 3 window, including the centre pixel. ➔ *19.27*, ➔ *19.28*

Table 19.3 (Continued)

Diagonal fill		
Function	Uses diagonal fill to eliminate 8-connectivity of the background. Example: $\begin{matrix}0 & 1 & 0\\ 1 & 0 & 0\\ 0 & 0 & 0\end{matrix}$ becomes $\begin{matrix}0 & 1 & 0\\ 1 & 1 & 0\\ 0 & 0 & 0\end{matrix}$	
MATLAB	new_image = bwmorph(old_image, *bwmorp, ‘diag’)*	
Dilation		
Function	Expand white/bright regions	
MATLAB	*dilate*	Perform dilation on binary image.
	imdilate	Dilate image.
QT	*dil*	Dilate with one of four pre-defined structuring elements
	gbd	General binary dilation (Use *cse* or *use* to define the SE.)
	gsd	Grey scale dilation (Use with *gsk*, or *ones* to define the SE.)
	crs	Dilate so that an isolated white point becomes a vertical cross.
	dcr	Dilate so that an isolated white point becomes a diagonal cross.
	dil	Dilate with one of four pre-defined structuring elements.
	dri	Dilate with a*b rectangular structuring element.
	dsq	Dilate so that an isolated white point becomes a square.
Figures	➔ *19.3*, ➔ *19.4*, ➔ *19.7–19.12*, ➔ *19.16*, ➔ *19.29*, ➔ *19.31*, ➔ *19.32*, ➔ *19.34*, ➔ *19.35*, ➔ *19.37*, ➔ *19.45*, ➔ *19.48*, ➔ *19.57*, ➔ *19.58*	
Erosion		
Function	Erosion; shrink white/bright regions	
QT	ero	Erode with one of four pre-defined structuring elements.
	gbe	General binary erosion
Figures	➔ *19.3*, ➔ *19.4*, ➔ *19.6*, ➔ *19.8–19.18*, ➔ *19.27–19.30*, ➔ *19.32–19.35*, ➔ *19.37*, ➔ *19.40*, ➔ *19.41*, ➔ *19.45–19.48*, ➔ *19.58–19.60*	
Eliminate small objects		
Function	Remove small binary objects	
MATLAB	*new_image = bwmorph (old_image, ‘fill’)*, removes isolated black pixels *new_image = bwareaopen (old_image,n,c)*, remove c-connected (c = 4 or 8) objects with fewer that n pixels.	
QT	*cln*, remove isolated white pixels *kgr(n,c)*, remove objects with fewer than *n* white pixels	
Figure	➔ *19.19*	
Euler number		
Function	Returns a number equal to the difference between the number of binary objects (blobs) and the number of lakes (holes). Although this is not a true morphological operation, it is computed in a similar manner, requiring the use of four or eight binary masks.	
MATLAB	*euler_number = bweuler(old_image,n), n =* 4 or 8	
QT	*euler_number = eul(n), n = 4 or 8*	

Table 19.3 (Continued)

Grass-fire transform	
Function	Intensity at each point within a binary object indicates the distance from that point to the nearest edge pixel. Background pixels remain black. Although this is not a true morphological operation, it is computed in a similar manner.
MATLAB	*new_image = bwdist(old_image,'euclidean')*
QT	*gft(n), n = 4 or 8*
Figures	➔ *19.18*, ➔ *19.23*, ➔ *19.55*, ➔ *19.56*
Grey-scale edge detection	
Function	Highlight regions where the intensity gradient is high
QT	*gse(x), sub* *gsd(x), sub*
Figure	➔ *19.48*
Majority	
Function	Sets a pixel to 1 if five or more pixels in its 3-by-3 neighbourhood are 1s; otherwise, it sets the pixel to 0.
MATLAB	*bwmorph, option = 'majority'*
QT	*maj*
Figures	➔ *19.19*, ➔ *19.38*, ➔ *19.51*
Opening	
Function	Performs morphological opening (erosion followed by dilation).
MATLAB	*bwmorph, option =* 'open'
QT	*opn* Binary opening. (Use *cse* or *use* to define the SE.) *gso* Grey-scale opening. (Use with *gsk*, or *ones* to define the SE)
Figures	➔ *19.10*, ➔ *19.12*, ➔ *19.19*, ➔ *19.33*, ➔ *19.50*, ➔ *19.53*
Shrink	
Function	With n = inf, shrinks objects to points. It removes pixels so that objects without holes shrink to a point, and objects with holes shrink to a connected ring halfway between each hole and the outer boundary. This option preserves the Euler number.
MATLAB	*new_image = bwmorph(old_image, 'shrink',n)*
QT	*shk*
Figures	➔ *19.23*, ➔ *19.33*, ➔ *19.36*, ➔ *19.47*, ➔ *19.50*
Skeleton joints/Branch points	
Function	Detect the joints of a skeleton; retain only those white pixels that have three or more white 8-neighbours.
MATLAB	See *Detect Pattern*
QT	*jnt*
Figure	➔ *19.23*
Skeleton limb ends	
Function	Detect the ends of the limbs of a skeleton; retain only those white pixels that have just one white eight-neighbour.
MATLAB	*new_image = bwmorph(old_image, 'endpoints')*
QT	*lmb*
Figure	➔ *19.23*

Table 19.3 (Continued)

Spurs	
Function	Remove spur pixels, which are white pixels at the ends of skeleton limbs, applied recursively.
MATLAB	*new_image = bwmorph(old_image, 'spur',n), n* is the recursion limit
QT	*spr(n)*
Figures	➔ *19.23*, ➔ *19.24*, ➔ *19.26*, ➔ *19.31*, ➔ *19.33*
Structuring elements	
Function	Create/select a structuring element.
MATLAB	strel
	Example *se = strel('disk',size), new_image = imdilate(old_image,se)*
QT	*use* User selects binary structuring element with cursor
	cse Select binary structuring element from set of options
	Example *x = cse(3,7), gbd(x)*
	gsk Select grey-scale structuring element from set of options.
	Example *x = gsk(3,7), gsd(x)*
Thicken	
Function	Thickens binary objects by adding pixels to the exterior of each one until doing so would result in previously unconnected objects becoming 8-connected. The Euler number remains unchanged.
MATLAB	*new_image = bwmorph(old_image, 'thicken',n), n* is a finite iteration limit.
QT	*thicken(n)*
Figures	➔ *19.25*, ➔ *19.26*
Thin	
Function	A binary object without lakes shrinks to a single point. An object with lakes shrinks to a figure consisting of a series of connected rings, surrounding each lake and the outer object boundary nearby. The Euler number remains unchanged.
MATLAB	*new_image = bwmorph(old_image,'thin',n), n* is a finite iteration limit.
QT	*thin(n)*
Figures	➔ *19.16*, ➔ *19.17*, ➔ *19.23*, ➔ *19.31*, ➔ *19.40*
Top Hat	
Function	The output image is the difference between the original image and the result of opening it.
MATLAB	*new_image = bwmorph(current,'tophat',se)*
QT	*tpt(a), a = 0 or 1*
Figures	➔ *19.19*, ➔ *19.20*, ➔ *19.32*, ➔ *19.33*, ➔ *19.41*, ➔ *19.52*
Ultimate thin	
Function	A binary object without lakes shrinks to a single point. An object with lakes shrinks to a figure consisting of a series of connected rings, surrounding each lake and the outer object boundary nearby. The Euler number remains unchanged.
MATLAB	*new_image = bwmorph(old_image,'thin',inf)*
QT	*thn*
Figures	➔ *19.23*, ➔ *19.30*, ➔ *19.56*

Table 19.3 (Continued)

Watershed	
Function	The image resembles a honey-comb; each 'cell' consists of a 4- or 8-connected set of points. Within each cell every point perceives the same blob in the input image as its nearest neighbour.
MATLAB	*new_image* = watershed*(old_image,n), n* = 4 or 8
QT	*wsh(n), n = 4 or 8*
Figures	❯ *19.26*, ❯ *19.38*

19.13.1 Erosion

$$A \,\Theta\, B = \{ z \in E | B_{\approx} \subseteq A \}$$

or

$$A \,\Theta\, B = \bigcap_{b \in B} A_{-b}$$

19.13.2 Dilation

$$A \oplus B = \left\{ z \in E | (B^s)_{\approx} \cap A \neq \emptyset \right\}$$

where *Bs* is given by

$$B^s = \{ x \in E | -x \in B \}.$$

Alternatively

$$A \oplus B = \bigcup_{b \in B} A_b$$

19.13.3 Opening

$$(A \circ B) = (A \Theta B) \oplus B$$

19.13.4 Closing

$$A \bullet B = (A \oplus B) \,\Theta\, B$$

19.13.5 Properties

- Dilation, erosion, opening and closing are all position invariant. The result on each blob is the same irrespective of where it is in the image.
- Dilation and erosion are increasing. So, if

$$A \subseteq C$$

then

$$A \oplus B \subseteq C \oplus B$$

$$A \Theta B_b \subseteq C \,\Theta\, B$$

- Opening and closing are also increasing.
- Dilation is commutative.
- Dilation is associative, so

$$(A \oplus B) \oplus C = A \oplus (B \oplus C).$$

Erosion satisfies

$$(A \Theta B) \Theta C = A \Theta (B \oplus C).$$

- Erosion and dilation satisfy the duality relationship

$$A \oplus B = (A^c \Theta B^s)^c.$$

- Opening and closing satisfy the duality relationship

$$A \bullet B = (A^c \circ B^s)^c.$$

- Dilation is distributive over set union
- Erosion is distributive over set intersection
- Dilation and erosion are pseudo-inverse of one another, in the sense that

$$A \subseteq (C \Theta B)$$

if and only if

$$(A \oplus B) \subseteq C.$$

- Opening and closing are idempotent
- Opening is anti-extensive, so

$$A \circ B \subseteq A$$

- Closing is extensive, so

$$A \subseteq A \bullet B.$$

References

1. Serra J (1982) Image analysis and mathematical morphology. Academic, London
2. Mathematical morphology, Wikipedia. http://en.wikipedia.org/wiki/Mathematical_morphology. Accessed 22 Sept 2010
3. Halle JAG, Saraga P (1978) The control of a printed circuit board drilling machine by visual feedback. In: Batchelor BG (ed) Pattern recognition. Plenum, London, pp 271–280. ISBN 0-306-31020-1
4. Floeder SP (1990) A reprogrammable image processing system. M.S. thesis, Electrical Engineering Department, University of Minnesota, Minneapolis
5. Hujanen AA, Waltz FM (1995) Pipelined implementation of binary skeletonization using finite-state machines. In: Proceedings SPIE Conference on machine vision applications in industrial inspection, vol 2423, paper No 2, San Jose, CA
6. Hujanen AA, Waltz FM (1995) Extending the SKIPSM binary skeletonization implementation. In: Proceedings SPIE Conference on machine vision applications, architectures, and systems integration IV, vol 2597, paper No 12, Philadelphia, PA
7. Waltz FM (1994) Application of SKIPSM to binary template matching. In: Proceedings SPIE conference. on machine vision applications, architectures, and systems integration III, vol 2347, paper No 39, Boston, MA
8. Waltz FM (1995) Application of SKIPSM to binary correlation. In: Proceedings SPIE conference on machine vision applications, architectures, and systems integration IV, vol 2597, paper No 11, Philadelphia, PA
9. Waltz FM, Miller JWV (2000) Software SKIPSM implementation for template matching. In: Proceedings SPIE conference on machine vision and three-

dimensional imaging systems for inspection and metrology, # 4189, Boston, MA

10. Waltz FM (1997) Implementation of SKIPSM for 3-D binary morphology. In: Proceedings SPIE conference on machine vision applications, architectures, and systems integration VI, vol 3205, paper No 13, Pittsburgh, PA

Further Reading

Batman S, Dougherty ER (1997) Size distributions for multivariate morphological granulometries. Opt Eng 36(5):1518–1529

Batman S, Dougherty ER, Sand F (2000) Heterogeneous morphological granulometries. Pattern Recogn 33(6):1047–1057

Barrera J, Banon GJF, Lotufo RA, Hirata R Jr (1998) MMach: a mathematical morphology toolbox for the Khoros system. J Electron Imaging 7(1):233–260

Barrera J, Dougherty ER, Tomita NS (1997) Automatic programming of binary morphological machines by design of statistically optimal operators in the context of computational learning theory. J Electron Imaging 6(1):54–67

Bertrand G (2005) On topological watersheds. J Math Imaging Vis 22(2–3):217–230

Beucher S, Meyer F (1993) The morphological approach to segmentation: the watershed transformation. In: Dougherty ER (ed) Mathematical morphology in image processing. Marcel Dekker, New York, pp 433–481, Chapter 12

Bieniel A, Moga A (2000) An efficient watershed algorithm based on connected components. Pattern Recogn 33:907–916

Bloch I, Maître H (1995) Fuzzy mathematical morphologies: a comparative study. Pattern Recogn 28(9):1341–1387

Chen Y, Dougherty ER (1994) Gray-scale morphological granulometric texture classification. Opt Eng 33(8):2713–2722

Couprie M, Najman L, Bertrand G (2005) Quasi-linear algorithms for the topological watershed. J Math Imaging Vis 22(2–3):231–249

Dougherty ER (ed) (1993) Mathematical morphology in image processing. Marcel Dekker, New York

Dougherty ER (1992) Euclidean gray-scale granulometries: representation and umbra inducement. J Math Imaging Vis 1:7–21

Dougherty ER, Astola JT (eds) (1999) Nonlinear filters for image processing. SPIE & IEEE, Bellingham

Dougherty ER, Astola J (1994) An introduction to nonlinear image processing. SPIE, Bellingham

Dougherty ER, Barrera J (2002) Pattern recognition theory in nonlinear signal processing. J Math Imaging Vis 16(3):181–197

Dougherty ER, Barrera J (1999) Logical image operators. In: Dougherty ER, Astola JT (eds) Nonlinear filters for image processing. SPIE & IEEE, Bellingham, pp 1–60

Dougherty ER, Barrera J (1999) Computational grayscale image operators. In: Dougherty ER, Astola JT (eds) Nonlinear filters for image processing. SPIE/IEEE, Bellingham, pp 61–98

Dougherty ER, Barrera J, Mozelle G, Kim S, Brun M (2001) Multiresolution analysis for optimal binary filters. J Math Imaging Vis 14(1):53–72

Dougherty ER, Chen Y (1999) Granulometric filters. In: Dougherty ER, Astola JT (eds) Nonlinear filters for image processing, SPIE/IEEE series on imaging science and engineering. IEEE, New York, pp 121–162

Dougherty ER, Chen Y (1997) Logical granulometric filtering in signalunion- clutter model. In: Goutsias J, Mahler R, Nguyen C (eds) Random sets: theory and applications. Springer, New York, pp 73–95

Dougherty ER, Chen Y (2001) Optimal and adaptive design of logical granulometric filters. In: Hawkes P (ed) Advances in imaging and electron physics, vol 117. Academic, New York, pp 1–71

Dougherty ER, Chen Y (1998) Logical structural filters. Opt Eng 37(6):1668–1676

Dougherty ER, Loce RP (1996) Optimal binary differencing filters: design, logic complexity, precision analysis, and application to digital document processing. J Electron Imaging 5(1):66–86

Dougherty ER, Loce RP (1993) Optimal mean-absolute-error hit-ormiss filters: morphological representation and estimation of the binary conditional expectation. Opt Eng 32(4):815–823

Dougherty ER, Loce RP (1993) Efficient design strategies for the optimal binary digital morphological filter: probabilities, constraints, and structuring-element libraries. In: Dougherty ER (ed) Mathematical morphology in image processing. Marcel Dekker, New York, pp 43–92, Chapter 2

Dougherty ER, Lotufo RA (2003) Hands-on morphological image processing. SPIE, Bellingham

Dougherty ER, Newell J, Pelz J (1992) Morphological texture-based maximum-likelihood pixel classification based on local granulometric moments. Pattern Recogn 25(10):1181–1198

Dougherty ER, Pelz J (1991) Morphological granulometric analysis of electrophotographic images – size distribution statistics for process control. Opt Eng 30(4):438–445

Dougherty ER, Pelz J, Sand F, Lent A (1992) Morphological image segmentation by local granulometric size distributions. J Electron Imaging 1(1):46–60

Dougherty ER, Sinha D (1994) Computational mathematical morphology. Signal Process 38:21–29

Giardina R, Dougherty ER (1988) Morphological methods in image and signal processing. Prentice-Hall, Englewood Cliffs, p 82

Goutsias J (1992) Morphological analysis of discrete random shapes. J Math Imaging Vis 2(2/3):193–215

Haralick RM, Sternberg SR, Zhuang X (1987) Image analysis using mathematical morphology. IEEE Trans Pattern Anal Mach Intell 9:532–550

Heijmans HJAM (1994) Morphological image operators. Academic, Boston

Heijmans HJAM (1997) Composing morphological filters. IEEE Trans Image Process 6(5):713–723

Heijmans HJAM (1999) Connected morphological operators for binary images. Comput Vis Image Und 73(1):99–120

Heijmans HJAM (1999) Introduction to connected operators. In: Dougherty ER, Astola JT (eds) Nonlinear filters for image processing. SPIE/IEEE, Bellingham, pp 207–235

Heijmans HJAM (1999) Easy recipes for morphological filters. In: Dougherty ER, Astola JT (eds) Nonlinear filters for image processing. SPIE/IEEE, Bellingham, pp 163–205

Heijmans HJAM, Nacken P, Toet A, Vincent L (1992) Graph morphology. J Vis Commun Image Rep 3:24–38

Heijmans HJAM, Vincent L (1993) Graph morphology in image analysis. In: Dougherty ER (ed) Mathematical morphology in image processing. Marcel Dekker, New York, pp 171–203, Chapter 6

Hirata NST, Dougherty ER, Barrera J (2000) Iterative design of morphological binary image operators. Opt Eng 39(12):3106–3123

Kuosmanen P, Koivisto P, Huttunen H, Astola J (1995) Shape preservation criteria and optimal soft morphological filtering. J Math Imaging Vis 5(4):319–335

Lotufo RA, Falcao AX (2000) The ordered queue and the optimality of the watershed approaches. In: Goutsias J, Vincent L, Bloomberg D (eds) Mathematical morphology and its application to image and signal processing, vol 12, Computational imaging vision. Kluwer, Dordrecht, pp 341–350

Maragos P (1989) Pattern spectrum and multiscale shape representation. IEEE Trans Pattern Anal Mach Intell 11:701–716

Maragos P, Schafer RW (1987) Morphological filters – part I: their settheoretic analysis and relations to linear shift-invariant filters. IEEE Trans Acoust Speech Signal Process 35:1153–1169

Maragos P, Schafer RW (1987) Morphological filters – part II: their relations to median, order-statistics, and stack filters. IEEE Trans Acoust Speech Signal Process 35:1170–1184

Marshall S, Sicuranza GL (eds) (2006) Advances in nonlinear signal and image processing, EURASIP book series on signal processing and communication. Hindawi, New York

Matheron G (1975) Random sets and integral geometry. Wiley, New York

Meyer F (1994) Topographic distance and watershed lines. Signal Process 38:113–125

Meyer F, Beucher S (1990) Morphological segmentation. J Vis Comm Image Rep 1(1):21–46

Rivest JF, Soille P, Beucher S (1993) Morphological gradients. J Electron Imaging 2(4):326–336

Roerdink JBTM, Meijster A (2000) The watershed transform: definitions, algorithms and parallelization strategies. Fundamenta Informaticae 41(1–2):187–228

Ronse C, Macq B (1991) Morphological shape and region description. Signal Process 25:91–105, 84

Salembier P (1992) Structuring element adaptation for morphological filters. J Vis Commun Image Rep 3(2):115–136

Salembier P (1994) Morphological multiscale segmentation for image coding. Signal Process 38(3):359–386

Salembier P, Serra J (1995) Flat zones filtering, connected operators, and filters by reconstruction. IEEE Trans Image Process 4(8):1153–1160

Salembier P, Brigger P, Casas JR, Pardas M (1996) Morphological operators for image and video compression. IEEE Trans Image Process 5(6):881–898

Sequeira RE, Preteux FJ (1997) Discrete Voronoi diagrams and the SKIZ operator: a dynamic algorithm. IEEE Trans Pattern Anal Mach Intell 19(10): 1165–1170

Serra J (ed) (1988) Image analysis and mathematical morphology, II: Theoretical Advances. Academic, London

Serra J (1988) Dilation and filtering for numerical functions. In: Serra J (ed) Image analysis and mathematical morphology, vol 2, Theoretical advances. Academic, New York

Serra J (1988) Introduction to morphological filters. In: Serra J (ed) Image analysis and mathematical morphology, vol 2, Theoretical advances. Academic, New York, Chapter 6

Serra J, Vincent L (1992) An overview of morphological filtering. IEEE Trans Circ Syst Signal Process 11:47–108

Sinha PS, Dougherty ER, Batman S (1997) Design and analysis of fuzzy morphological algorithms for image processing. IEEE Trans Fuzzy Syst 5(4):570–584

Sivakumar K, Goutsias J (1997) Discrete morphological size distributions and size densities: estimation techniques and applications. J Electron Imaging 6:31–53

20 Image Processing Using Finite-State Machines

Frederick M. Waltz
University of Minnesota, Minneapolis, MN, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_20,

Abstract: In a series of papers starting in 1994, Waltz described a powerful and unconventional method of implementing many important neighbourhood image processing operations. These techniques, taken together, have been given the acronym SKIPSM (Separated Kernel Image Processing using finite-State Machines). Note that "Separated" here does NOT imply that the operations must be linearly separable. The resulting implementations are almost always faster, and sometimes much faster, than the same operations implemented in conventional ways *on the same computational platforms.* This chapter presents the basic concepts of image processing using FSMs (finite-state machines), lists many specific image-processing operations which have been implemented with FSMs, gives examples of several typical operations (along with the C-code used to implement them), outlines a completely routine procedure for generating and optimizing SKIPSM C-code for a large class of image operations and neighbourhood sizes, and concludes with some execution-speed comparisons which support the claims of fast execution.

20.1 Introduction

In some applications of image processing, the execution speed of the various operations is not particularly important; e.g., in the development phase of machine-vision inspection algorithms, in which a trial-and-error approach is often used, as discussed elsewhere in this handbook. However, the *on-line applications* of those very algorithms sometimes require extremely fast execution. Years of experience have convinced the authors that, no matter how fast our image processors are, or how much their price/performance ratios decrease, there will always be applications for which they are either too slow or too expensive.

One can attempt to increase execution speed by choosing faster computational hardware. The law of diminishing returns operates with a vengeance here, in that choosing a microprocessor with a faster clock rate has a *minor* effect on execution speed, compared to the *architecture* of the chip; e.g., RISC versus CISC, bus speed, memory access speed, instruction and data pipelining, and internal data-path bit widths. Another way of saying this is that architectures optimized for general-purpose computation are often not well-adapted to image processing. At any given time, there will be a maximum speed achievable with standard hardware. This speed may not be fast enough for the application being considered. Therefore, any implementation scheme which allows image processing operations to execute faster *on a given platform* will be useful.

Once a computational platform has been chosen, the conventional way to improve execution speed is to expend the extra effort to write highly efficient software code; e.g., write the algorithm in assembly language, or write "fall-through" non-branching code to eliminate branches and loops. An expert programmer using this approach might cut execution time in half (or even less) compared to the same operation written by the *same* expert programmer in an intermediate-level language such as C. Beyond a certain point, however, great additional effort often achieves only minor improvements. And once again, the resulting implementation may not be fast enough for the application being considered.

Finally, there is the "special tricks" approach – formulating the problem in an unconventional way in order to take advantage of some *special property* of that *particular operation.* The literature is full of these, and one often marvels at the ingenuity of the people who first thought of them. Of course, things that were "special tricks" when first presented, if they are in fact better, soon become the standard way of doing those operations, and we may forget our initial admiration for their inventors. A good (if ancient) example is the two-dimensional Fast

Fourier Transform: There are some ingenious "tricks" involving the shuffling of data and coefficients which contribute to the speed improvements, but the *important* speed improvement results from the fact that Fourier operations are *linearly separable* – i.e., can be implemented using a row operation followed by a column operation.

The implementations described here use finite-state machines (FSMs), recursion, and row-column separation (of a type *much* more general than linear separability) to carry out image-processing operations. At first glance, one might suppose that these belong in the "special tricks" category. They are certainly extremely unconventional, but they don't fit the "special tricks" definition because *the same basic FSM techniques can be used for a very wide range of image-processing operations*. In fact, they constitute a radically new image-processing paradigm.

This new paradigm was originally developed in about 1990 as a technique for increasing the speed and more importantly the *versatility* of commercially-available pipelined image-processing hardware. As general-purpose computers became faster and began to be used for on-line inspection, it became clear that the widest application of the FSM approach would be for speeding up *software* image-processing programs running on PCs. This chapter therefore concentrates on software implementations written in the C language. Because this paradigm is fundamentally different from conventional image processing algorithms, we begin with a general overview of the method. This is followed by some examples selected from the wide range of operations that can be implemented in this way. Finally, execution-speed comparisons for some of these examples are presented.

Overview of the SKIPSM paradigm: In a series of papers starting in 1994, Waltz (1 through 6) described a powerful and radically new method of implementing many important neighbourhood image processing operations. These techniques, taken together, have been given the acronym SKIPSM (Separated Kernel Image Processing using finite-State Machines). The resulting implementations are almost always faster, and sometimes much faster, than the same operations implemented in conventional ways *on the same computational platforms*. In some cases the improvements in performance (speed or neighbourhood size) are so great, compared to conventional implementations, as to demand a radical rethinking of what is or is not practical. For example, operations such as binary erosion with very large structuring elements (e.g., 51 × 51 = 2,601 pixels), have been implemented *faster* on ordinary desktop computers than these same computers can do a simple 3 × 3 binary erosion (9 pixels) using conventional programming – a speed-improvement ratio of 289:1. Speed improvements or neighbourhood-size increases of 100:1 or more have also been achieved for some other image processing operations.

The image processing categories to which SKIPSM can be applied in software and/or special-purpose image-processing hardware implementations include but are not limited to the following:

- Binary erosion and dilation with large, arbitrary SEs (structuring elements). SEs up to 25 × 25 and larger and with "holes" and other non-convex shapes can be applied in a single image pass, in software or hardware. Multiple SEs can be applied simultaneously in a single pass [1–7]. For example, six erosion stages of the "Grassfire Transform" have been carried out in *one pass* [3].
- Binary opening and closing in a single pass [2, 23].
- Binary template matching with large, arbitrary templates [8].
- "Fuzzy" binary template matching [8].
- Binary correlation [9].

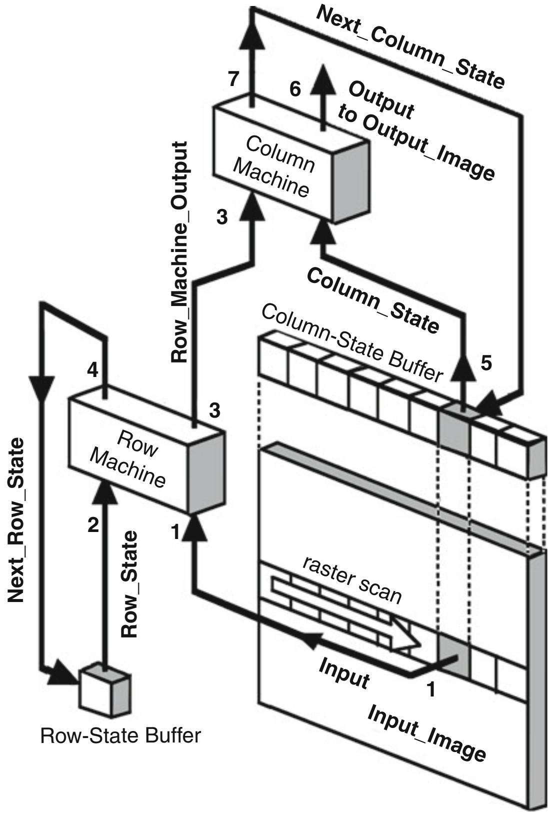
- Binary connected-component analysis, with assistance from a random-access processor for part of the problem [10].
- Grey-scale morphology [11, 12].
- Grey-scale template matching [11].
- Large-kernel Gaussian blurs, difference-of-Gaussian filters, etc. [13].
- Many standard linear neighbourhood operations: edge enhancement, area averaging, horizontal, vertical, and diagonal gradients [14].
- Many standard nonlinear neighbourhood operations: maximum, minimum, maximum of row minimums, minimum of row maximums, Robert's cross, median and other ranked filters, largest gradient, direction of largest gradient, direction of brightest or darkest neighbour, pixels critical for connectivity, all possible functions defined on 3 × 3 binary neighbourhoods, etc. [14–16].
- One-pass binary skeletonisation [17, 18].
- Binary run-length encoding on rows, columns, or diagonals [19].
- Grey-scale run-length encoding on rows, columns, or diagonals [19].
- Certain operations previously thought to be impossible in pipelined systems, such as "blob fill" and "patterned blob fill." [19].
- Various "smearing" operations, including row, column, and diagonal summations and Hough transforms [19].
- One-pass generation of certain standard images (e.g., grey-level wedges) and arbitrary image textures, either whole-image or masked arbitrarily [19].
- Grey-scale morphology and other operations on colour images [20].
- Operations in colour space [20].
- Three-dimensional and higher-dimensional binary morphology [24].

20.2 The SKIPSM Paradigm: Sequence of Operations

All SKIPSM applications for two-dimensional data arrays take the same general form (see ➲ *Fig. 20.1*) and have these characteristics:

- The overall operation is separated into a row operation followed by a column operation (or column operation, then row operation). *Note that the separability requirement is not particularly restrictive*, because most operations are separable by this technique, which is much more general than separability as defined for linear convolutions, Fourier Transforms, etc.
- These row and column operations are put in recursive form – that is, in the form needed for a single-pass software implementation or a pipelined hardware implementation.
- The resulting operations are realized as finite-state machines (FSMs).
- These FSMs can be implemented either in software or in pipelined hardware.

Overall scheme: The pixels of the input image are processed in raster-scan order. It makes no difference if the scan is row-by-row or column-by-column, top-to-bottom or bottom-to-top, left-to-right or right to left. (*Interlaced* scans are not allowed, because the concept of a pixel's *neighbours* is destroyed in interlaced scans.) The usual case, in which the scanning is along the rows, is shown in ➲ *Fig. 20.1*. In whatever order the pixels are scanned, each input-image pixel is accessed *once and only once*. All needed neighbourhood information is encoded (often in extremely compact form) in the state buffers of the two FSMs.



■ **Fig. 20.1**
SKIPSM sequence diagram

The inner-loop steps, applicable to all implementations, are executed *once* for each pixel in the input image. The steps are shown in ➲ *Fig. 20.1*, and are as follows:

1. Fetch **Input**, the next input pixel value (in raster-scan order) from **Input_Image**.
2. Fetch **Row_State** (the contents of the **Row-State Buffer**).
3. *Row machine computation*: Using the **Input** value and the **Row_State** value, carry out the row-machine computations required for the particular image-processing operation being implemented. These computations can be carried out either with a lookup table or with a sequence of computational steps in C code. *By definition*, this computation produces two values:
   (a) The value of the **Next_Row_State**. This is written to the **Row-State Buffer**, overwriting the previous value, which will never be needed again. This is called *updating the row state.*
   (b) The **Row_Machine_Output**, which is used immediately by the column machine. See Step 5 below.
4. Fetch the **Column_State** value from the register in the **Column-State Buffer** corresponding to the current image column.

5. *Column machine computation*: Using the **Row_Machine_Output** (step 3b) and the **Column_State** value (step 4), carry out the column-machine computations required for the particular image-processing operation being implemented. (As in step 3, these computations can be carried out either with a lookup table or with a sequence of computational steps in C code.) *By definition*, this produces two values:
   (a) The value of the **Next_Column_State**. This is written to the same place in the **Column-State Buffer** from which the column state was obtained in step 4, thus overwriting the old value, which will never be needed again. This is called *updating the column state.*
   (b) The overall **Output** (i.e., column-machine output). This is written immediately to the appropriate location in **Output_Image**, corresponding to the *centre* of the neighbourhood. For any neighbourhood larger than 1 × 1, this location is *always* "behind" the input pixel in the raster-scan sequence. *Therefore, the input image buffer may also be used for the output image*, if desired. No information will ever be lost if this is done, because *all the needed neighbourhood information is encoded in the states of the two FSMs.*

Now, repeat these steps for the next pixel in the raster scan. And so on.

Steps 3 and 5 can be done with special-purpose hardware, or with pre-computed lookup tables, or with compiled step-by-step run-time code. For some operations, the lookup-table approach can be much faster that conventional implementations of the same operations on the same computer. For software implementations, only the code in steps 3 and 5 (or the values in the lookup tables) change when a different operation is implemented.

In the lookup-table implementation, the lookup tables are created in advance using the same inner-loop code (steps 3 and 5), applied to all possible inputs and all resulting FSM states. At inspection-system boot-up, *all* the lookup tables needed for *all* the operations involved in an application are usually preloaded from disk or ROM or RAM. Stepping through a sequence of operations then consists of selecting a sequence of pointers, pointing to appropriate locations for each lookup-table in turn. This kind of implementation is most advantageous in production situations, when the same inspection algorithm is used over and over. The lookup tables themselves are usually small (a few KB) but can be large for some operations, such as applying gigantic (e.g., 25 × 25) morphology structuring elements in one pass. Large lookup tables, when required, would appear to be a factor working against the lookup-table implementation, but this is also precisely the situation in which enormous speedups are achieved. In such cases, large lookup tables are often justified. (Using lookup tables, it takes approximately the same time to execute a 25 × 25 operation as to execute a 3 × 3 operation. Execution speed is essentially independent of neighbourhood size.)

20.3 Edge Effects

Whenever a neighbourhood image-processing operation is used, the values of the output image are, strictly speaking, not defined unless the computational neighbourhood is completely *inside* the boundaries of the input image. Therefore, pixel values for some rows and columns next to the four edges of the output image can not be computed according to the same rule used for interior pixels. A decision must be made about these pixels. One approach to this shrinkage is to specify an output image which is *smaller* than the input image, and ignore the "undefined" pixels. For example, suppose that the processing

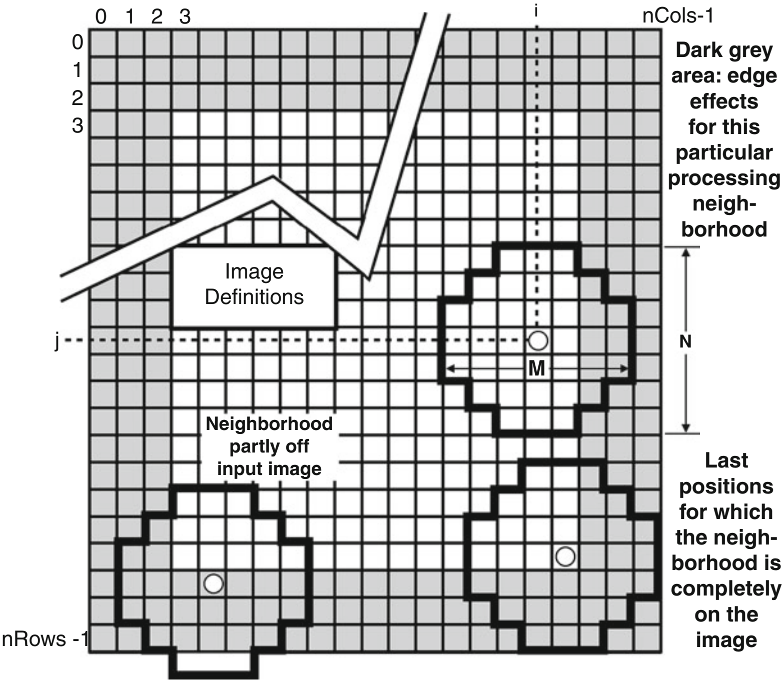

0 1 2 3
i
nCols-1
0
1
2
3
Dark grey area: edge effects for this particular processing neigh-borhood
Image Definitions
j
N
M
Neighborhood partly off input image
Last positions for which the neigh-borhood is completely on the image
nRows -1

Fig. 20.2
Image definitions and edge effects

neighbourhood has M columns and N rows, and that the computed output is to be placed at the pixel corresponding to the centre of this neighbourhood. (See *Fig. 20.2*.) Then, if M is an odd number, M2 = INTEGER(M/2) columns will be lost at the left edge of the output image, and M2 columns will be lost at the right edge of the output image. Likewise, if N is an odd number, N2 = INTEGER(N/2) rows will be lost at the top edge of the output image, and N2 rows will be lost at the bottom edge of the output image. For example, if M = 19 and N = 15, then nine columns will be lost at the left edge of the output image, and nine columns will be lost at the right edge of the output image. Likewise, seven rows will be lost at the top edge of the output image, and seven rows will be lost at the bottom edge of the output image.

Because many of the algorithms used in machine vision involve the comparison and/or combination (addition, subtraction, OR, XOR, AND, etc.) of images at various stages of the computation, this image shrinkage can cause difficulties. Therefore, it is common practice in machine vision to "pad out" the output image to the same size as the input image by filling in these border rows and columns in some defined way.

A full discussion of edge effects, which are present in *all* neighbourhood operations, regardless of computational method, is beyond the scope of this short article. However, some decision must be made in every case, either to omit the edge pixels or to fill them in according to some rule(s).

One simple approach is to set these output pixels to a fixed value, such as zero. Another approach is to attempt to provide smoother edge transitions and allow the border pixels to contain *some* information, by "phasing in" the computations at the top and left edges as the neighbourhood operator moves onto the input image, and "phasing out" the computations at the right and bottom edges as the operator moves beyond the input image. In effect, one computes the output based on only the part of the processing neighbourhood that corresponds to input image pixels. This makes eminent sense for certain operations, such as MAX or MIN or MAX-MIN. For others, special provisions should be made: e.g., for averaging, one should adjust the normalizing constant to reflect the number of image pixels actually being averaged. These details will not be pursued further here, and the inner-loop C-code given in the examples below does not include steps to compute the values of the output image edge pixels.

20.4 Automated Generation of Lookup Tables

The lookup-table approach would be of little practical interest unless the LUTs could be created easily and routinely. Fortunately, while the *results* of these operations can be very complex, the principles behind the automated creation of SKIPSM lookup tables are very straightforward: Calculate the results of applying the specified neighbourhood image processing operation to all possible *combinations* of input values (i.e., input image pixel values) in the defined neighbourhood, sort them into numerical order, and make a table of the results. This might seem impossibly difficult, but for some simple operations it can be done "in your head" and for others the procedure can easily be automated. The earliest automated procedure was implemented for binary morphology on a 1995-era Macintosh desktop computer using a Microsoft Excel® spreadsheet. Cell formulas were used to define the desired neighbourhood image processing operation. A macro took the result of the image processing for each sequentially-applied input image value and the current state value and determined the resulting new state and the output. It then appended these newly-calculated states to the list of states, sorted the states into numerical order, and deleted duplicate states. This continued until no new states were generated. (This process ALWAYS terminates because these are finite-state machines. In practice, even "analog" values are represented by a finite number of discrete levels, typically 256, so that finite-state machines can be used for grey-scale operations also.) Finally, the macros sorted all of these results, renamed the states with consecutive integer values, and created the final lookup table. Small tables (100 or fewer states) were created in a second or two. For the most complex binary-morphology situation we attempted, it took a few minutes to grind out a pair of state tables with more than 10,000 states. This approach is shown in some detail in [2, 3, 8].

Later, the same procedures for binary morphology were implemented in C-code as four programs, which were faster than the spreadsheet implementation but otherwise gave identical results:

CREATE_ROW_FSM The user specifies a binary row pattern, e.g., [0 0 1 1 1 0 0]. The output of this program is the corresponding row LUT. This process is repeated for all of the SE row patterns.

COMBINE_ROW_FSMs The user specifies the file names of a *pair* of row LUTs, and the program produces a combined row LUT. Using the same program, this LUT is combined in turn with *another* of the row LUTs, until all have been combined into a single combined row LUT.

CREATE_COLUMN_FSM The user specifies the row-pattern identifiers for the SE. For example, in ➲ *Fig. 20.4*, the nine distinct row-pattern identifiers are, from top to bottom [1 2 3 3 3 3 3 2 1]. The output of this program is a column LUT. If only one SE is being applied, this is the last step.

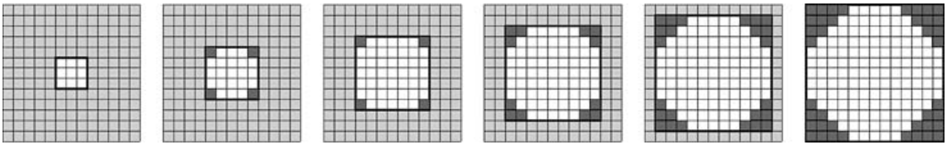


Fig. 20.3
SEs for six simultaneous erosions, performed very rapidly in one pass using SKIPSM LUTs

There is one additional program, **COL_COMBINER**, which is used to create combined column LUTs. A combined column LUT, along with the combined row LUT used to generate it, carries out more than one neighbourhood image-processing operation simultaneously. This allows the execution of *multiple SEs* in one pass. For example, six different concentric (nested) circular SEs have been combined into a single operator to provide *six stages* of Grassfire transform erosion in a single pass. (See *Fig. 20.3*)

There is, in principle, no limit to how many different SEs can be applied simultaneously in a single pass, although attempts to combine very large and/or very complex SEs may result in unmanageably-large LUTs. There is also the restriction that all the SEs being combined into a single operation must share the same combined row LUT, which means that they must have the same neighbourhood size. This is easily overcome by "padding" the smaller SEs with "don't care" pixels to fill them out to the size of the largest one, as was done in *Fig. 20.3*. The added pixels are shown in a lighter grey. Needless to say, when a small SE is "padded," it loses the same number of edge rows and columns as the largest SE in the set, and therefore can not "reach as far into the corners" of the image as if it had been applied by itself.

20.5 Binary Erosion with Very Large Rectangular Structuring Elements (SEs)

Binary morphology has been used widely in inspection algorithms, optimal-packing problems, etc. When large SEs (e.g., 25 × 25, 51 × 51) are required, the usual approach is to carry out many erosion operations in sequence using small (typically 3 × 3) SEs, a time-consuming process. Using SKIPSM, such large-SE erosions can be done very rapidly in one image pass. Binary dilations can be implemented just as easily.

As a first example, consider large square or rectangular binary-erosion SEs with N rows and M columns. These simple SEs are not very interesting in themselves. Furthermore, they are very easy to implement using other techniques. However, they will serve to illustrate the concept of finite-state machines, separation, and recursion.

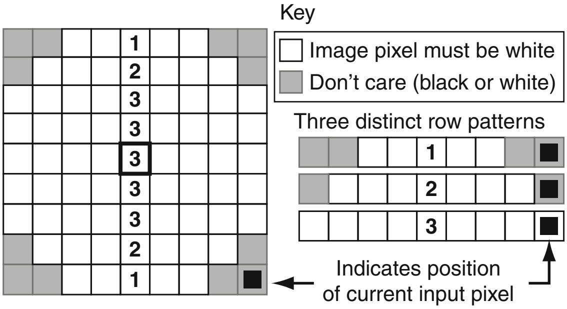
Here is the complete inner-loop code for a direct implementation (i.e., using compiled C-code) for this example. As always, each input-image pixel is fetched *once*. There are *no other inner-loop steps*, no matter how large the SE happens to be. Thus, the speed is essentially independent of N and M. Note that a *threshold* step is included to allow this code to operate on either binary images (values 0 and 1) or grey-scale images. That is, the thresholding operation is built into the code so that a separate thresholding operation is not needed. Edge effects are ignored here.

```
// The SKIPSM row machine for a rectangular binary-erosion SE with M columns
   if (Input_Image[j][i] < Threshold) Row_State = 0;
       // Input pixel is black. Set Row_State to 0.
   else Row_State = Minimum(Row_State + 1, M);
       // Input pixel is white. Increment the Row_State, clamping its value at M.
   if (Row_State < M) Row_Machine_Output = 0;
       // There is at least one black pixel in the 1xM neighborhood. Set
       Row_Machine_Output to 0.
   else Row_Machine_Output = 1;
       // The current 1xM row neighborhood has M white pixels. Set
       Row_Machine_Output to 1.
// The SKIPSM column machine for a rectangular binary-erosion SE with N rows
   Column_State[i] = Minimum(Column_State[i] + Row_Machine_Output, N);
       // Increment the column state, clamping its value at N.
   if (Column_State[i] < N) Output = BlackValue;
       // The SE does not ``fit'' here. Set Column_State to zero.
   else Output = WhiteValue;
       // The SE ``fits'' at the current location and the final output is white.
```

There are much more efficient ways to implement these steps, including a very fast method using only left-shifts and masking. However, the goal here is to illustrate the SKIPSM concepts, so a straightforward C-code implementation is shown because it is easier to understand.

20.6 Lookup-Table Implementation of Binary Erosion with a 9 × 9 Circular SE

Next, consider the more difficult and much more interesting task of binary erosion with large and complex SEs. As an example, we will use the 9 × 9 circular SE shown in Fig. 20.4, although much larger SEs can be implemented with almost equal ease. The conventional approach to this particular operation is to make four successive passes through the image



Key
□ Image pixel must be white
▨ Don't care (black or white)
Three distinct row patterns
1
2
3
3
3
3
3
2
1
Indicates position of current input pixel

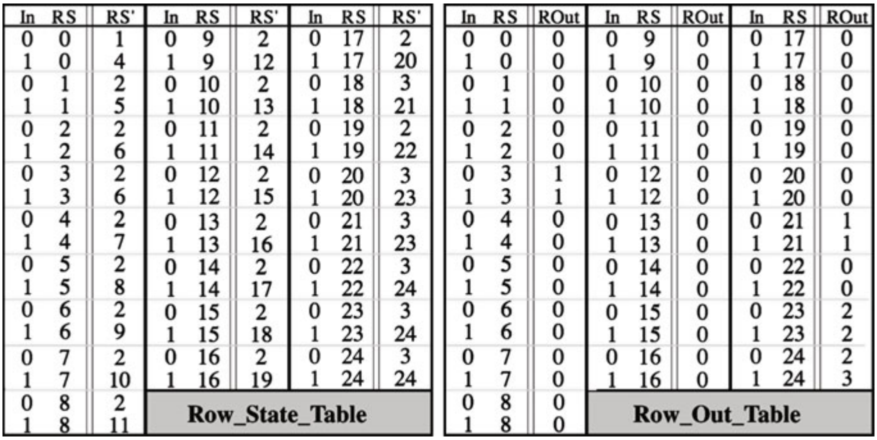
Fig. 20.4
9 × 9 circular structuring element

with 3 × 3 SEs. (Note that SEs larger than 9 × 9 would require even more 3 × 3 passes, and that SEs with holes or concavities, which are difficult or impossible to achieve using successive small SEs, can also be done easily in one pass using SKIPSM.) Instead, we will use a lookup-table implementation to execute this operation in one pass.

Because the column machine is required to keep track of all the partially-completed patterns as it moves down the columns of the image, it always needs to know which (if any) of the distinct row patterns the *current* image row satisfies. (The column machine incorporates all needed information about *previous* rows into its own column state.) Thus, the overall row-and-column operator must somehow make *simultaneous* comparisons of the current image row with *all three* of the different row patterns, shown in the figure as distinct row patterns 1, 2, and 3. Since the column machine's *only* source of this information is the row machine output, this implies that the row machine must detect and "encode" all three row patterns simultaneously into a single number.

It turns out that only 25 states are sufficient to encode all the necessary row information for the *simultaneous* testing of all three distinct row patterns. The combined row machine output is (and must be) in the form of a *single number.* In this case, it takes on the values **0**, **1**, **2**, or **3**. An output of **3** implies that all three row patterns are "satisfied." **2** implies that only row patterns **1** and **2** are "satisfied." **1** implies that only row pattern **1** is "satisfied." **0** implies that none of the row patterns is "satisfied." (These particular row patterns can be said to be "well ordered," in that they can easily be placed in increasing order: all of the white pixels in **1** are present in **2** and all of the white pixels in **2** are present in **3**. Cases where the rows are not well-ordered, such as 0 1 1 0 1 1 0 and 1 1 0 0 0 1 1, require additional row-machine output values to encode the various combinations, but are easily handled. The details are omitted here.)

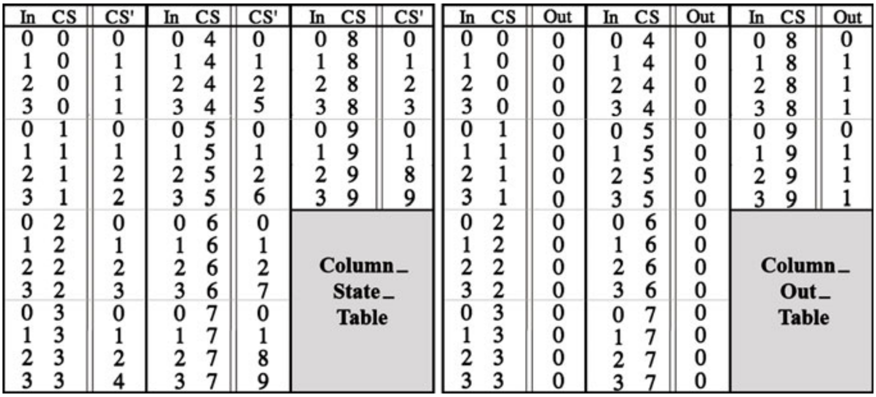
For the binary structuring element shown in ➤ *Fig. 20.4*, ➤ *Fig. 20.5* gives the *updated* **Row_State** and the **Row_Machine_Output** as functions of the input pixel **Input** (value 0 or 1) and the current row state **Row_State**, fetched from the row-state register.



In	RS	RS'	In	RS	RS'	In	RS	RS'
0	0	1	0	9	2	0	17	2
1	0	4	1	9	12	1	17	20
0	1	2	0	10	2	0	18	3
1	1	5	1	10	13	1	18	21
0	2	2	0	11	2	0	19	2
1	2	6	1	11	14	1	19	22
0	3	2	0	12	2	0	20	3
1	3	6	1	12	15	1	20	23
0	4	2	0	13	2	0	21	3
1	4	7	1	13	16	1	21	23
0	5	2	0	14	2	0	22	3
1	5	8	1	14	17	1	22	24
0	6	2	0	15	2	0	23	3
1	6	9	1	15	18	1	23	24
0	7	2	0	16	2	0	24	3
1	7	10	1	16	19	1	24	24
0	8	2	**Row_State_Table**					
1	8	11						

In	RS	ROut	In	RS	ROut	In	RS	ROut
0	0	0	0	9	0	0	17	0
1	0	0	1	9	0	1	17	0
0	1	0	0	10	0	0	18	0
1	1	0	1	10	0	1	18	0
0	2	0	0	11	0	0	19	0
1	2	0	1	11	0	1	19	0
0	3	1	0	12	0	0	20	0
1	3	1	1	12	0	1	20	0
0	4	0	0	13	0	0	21	1
1	4	0	1	13	0	1	21	1
0	5	0	0	14	0	0	22	0
1	5	0	1	14	0	1	22	0
0	6	0	0	15	0	0	23	2
1	6	0	1	15	0	1	23	2
0	7	0	0	16	0	0	24	2
1	7	0	1	16	0	1	24	3
0	8	0	**Row_Out_Table**					
1	8	0						

Fig. 20.5
Row_State_Table and Row_Out_Table, the row machine lookup tables for 9 × 9 circular SE



In	CS	CS'	In	CS	CS'	In	CS	CS'
0	0	0	0	4	0	0	8	0
1	0	1	1	4	1	1	8	1
2	0	1	2	4	2	2	8	2
3	0	1	3	4	5	3	8	3
0	1	0	0	5	0	0	9	0
1	1	1	1	5	1	1	9	1
2	1	2	2	5	2	2	9	8
3	1	2	3	5	6	3	9	9
0	2	0	0	6	0	**Column_ State_ Table**		
1	2	1	1	6	1			
2	2	2	2	6	2			
3	2	3	3	6	7			
0	3	0	0	7	0			
1	3	1	1	7	1			
2	3	2	2	7	8			
3	3	4	3	7	9			

In	CS	Out	In	CS	Out	In	CS	Out
0	0	0	0	4	0	0	8	0
1	0	0	1	4	0	1	8	1
2	0	0	2	4	0	2	8	1
3	0	0	3	4	0	3	8	1
0	1	0	0	5	0	0	9	0
1	1	0	1	5	0	1	9	1
2	1	0	2	5	0	2	9	1
3	1	0	3	5	0	3	9	1
0	2	0	0	6	0	**Column_ Out_ Table**		
1	2	0	1	6	0			
2	2	0	2	6	0			
3	2	0	3	6	0			
0	3	0	0	7	0			
1	3	0	1	7	0			
2	3	0	2	7	0			
3	3	0	3	7	0			

Fig. 20.6
Column_State_Table and Column_Out_Table, the column-machine lookup tables for 9 × 9 circular SE

For the binary structuring element shown in Fig. 20.4, Fig. 20.6 give lookup tables for the *updated* **Column_State** and the **Output** as functions of **Row_Machine_Output** and the current **Column_State**, fetched from the column-state buffer.

The inner-loop code for this implementation is shown below. As always, the two outer loops (not shown) step through the image, one pixel at a time, in raster-scan order. *Each input-image pixel is fetched once. There are no other inner-loop steps.* Note that all lookup-table implementations of SKIPSM operators will have this general form, with the exception of the first two steps, which depend on the operator and the type of image being processed. Although there are more efficient ways to implement these steps using single-index lookup tables, an approach using dual-index tables is presented here because it is the easiest to understand. Again, edge effects are ignored here.

```
// Row machine.
  if (Input_Image[j][i] < Threshold) In = 0;
      // If the input is a grey-scale image, this step converts it to binary so
      that binary erosion
      // can be performed. Thus, this code works with either kind of image.
  else In = 1;
      // Input pixel is ``white.'' Set In to 1.
  Row_Machine_Output = Row_Out_Table[In][Row_State];
      // Fetch the row machine output from the row-machine LUT.
  Row_State = Row_State_Table[In][Row_State];
      // Fetch the next row state from the LUT and update the row-state register.
// Column machine Uses Row_Machine_Output as the column-machine input.
   Output_Image[j-M2] [i-N2] = Column_Out_Table[Row_Machine_Output]
   [Column_State[i]];
       // Fetch result from LUT, write output.
   Column_State[i] = Column_State_Table[Row_Machine_Output]
   [Column_State[i]];
       // Fetch next column state and update column-state buffer.
```

20.7 Gaussian Blurs

Many different low-pass filtering operations are used in image processing – low pass in the sense of passing low spatial frequencies and rejecting high spatial frequencies. These are usually given names suggesting smoothing, averaging, or blurring. One of the most widely-used operations of this type is the so-called Gaussian blur, which has the advantages of being very "smooth" and also circularly symmetric, so that edges and lines in various directions are treated similarly. For machine vision applications, blurring operators are often defined on small neighbourhoods (e.g., 3 × 3, 11 × 11), and for a small finite number of grey levels (e.g., 256). In such cases, the ideal Gaussian "bell curve" must be approximated by a rather small number of integer values. The values are usually based on Pascal's triangle (the binomial coefficients). These approach the true Gaussian curve more and more closely as the number of points increases. However, the execution times of these operations can be rather long, especially where large kernels are involved.

Proper use of two properties of Gaussian blurs can help to reduce these long execution times:

1. Large kernels can be decomposed into the sequential application of small kernels. However, this can result in the accumulation of unacceptable rounding errors.
2. Gaussian blurs are separable into independent row and column operations. The row and column operators can be interchanged, with identical results, except for differences in rounding errors.

   The SKIPSM implementation makes use of both of these characteristics and adds a third one:
3. The row and column operations can be formulated as finite-state machines (FSMs) to produce highly efficient code. Furthermore, for multi-step sequential decompositions, *the necessity of writing results to intermediate buffers and then fetching these results for the next operation is eliminated.* This not only saves time, but eliminates the accumulation of rounding errors and often provides a significantly better approximation to the desired Gaussian "bell" curve.

The 3 × 3 *Gaussian blur*: The SKIPSM 3 × 3 implementation requires two temporary variables, two memory locations for the state of the row machine, and two column state buffers. The column state buffers are set to zero at the start of the overall operation, and the row state registers are set to zero at the start of each row. The inner loop code is given below. The output is written to address [j−1][i−1] because this is the centre of the current 3 × 3 neighbourhood.

Inner loop code for 3 × 3 Gaussian blur with SE rows [1 2 1] [2 4 2] [1 2 1]

```
    Temp1 = Input_Image[j][i]                      // Step 1: input
    Temp2 = Row_State0 + Temp1                     // Step 2: add
    Row_State0 = Temp1                             // Step 3: update
    Temp1 = Row_State1 + Temp2                     // Step 4: add
    Row_State1 = Temp2                             // Step 5: update
    Temp2 = Column_State0[j] + Temp1               // Step 6: add
    Column_State0[j] = Temp1                       // Step 7: update
    Temp1 = Column_State1[j] + Temp2               // Step 8: add
    Output_Image[j-1][i-1] = (Temp1 + 8)/16       // Step 9: output
    Column_State1[j] = Temp1                      // Step 10: update
```

This 3 × 3 SKIPSM implementation requires ten steps. The brute-force approach for this case requires 24 steps (9 dual-index pixel fetches, 5 multiplies, 8 additions, 1 scaling step, and a dual-index "write" to the output image).

The 5 × 5 *Gaussian blur*: [1 4 6 4 1] [4 16 24 16 4] [6 24 36 24 6] [4 16 24 16 4] [1 4 6 4 1]

The SKIPSM 5 × 5 implementation requires two temporary variables, four memory locations or registers for the state of the row machine, Row_State0 through Row_State3, and four column state buffers. The column state buffers are set to zero at the start of the overall operation, and the row state buffers are set to zero at the start of each row. The inner loop code is given below. The output is written to address [j−2][i−2] because this is the centre of the current 5 × 5 neighbourhood.

Inner-loop code for 5 × 5 Gaussian blur

```
// Row machine
    Temp1 = Input_Image[j][i]                      // Step 1: input
    Temp2 = Row_State0 + Temp1                     // Step 2: add
    Row_State0 = Temp1                             // Step 3: update
    Temp1 = Row_State1 + Temp2                     // Step 4: add
    Row_State1 = Temp2                             // Step 5: update
    Temp2 = Row_State2 + Temp1                     // Step 6: add
    Row_State2 = Temp1                             // Step 7: update
    Temp1 = Row_State3 + Temp2                     // Step 8: add
    Row_State3 = Temp2                             // Step 9: update

// Column machine
    Temp2 = Column_State0[i] + Temp1               // Step 10: add
    Column_State0[i] = Temp1                       // Step 11: update
    Temp1 = Column_State1[i] + Temp2               // Step 12: add
    Column_State1[i] = Temp2                       // Step 13: update
    Temp2 = Column_State2[i] + Temp1               // Step 14: add
    Column_State2[i] = Temp1                       // Step 15: update
    Temp1 = (Column_State3[i] + Temp2 + 128)/256 // Step 16: add and normalize
    Column_State3[i] = Temp2                       // Step 17: update
    Output_Image[j-2][i-2] = Temp1                 // Step 18: output
```

This 5 × 5 SKIPSM Gaussian blur implementation requires 18 steps. In comparison, the brute-force approach for this case requires 72 steps (25 dual-index pixel fetches, 21 multiplications, 24 additions, 1 scaling step, and 1 "write" to the output image). The numbers in the block diagram (➲ *Fig. 20.7*) correspond to step numbers in the above code.

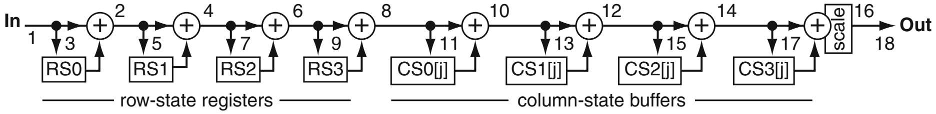


Fig. 20.7
Block diagram for 5 × 5 Gaussian blur in one pass

Difference-of-Gaussian (DoG) filters: The SKIPSM paradigm can be used to apply Difference-of-Gaussian (DoG) band-pass filters in one pass. This involves carrying out two simultaneous concentric Gaussian blurs of different sizes, and subtracting one result from the other. DoG filters can be used to enhance objects or features selectively, according to their size. Thus, they operates in the spatial domain to produce results similar to the effect of a Fourier filter operating in the spatial-frequency domain, but without the excessive time penalty incurred when *two* so-called Fast Fourier Transforms must be applied. However, even though DoG filters are faster than Fourier transforms when only a few spatial-frequency bands need to be detected or enhanced, they are still fairly slow for large diameters. SKIPSM provides a meaningful decrease in DoG filter execution time, compared to conventional implementations of DoG filters. See [13].

20.8 The 4M Filter, a Pseudo-Median 3 × 3 Filter

This example involves a useful filtering operation which has not been widely discussed in the literature. The operation is similar to a median filter, but is faster because the sorting steps required by ranked filters, which are generally very slow, are replaced by a few comparisons.

Definition of the 4M filter: See ➋ *Fig. 20.8*. Two operations are performed simultaneously on the current 3 × 3 neighbourhood, and the results are averaged. The two operations are

1. **min-max** (minimum of the row maximums): Select the *maximum* of each of the three rows in the current 3 × 3 neighbourhood. From these three row values, select the *minimum* value. This is the output of the **min-max** operation.
2. **max-min** (maximum of the row minimums): Select the *minimum* of each of the three rows in the same 3 × 3 neighbourhood. From these three row values, select the *maximum* value. This is the output of the **max-min** operation.

Requirements for storage of state information:

Three registers for row state: Row_State1, Row_State2, Row_State3
Five temporary registers: Temp, Row_Machine_Output1, Row_Machine_Output2, COut1, COut2
Four column state buffers: Column_State1[j], Column_State2[j], Column_State3[j], Column_State4[j], j = 1,. . .,N

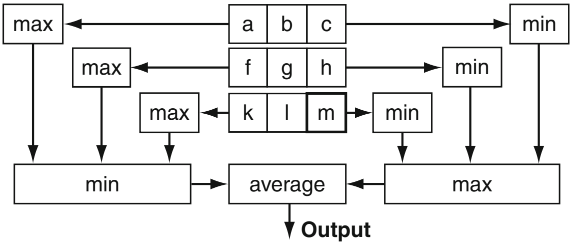


Fig. 20.8
Functional diagram of the 4M filter

Inner loop code for the 4M filter

This code is executed *once* for each input image pixel. There are no other main-loop steps.

```
    Temp = Input_Image[j][i];
    Row_Machine_Output1 = Maximum(Temp,Row_State2);
    Row_Machine_Output2 = Minimum(Temp,Row_State3);
    Row_State2 = Maximum(Temp,Row_State1);
    Row_State3 = Minimum(Temp,Row_State1);
    Row_State1 = Temp;
    COut1 = Minimum(Row_Machine_Output1,Column_State2[j]);
    COut2 = Maximum(Row_Machine_Output2,Column_State4[j]);
    Output[j-1][i-1] = (COut1+COut2)/2;
    Column_State2[j] = Minimum(Row_Machine_Output1,Column_State1[j]);
    Column_State4[j] = Maximum(Row_Machine_Output2,Column_State3[j]);
    Column_State1[j] = Row_Machine_Output1;
    Column_State3[j] = Row_Machine_Output2;
```

The *Minimum* and *Maximum* operators used here involve the comparison of two grey-scale values, and can be implemented in various ways. The individual **min-max** and **max-min** operators, when implemented separately in SKIPSM, are relatively fast. There is a further saving of execution time when they are implemented simultaneously because certain steps can be shared. If the processor being used has an adequate cache, most of the intermediate results will remain in cache, further speeding up the operation.

20.9 "Separable Median Filter"

The so-called "separable median filter" [16, 21, 22] does not produce the true median (rank 5) of the 9 pixels in its 3 × 3 neighbourhood. In spite of this, it can be very useful because it *approximates* the median and is relatively fast compared to a true median filter.

From a philosophical or theoretical standpoint, there is seldom solid justification for insisting on one particular image-processing operation in preference to other similar operations. A typical use of median or other ranked filters is for "cleaning up" images subject to "dropouts," "snow," "salt and pepper" noise, etc. For these applications, it is difficult in most cases to find a convincing argument that the separable median filter (rank 4, 5, or 6) described here is significantly inferior in any meaningful way to the true median filter (rank 5) in most cases. A flexible prototyping system for the development of machine vision algorithms, as advocated and described elsewhere in this handbook, can be used to verify *experimentally* whether a true median filter (rank 5) is required in any particular application, in preference to this "rank 4-5-6" filter. When speed of operation is important, the separable median filter, (which calculates the median of the row medians) is a useful option in the prototyping system, because it has only 11 program steps in the inner loop, making it faster than the rank-5 filter, which has more steps.

➔ *Figure 20.9* shows a functional diagram for this filter. The rectangular boxes in this diagram are three-input sorters, the middle outputs of which are the medians of the three inputs. In practice, three-input sorters are implemented using three two-input sorters. Each two-input sorter uses one if-then-else statement, followed by either the interchanging of

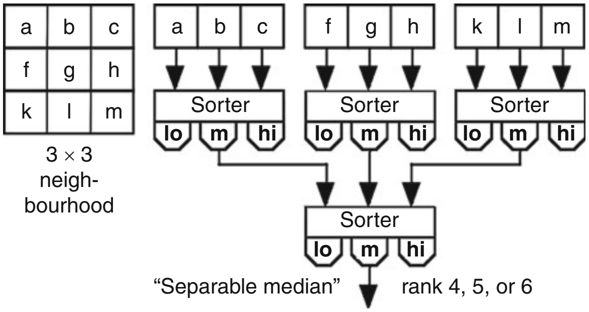


Fig. 20.9
Separable median functional diagram

the two input variables, or leaving them unchanged. This is the operation used in a "bubble" sorter. The three-input sorters used here can be thought of as three-stage bubble sorters in which only the middle output is used, and the other two outputs are ignored.

Figure 20.10 shows the same algorithm in pipelined form: Each pixel is fetched only once, and all information necessary to compute the output is incorporated into the "states" of two finite-state machines (FSMs). Two registers, Row_State0 and Row_State1, are needed to store the state of the row machine. Two column state buffers, Column_State0[i] and Column_State1 [i], each with a number of addresses equal to the number of pixels in an image row, are needed to "remember" the relevant information from the two previous rows. It is sufficient that these registers and buffers have the same word length as the input image, although longer word lengths may be used if more convenient for programming. In the figure, **B** = median (**a**, **b**, **c**), **G** = median (**f**, **g**, **h**), and **L** = median(**k**, **l**, **m**).

The corresponding inner loop code is given below. The numbers in the small grey boxes in the figure refer to the step numbers in the code comments. Because of the 1-pixel offset, the output may be written back into the input image buffer, if desired.

```
// Row machine for ``separable median'' filter
  if (Row_State1 > Row_State0) { Min = Row_State0; Med = Row_State1; }
      // Step 1a: Compare k with l
  else { Min = Row_State1; Med = Row_State0; }
      // Step 1b: Med=max(k,l). Save Min=min(k,l) for step 5
  Row_State1 = Row_State0;
      // Step 2: Update row state register Row_State1; it now contains l
  Row_State0 = Input_Image[j][i];
      // Step 3: Load next pixel into Row_State0; it now contains m
  if (Row_State0 < Med) Med = Row_State0;
      // Step 4: Med = min(m, max(k, l))
  if (Min > Med) Med = Min;
      // Step 5: Med = med(k, l, m). Save for step 8
// Column machine for ``separable median'' filter
   if (Column_State1[i] > Column_State0[i]) { Min = Column_State0[i]; Out =
   Column_State1[i]; }
      // Step 6a: Compare previous medians B and G
```

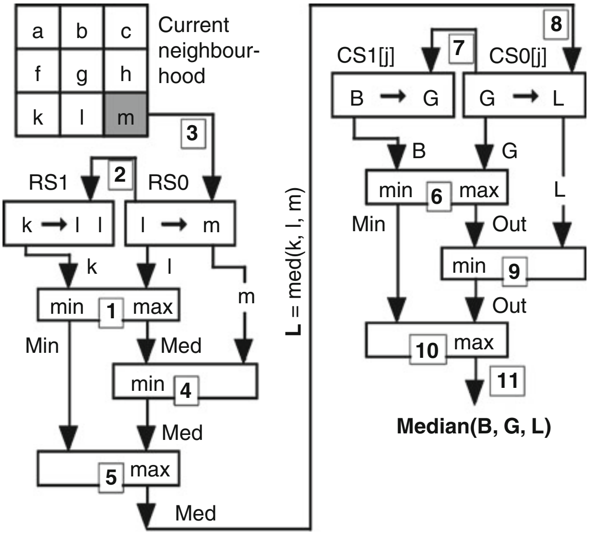


Fig. 20.10
Separable median sequence diagram

```
else { Min = Column_State1[i]; Out = Column_State0[i]; }
    // Step 6b: Result: Out = max(B, G); Min = min(B, G)
Column_State1[i] = Column_State0[i];
    // Step 7: Column state register Column_State1[i] now contains G
Column_State0[i] = Med;
    // Step 8: Column state register Column_State0[i] now contains L
if (Column_State0[i] < Out) Out = Column_State0[i];
    // Step 9: Out = min(L, max(B, G))
if (Min > Out) Out = Min;
    // Step 10: Final output Out = med(B, G, L)
Output_Image[j-1][i-1] = Out
    // Step 11: Write output, with offset to compensate for latency
```

Notations such as **B** → **G** in the register boxes in Fig. 20.10 indicate the register contents *before* inner-loop processing starts (e.g., **B**) and *after* it is completed (e.g., **G**). Note that, as always, once state buffer initializations and edge-effect computations have been completed, these 11 steps are executed once for each pixel in the image. No other steps are required. Note also that the temporary variables **Min**, **Med**, and **Out** are used and reused in such a way as to eliminate the "else" clause from four of the six "if" statements (steps 4, 5, 9, and 10).

Comparison of the SKIPSM "separable median" algorithm with algorithms intended for similar uses but implemented in other ways (See [16] for an extended discussion of this topic): The well-known "bubble-sort" algorithm for a *full ranked filter* (nine ranked outputs) requires 36 conditional-interchange operations, plus all the usual image reads, image writes, etc. A "bubble-sort" algorithm for the *true median* based on the ranked filter but omitting calculation of the other eight ranks requires 30 conditional-interchange operations. Mahmoodi's improved version for the *true median* (private correspondence) reduces this to 19 conditional-interchange operations. A rank 4-5-6 filter based on Mahmoodi's algorithm, which is the starting point for the SKIPSM "separable median" filter (see ➲ *Fig. 20.9*), has 12 conditional-interchange operations, three for each three-input sorter. By saving the previously-computed row medians as part of the column state, the SKIPSM implementation reduces this to only *six* conditional-interchange operations. Furthermore, it eliminates most of the input-image pixel fetches and many of the other intermediate steps used in "brute-force" and "bubble-sort" implementations, resulting in considerably faster execution times.

20.10 Grey-Scale Erosion with a 7 × 7 Diamond

➲ *Figure 20.11* shows the structuring element and sequence diagram for dilation of a grey-scale image with a 7 × 7 diamond-shaped structuring element. The corresponding inner-loop code is given below. The boxes labelled "MN" calculate the minimum of the two inputs.

```
// Row machine for grey-scale erosion with a 7 × 7 diamond:
    P = Input_Image[j][i];                              // Step 1
    R1 = Row_State6;                                    // Step 2
    R2 = Row_State3;                                    // Step 3
    R3 = Row_State1;                                    // Step 4
    R4 = Min(Row_State0,P);                             // Step 5
    Row_State0 = Min(Row_State1,P);                     // Step 6
    Row_State1 = Min(Row_State2,P);                     // Step 7
    Row_State2 = Min[Row_State4,P);                     // Step 8
    Row_State3 = Row_State4;                            // Step 9
    Row_State4 = Min(Row_State5,P);                     // Step 10
    Row_State5 = Min(Row_State8,P);                     // Step 11
    Row_State6 = Row_State7;                            // Step 12
    Row_State7 = Row_State8;                            // Step 13
    Row_State8 = P;                                     // Step 14

// Column machine for grey-scale erosion with a 7 × 7 diamond:
    Output_Image[j-3][i-3] = Min(Column_State0[j],R1); // Step 15
    Column_State0[j] = Min(Column_State1[j],R2);        // Step 16
    Column_State1[j] = Min(Column_State2[j],R3);        // Step 17
    Column_State2[j] = Min(Column_State3[j],R4);        // Step 18
    Column_State3[j] = Min(Column_State4[j],R3);        // Step 19
    Column_State4[j] = Min(Column_State5[j],R2);        // Step 20
    Column_State5[j] = R1;                              // Step 21
```

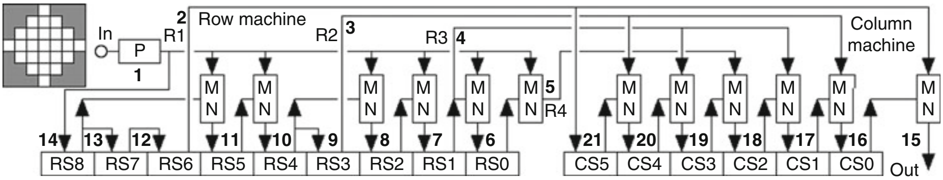


Fig. 20.11
Grey-scale erosion with 7 × 7 diamond

20.11 Grey-Scale Dilation with a 7 × 7 Circle

Figure 20.12 shows the structuring element and sequence diagram for dilation of a grey-scale image with a 7 × 7 circular structuring element. The corresponding inner-loop code is given below. The boxes labelled "MX" in the figure calculate the maximum of the 2 inputs, and 11 such operations are required. These are indicated by "Max" in the code below.

```
// Row machine for grey-scale dilation with a 7 × 7 circle:
    P = Input_Image[j][i];                                    // Step 1
    R1 = Row_State3;                                          // Step 2
    R2 = Row_State2;                                          // Step 3
    R3 = Max(Row_State0,P);                                   // Step 4
    Row_State0 = Max(Row_State1,P);                           // Step 5
    Row_State1 = Max(Row_State2,P);                           // Step 6
    Row_State2 = Max(Row_State4,P);                           // Step 7
    Row_State3 = Row_State4;                                  // Step 8
    Row_State4 = Max(Row_State5,P);                           // Step 9
    Row_State5 = Max(Row_State6,P);                           // Step 10
    Row_State6 = P;                                           // Step 11
// Column machine for grey-scale dilation with a 7 × 7 circle:
    Output_Image[j-3][i-3] = Max(Column_State[j],R1); // Step 12
    Column_State0[i] = Max(Column_State1[j],R2);           // Step 13
    Column_State1[i] = Max(Column_State2[j],R3);           // Step 14
    Column_State2[i] = Max(Column_State3[j],R3);           // Step 15
    Column_State3[i] = Max(Column_State4[j],R3);           // Step 16
    Column_State4[i] = Max(Column_State5[j],R2);           // Step 17
    Column_State5[i] = R1;                                    // Step 18
```

20.12 Automated Generation of Efficient Code for SKIPSM Grey-Scale Image Processing

When the SKIPSM paradigm is applied to grey-scale image-processing operators, either direct code or lookup tables must be created for two finite-state machines (row and column). The computer programs for the creation of lookup tables (**CREATE_ROW_FSM**, etc.) mentioned in Sect. 20.4 are for *binary image processing* only, and therefore are of no use here. In fact, with grey-scale images having more than a few grey levels (16 or so), the lookup-table implementation becomes impractical. Therefore, all of the grey-scale examples presented

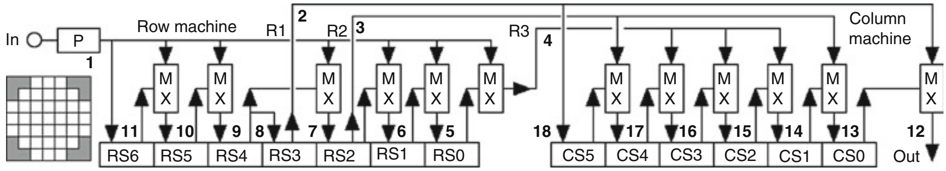


Fig. 20.12
Grey-scale dilation with a 7 × 7 circle

here use *direct* rather than LUT implementations. The essential question then becomes "How does one generate the direct software code for grey-scale applications?" Careful inspection of the examples above shows certain patterns in the column-machine code, but neither guesswork nor intuition could ever lead to the code for the *combined* row machines.

In response to this difficulty, this section presents a *completely routine procedure* for generating efficient compilable C code for *arbitrary operations* (within rather broad limits) on *arbitrary neighbourhoods* specified by the user [25]. No user decisions are required beyond the initial choice of the neighbourhood and of the function to be performed (e.g., grey-scale dilation, grey-scale erosion, area sum, geometric mean, blur, etc.) The specified neighbourhoods may be square or round, "solid" or with holes, connected or disjoint, convex or not convex, and of any size. As a general rule, complex shapes cause more code steps to be generated. There is not enough space here to give the complete routine procedure in detail. Instead, one step-by-step example will be presented.

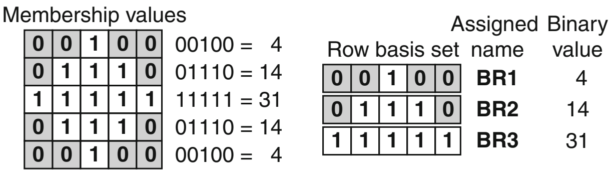
20.12.1 Step 1: Define the Processing Neighbourhood

Start with a user-defined rectangle which is just large enough to encompass all the pixels of interest. ➋ *Figure 20.13* shows the 5 × 5 diamond-shaped processing neighbourhood chosen for this example. Let shaded squares represent pixels within the rectangle which are *not* part of the computation neighbourhood, and assign to them the *membership value* 0 (i.e., MV = 0). Let white squares represent pixels which *are* part of the neighbourhood, and assign to them the *membership value* 1 (i.e., MV = 1). Note that this *does not* imply that binary processing is being done. Ones and zeros are used *only* to indicate that the given pixel belongs to or does not belong to the processing neighbourhood.

Let the membership values for each row (in left-to-right order) be considered as a binary number. Use these numbers to identify and delete all repeated rows. Arrange the non-deleted rows in increasing numerical order. Call this the *row basis set*. In this example there are three rows in the row basis set. Arbitrarily assign names to the three basis rows. Here, the names **BR1**, **BR2**, and **BR3** are used.

20.12.2 Step 2: Create the Preliminary Code

For each of the basis rows, create a sequence of code statements of the form shown below, with as many *code lines* as there are *pixels in the basis row* (including *don't care* pixels), starting with the row having the highest binary value and continuing in descending order of binary value. For each row machine, the right-hand side of the bottom row is always either **G**{P, **null** | 0} or **G** {P, **null** | 1}, where **G** is the special *SKIPSM Generator Function* defined below. The rows above



Membership values

0	0	1	0	0	00100 = 4
0	1	1	1	0	01110 = 14
1	1	1	1	1	11111 = 31
0	1	1	1	0	01110 = 14
0	0	1	0	0	00100 = 4

Row basis set					Assigned name	Binary value
0	0	1	0	0	**BR1**	4
0	1	1	1	0	**BR2**	14
1	1	1	1	1	**BR3**	31

Fig. 20.13
Membership function and row basis set

the bottom rows are “chained” in the sense that the left-hand side of each code line is the second arguments of **G** for the code line above it, with the first argument always being the current input pixel grey value **P**. The actual neighbourhood being implemented is then determined by the 0s or 1s which are inserted as the parameter after the “|” in the function **G** in accordance with the corresponding membership value MV (0 or 1). These membership values are *the only user input* to the code-generation process, other than the original user choice of the 5 × 5 neighbourhood and the choice of the functions **F**r and **F**c, defined below. All the remaining steps are automatic. Here is the resulting preliminary code:

```
R3 = G{ P, Row_State3_0 | 1 };          // Output of row machine 3, to column
                                        machine
Row_State3_0 = G{ P, Row_State3_1 | 1 };
Row_State3_1 = G{ P, Row_State3_2 | 1 };
Row_State3_2 = G{ P, Row_State3_3 | 1 };
Row_State3_3 = G{ P, null | 1 };        // Input to row machine 3, from input
                                        image

R2 = G{ P, Row_State2_0 | 0 };          // Output of row machine 2, to column
                                        machine
Row_State2_0 = G{ P, Row_State2_1 | 1 };
Row_State2_1 = G{ P, Row_State2_2 | 1 };
Row_State2_2 = G{ P, Row_State2_3 | 1 };
Row_State2_3 = G{ P, null | 0 };        // Input to row machine 2, from input
                                        image

R1 = G{ P, Row_State1_0 | 0 };          // Output of row machine 1, to column
                                        machine
Row_State1_0 = G{ P, Row_State1_1 | 0 };
Row_State1_1 = G{ P, Row_State1_2 | 1 };
Row_State1_2 = G{ P, Row_State1_3 | 0 };
Row_State1_3 = G{ P, null | 0 };        // Input to row machine 1, from input
                                        image
```

Notation: Row_State3_0 denotes the first row-state register for row machine 3, Row_State3_1 denotes the second row-state register for row machine 3, etc. R3, R2, and R1 are the row-machine outputs. Note that there are 12 state variables and three outputs.

➲ *Figure 20.14* shows the functional diagram corresponding to this code, which applies to all 5 × 5 SEs with three basis rows. Initially, there is one memory location or register for each row state variable. This is the only memory required for the combined row machine, and (as will be seen below) some of these registers will be eliminated.

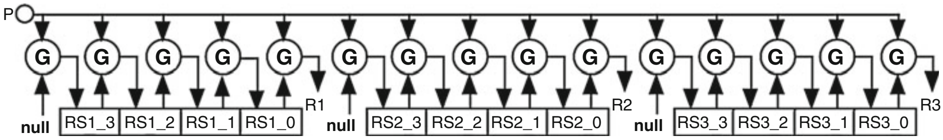


Fig. 20.14
Preliminary diagram for the three generic row machines

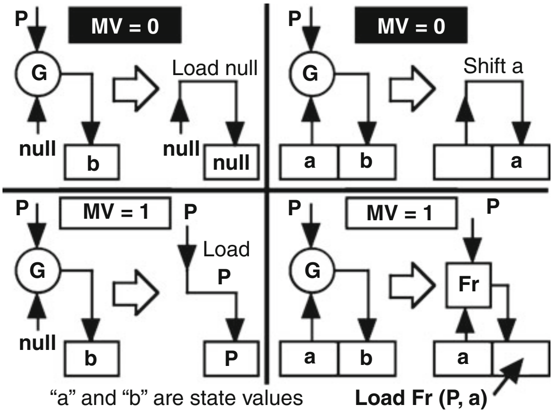


Fig. 20.15
Rules for evaluating G-functions

This code is NOT the final code. It is only the first routine step in the generation of the code. After a series of transformations and simplifications following the deterministic rules given below, these steps will be transformed into executable C-code for the combined row machine, which implements all three row machines.

20.12.3 Step 3: Evaluate G and Propagate the Null

To complete the C-code for these row machines, the generator function **G** must be evaluated according to these rules, also shown graphically in *Fig. 20.15*:

$$\begin{array}{l|l} \mathbf{G}\{\text{P}, \mathbf{null}|0\} = \mathbf{null} & \mathbf{G}\{\text{P}, \text{Row\_StateX}|0\} = \text{Row\_StateX} \\ \hline \mathbf{G}\{\text{P}, \mathbf{null}|1\} = \text{P} & \mathbf{G}\{\text{P}, \text{Row\_StateX}|1\} = \mathbf{Fr}(\text{P}, \text{Row\_StateX}) \end{array}$$

The row-machine function **Fr** specifies the image-processing operation being performed on the rows; e.g., addition, maximum, minimum, etc. **Fr** and the corresponding column-machine function **Fc** will be discussed later. In practice, **Fr** and **Fc** are often the same function, but this is not *necessary*. For example, a minimum-maximum function would use **Fr** = minimum(A,B) and **Fc** = maximum(P,Q).

The **null** variable plays an essential role here: When the input to a register is **null** for whatever reason, *that register itself is eliminated*, and the **null** is passed on to the next-stage **G** function. This is called *propagating the* **null**, and leads to a reduction in the number of registers and hence computing steps. To propagate the **null**, start at the **bottom** of the list for each row machine, and apply the rules for **G**. Then move **up** the list, propagating the **null** as far as it will go.

The resulting code is fully functional at this point. However, when there is more than one row machine, a new possibility arises: Instead of simply computing all the row machines separately, in parallel or sequentially, it is usually possible to reduce the number of state registers (and thus computational steps) by combining expressions with identical right-hand sides. This will now be done for this example.

20.12.4 Step 4: Simplify Recursively

Propagating the **nulls** and evaluating the **G** functions for this example (and using a strikethrough to mark the **null** lines for deletion) gives the code shown in Stage 1 below. Notice that lines 5, 9, and 13 all have the same right-hand side, and hence define states which are always identical. We choose to retain the first state name (Row_State3_3) and discard the other two, after first replacing Row_State2_2 and Row_State1_1 by Row_State3_3 everywhere.

// Stage 1 of simplification

```
// Row machine 1
R3 = Fr ( P, Row_State3_0 ) ;
Row_State3_0 = Fr ( P, Row_State3_1 ) ;
Row_State3_1 = Fr ( P, Row_State3_2 ) ;
Row_State3_2 = Fr ( P, Row_State3_3 ) ;
Row_State3_3 = P ;
// Row machine 2
R2 = Row_State2_0 ;
Row_State2_0 = Fr ( P, Row_State2_1 ) ;
Row_State2_1 = Fr ( P, Row_State2_2 ) ;
Row_State2_2 = P ;
~~Row_State2_3 = null;~~
// Row machine 3
R1 = Row_State1_0 ;
Row_State1_0 = Row_State1_1 ;
Row_State1_1 = P ;
~~Row_State1_2 = null;~~
~~Row_State1_3 = null;~~
```

Stage 2 shows the result of substituting Row_State3_3 for Row_State2_2 and Row_State1_1 and deleting the null steps (lines 10, 14, and 15). We note that lines 9 and 13 are now duplicates of line 5, and mark them for deletion with strikethroughs. We note further that, because of the substitutions, lines 4 and 8 have the same right-hand side. Therefore, states Row_State3_2 and Row_State2_1 can be merged into one state.

// Stage 2 of simplification

```
// Row machine 1
R3 = Fr ( P, Row_State3_0 ) ;
Row_State3_0 = Fr ( P, Row_State3_1 ) ;
Row_State3_1 = Fr ( P, Row_State3_2 ) ;
Row_State3_2 = Fr ( P, Row_State3_3 ) ;
Row_State3_3 = P ;
// Reduced row machine 2
R2 = Row_State2_0 ;
Row_State2_0 = Fr ( P, Row_State2_1 ) ;
Row_State2_1 = Fr ( P, Row_State3_3 ) ;
~~Row_State3_3 = P ;~~
// Row_State3_3 replaces Row_State2_2
// Reduced row machine 3
R1 = Row_State1_0 ;
Row_State1_0 = Row_State3_3 ;
~~Row_State3_3 = P ;~~
// Row_State3_3 replaces Row_State1_1
```

Stage 3 shows the result of deleting lines 9 and 13, replacing Row_State2_1 everywhere by its equivalent state Row_State3_2, and marking line 8 (which is now identical to line 4) for deletion. We *now* notice that lines 3 and 7 have the same right-hand side. Therefore, states Row_State3_1 and Row_State2_0 can be merged into one state.

// Stage 3 of simplification

```
// Row machine 1
R3 = Fr( P, Row_State3_0 );
Row_State3_0 = Fr( P, Row_State3_1 );
Row_State3_1 = Fr( P, Row_State3_2 );
Row_State3_2 = Fr( P, Row_State3_3 );
Row_State3_3 = P ;
// Reduced row machine 2
R2 = Row_State2_0 ;
Row_State2_0 = Fr( P, Row_State3_2 );
~~Row_State3_2 = Fr( P, Row_State3_3 );~~
// Row_State3_2 replaces Row_State2_1
// Reduced row machine 3
R1 = Row_State1_0 ;
Row_State1_0 = Row_State3_3 ;
```

Stage 4 shows the result of deleting line 8, replacing Row_State2_0 by Row_State3_1, and marking line 7 for deletion.

// Stage 4 of simplification

```
// Row machine 1
R3 = Fr( P, Row_State3_0 );
Row_State3_0 = Fr( P, Row_State3_1 );
Row_State3_1 = Fr( P, Row_State3_2 );
Row_State3_2 = Fr( P, Row_State3_3 );
Row_State3_3 = P ;
// Reduced row machine 2
R2 = Row_State3_1 ;
~~Row_State3_1 = Fr( P, Row_State3_2 );~~
// Row_State3_1 replaces Row_State2_0
// Reduced row machine 3
R1 = Row_State1_0 ;
Row_State1_0 = Row_State3_3 ;
```

Stage 5 shows the almost-final code, with line 7 deleted. Since there are no more duplicated right-hand sides, no further reductions are possible. However, the cumbersome state names, needed for the generic formulation, can be replaced, if desired, by something simpler.

// Stage 5 of simplification

```
R3 = Fr( P, Row_State3_0 );
Row_State3_0 = Fr( P, Row_State3_1 );
Row_State3_1 = Fr( P, Row_State3_2 );
Row_State3_2 = Fr( P, Row_State3_3 );
Row_State3_3 = P ;
R2 = Row_State3_1 ;
R1 = Row_State1_0 ;
Row_State1_0 = Row_State3_3 ;
// Optional step: Rename the states:
Row_State3_0 = Row_State0,
Row_State3_1 = Row_State1,
Row_State3_2 = Row_State2,
Row_State1_0 = Row_State3,
Row_State3_3 = Row_State4
```

This is done in the column headed by *Necessary final step*, in which the code lines have been rearranged so that *all* appearances of a state variable on the *right-hand side* of an expression occur in the list *above* its appearance on the left-hand side. This is necessary so that all the occurrences of the *current* values appearing in the code are used for the current computations *before* they are updated due to the arrival of the next pixel value from the raster scan of the input image. The order shown is one of the possible ways to do this.

// Necessary final step:

```
// Rearrange the code so that every use of a variable (right-hand side)
// is executed before the updating of that variable (left-hand side).
R3 = Fr( P, Row_State0 );
R2 = Row_State1 ;
R1 = Row_State3 ;
Row_State0 = Fr( P, Row_State1 );
Row_State1 = Fr( P, Row_State2 );
Row_State2 = Fr( P, Row_State4 );
Row_State3 = Row_State4 ;
Row_State4 = P ; // Next input pixel.
```

After **null** propagation, the original code for the three separate row machines had 12 states, and thus 12 registers and 15 lines of code. This has been reduced to five registers and eight lines

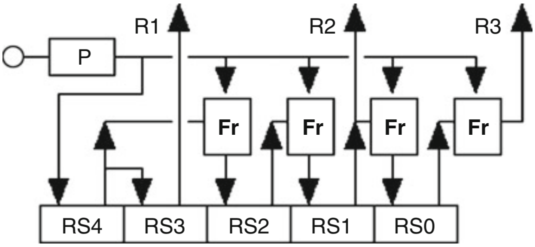


Fig. 20.16
Combined row machine

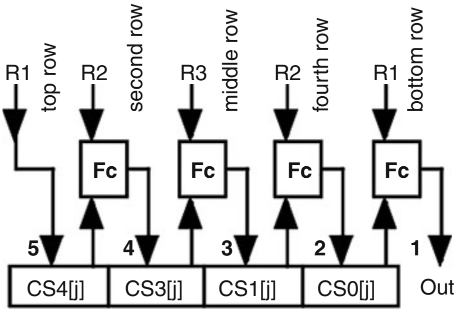


Fig. 20.17
Column machine

of code. More importantly, the number of **Fr** steps, which are often computationally intensive, has been reduced from *six to four*. *Figure 20.16* shows a block diagram corresponding to this code.

20.12.5 Step 5: Create the Column Machine

Compared to combined row machines, column machines are very straightforward. *Figure 20.17* shows the result for this case, as well as the general pattern: The number of column state buffers is always one fewer than the number of rows in the neighbourhood. Each receives its input from the row machine corresponding to its row pattern via the functional block **F**c, which is the function appropriate to the operation being performed. The sequence is right-to-left in the diagram, as indicated by the little numbers. (As always, one must "read before writing.") Here is the corresponding code:

```
Output_Image[j-2][i-2] = Fc( R1, Column_State0[j]);
Column_State0[j]       = Fc( R2, Column_State1[j]);
Column_State1[j]       = Fc( R3, Column_State2[j]);
Column_State2[j]       = Fc( R2, Column_State3[j]);
Column_State3[j]       = R1;
```

Because of the top-to-bottom symmetry of the neighbourhood in this particular example, one can not tell from this diagram which order (left-to-right or right-to-left) the row-machine outputs should be applied, so the diagram includes this information also. This is the pattern, in general, for all five-row column machines.

Comments on functions **F**r *and* **F**c: A wide range of operators may be used for **F**r and **F**c. Examples: maximum, minimum, addition, multiplication, exponential decay. Some other operators *will not work*. Examples: subtraction, division, median (or any rank *except* maximum and minimum). The sole requirement on the functions is that they have what could be called a uni-directional associative property, defined thus: Let a, b, c, d, e, f, g, . . . denote a string of pixels *presented in raster-scan order*. Then, for **F**r to be used with a recursive implementation such as SKIPSM, **F**r must satisfy

$$\mathbf{F}\mathrm{r}(c, \mathbf{F}\mathrm{r}(b, a)) = \mathbf{F}\mathrm{r}(c, b, a), \quad \mathbf{F}\mathrm{r}(d, \mathbf{F}\mathrm{r}(c, \mathbf{F}\mathrm{r}(b, a))) = \mathbf{F}\mathrm{r}(d, c, b, a), \quad \text{etc.}$$

The uni-directional requirement is just another way of saying that the functions need not be commutative. Most operators of interest are both associative and commutative, but this is not necessary.

Examples which meet the uni-directional associative condition:

Fr = Addition	$c + (b + a) = c + b + a$
Fr = Multiplication	$c \bullet (b \bullet a) = c \bullet b \bullet a$
Fr = Maximum	Maximum(c, Maximum(b, a)) = Maximum(c, b, a)
Fr = Minimum	Minimum(c, Minimum(b, a)) = Minimum(c, b, a)

Examples which fail to meet the condition:

Fr = Subtraction	$c - (b - a) \neq c - b - a$
Fr = Median	Median(c, Median(b, a)) may not equal Median(c, b, a)

The column function must meet the same condition. Let A, B, C, D, E, . . . represent row-machine outputs presented to the column machine in raster-scan order: Then **F**c(C, **F**c(B, A)) must equal **F**c(C, B, A), etc. **Fr** and **Fc** are usually the same function, but this is not necessary. For example, **Fr** = Minimum, **Fc** = Maximum is a useful combination. (See the 4M filter example in ➔ Sect. 20.8.)

20.13 Speed Comparisons

In the late 1990s, comparison tests were made for three examples to estimate the relative execution speeds of SKIPSM implementations versus conventional implementations. As the reader will note from the discussion below, familiar measures of computing power (such as processor speed) turned out to be not particularly reliable in estimating the relative speeds of the various implementations. Computer *architecture* seems to play an important role, not just raw computing power. Ideally, similar tests would be performed using currently-available computing platforms and architectures, in order to get a realistic picture of what speeds one could expect today. However, there is neither budget nor equipment available to perform these tests. Therefore, the conclusions about relative speeds should be taken as *indications* of what one might expect, but should not be considered as definitive.

Test 1 – Gaussian Blur: One set of tests was for Gaussian blurs. Because the tests were made some years ago, computing platforms which are now obsolete were used. Nonetheless, the *relative-speed* results are still valid. A comparison was made between SKIPSM and two other approaches for implementing Gaussian filters. The "brute force" technique used the 5 × 5 kernel given below. In order to get a filtered pixel value, 25 multiplications and 24 additions are needed.

1	4	6	4	1	Normalizing constant: 256
4	16	24	16	4	
6	24	36	24	6	
4	16	24	16	4	
1	4	6	4	1	

The second approach used the usual one-dimensional decomposition [1 4 6 4 1] and [1 4 6 4 1]-transposed.

Each program was compiled using the DOS port of GCC (DJGPP) using -O3 for optimization. No attempt was made to use the special Pentium® compiler with -O6 optimization but results would probably be around 10–15% faster with it. The different approaches were all tested on a PC-compatible computer using a 166 MHz AMD K6 CPU with 512 KB of secondary cache and 64 MB of memory under Windows 95®. There were a number of background processes running which may have increased times slightly but would not invalidate the comparisons since all tests were performed under the same conditions. Given that the execution times for a single image were under a second for all of the approaches, processing was repeated on the input image as needed so that the total time was on the order of 1 min. The average times for each iteration were calculated for the three different methods, and are as follows:

Implementation approach	Execution time per iteration
5 × 5 conventional convolution	919 ms
5 × 5 decomposition	450 ms
5 × 5 SKIPSM	108 ms

As expected, implementing the Gaussian blur filter through standard convolution was relatively slow. Decomposition was about twice as fast. But the best results by far were obtained with the SKIPSM approach, which was almost nine times as fast as standard convolution. The speed advantages of the SKIPSM implementation should be even greater for larger values of N.

Test 2 – Binary morphology with large rectangular structuring elements: Speed comparisons were made between the SKIPSM LUT method, the SKIPSM direct-code method, sequential decompositions, and "brute force" code (rectangles only). All tests were on 512 × 512 images. Averages were taken over 500 repeats of each operation. The system used was a Micron Millennia Mme, model Anchorage-233M-T, with 160 MB of RAM and an Intel Pentium MMX with a chip clock speed of 233 MHz and a bus clock speed of 66 MHz. This system has 512 KB of Level 2 SRAM cache and 640 KB of system RAM cache. The operation system was Windows 95 version 4.00.1111, booted in DOS mode. Code was compiled using DJGPP Rhide with no options set.

Comments on "brute-force" code: For each input pixel, this code fetches and examines all the pixels in the M×N neighbourhood, *but stops as soon as it finds a black pixel.* This is a "smart" way to do brute-force processing, possible with binary erosion but not with most other operations. However, it makes the timing very dependent on the number of object (white) pixels in the image. The image used for these timing tests happened to have relatively few white pixels. Therefore, the results (➋ *Fig. 20.18*) should be taken as very optimistic for the "brute force" code. The times for all the other methods are dependent on the *size of* the image but totally independent of the *content* of the image.

These results were surprising. We expected the direct SKIPSM code to be fast, as software implementations go, but we did not expect that in some cases it would be 25% faster than the LUT method, which until then had always been the "speed champion," consistently outperforming both "brute-force" methods and sequential decompositions for morphological operations. Additional comment: In some later tests (with a faster PC, but still ancient by today's standards) speeds exceeding 30 frames/s were achieved on similar operations.

However, note that for sufficiently large non-rectangular SEs the SKIPSM LUT method does *not* slow down measurably, but the direct-code method does. Thus, for large or complex

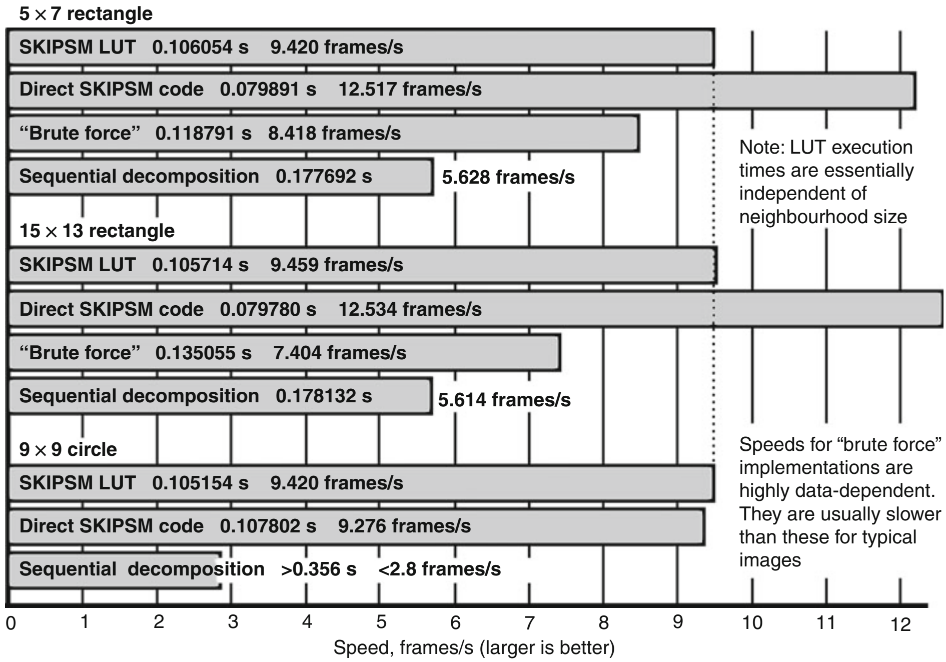


Fig. 20.18

Execution speed comparisons for binary morphology. Longer bars indicate faster execution

SEs, the LUT approach will again win out. This expectation is supported by the results in these tests. The dotted line overlaid on *Fig. 20.18* shows that execution times of the three SKIPSM LUT implementations (neighbourhood sizes 5 × 7 = 35, 15 × 13 = 195, and 9 × 9 = 81) were only trivially different, and hence are approximately independent of neighbourhood size. We suspect that part of the reason for the slower-than-expected times for the LUT method results from the fact that the Intel MMX vector technology used for these comparisons generally does a poor job with lookup tables, so that the speed increases hoped for with MMX increase the speed of direct implementations but do not materialize for the LUT implementations.

It would be ideal for SKIPSM implementations of all types (not just this one), if the chip architecture allowed at least part of the high-level chip cache to be dedicated to functions *specified by the user*. For the class of applications described here, the column state buffer would be left in cache at all times. For other SKIPSM applications, the lookup tables would be placed there. This is not likely to happen, because such partial dedication of the cache space requires additional on-chip cache-management hardware, which vendors are unwilling to provide.

Also, it would be interesting to see how these approaches would compare if a RISC machine were used. These tests have not been made. In earlier tests in which direct comparisons were made, Macintosh computers based on PowerPC RISC chips significantly and consistently outperformed Intel CISC chips of comparable clock speeds.

Conclusions: Speed comparisons for additional image-processing operations would be desirable, but it could be argued that additional tests are really not necessary, because it is easy to count the number of code steps in the inner loop in various implementations to get a *very* good idea of comparative speeds. Conventional implementations typically have more steps that their SKIPSM counterparts.

References

1. Waltz FM (1994) SKIPSM: Separated-kernel image processing using finite-state machines. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration III, Paper 2347–36, Boston, November 1994
2. Waltz FM, Garnaoui HH (1994) Application of SKIPSM to binary morphology. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration III, Paper 2347–37, Boston, November 1994
3. Waltz FM, Garnaoui HH (1994) Fast computation of the Grassfire Transform using SKIPSM. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration III, Paper 2347–38, Boston, November 1994
4. Waltz FM (1996) Automated generation of finite-state machine lookup tables for binary morphology. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration V, pp. 249–258, Boston, November 1996
5. Waltz FM (1997) Binary dilation using SKIPSM: Some interesting variations. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration VI, Paper 3205–3215, Pittsburgh, October 1997
6. Hack R, Waltz FM, Batchelor BG (1997) Software implementation of the SKIPSM paradigm under PIP. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration VI, Paper 3205–3219, Pittsburgh, October 1997
7. Waltz FM, Miller JWV (1999) Execution speed comparisons for binary morphology. In: Proceedings of the SPIE conference on machine vision systems for inspection and metrology VII, Paper 3836–3901, Boston, September 1999
8. Waltz FM (1994) Application of SKIPSM to binary template matching. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration III, Paper 2347–2439, Boston, November 1994
9. Waltz FM (1995) Application of SKIPSM to binary correlation. In: Proceedings of the SPIE Conference on machine vision applications, architectures, and systems integration IV, Paper 2597–2611, Philadelphia, October 1995
10. Waltz FM, Miller JWV (1999) Comparison of connected-component algorithms. In: Proceedings of the SPIE conference on machine vision systems for inspection and metrology VIII, vol 3836–3902, Boston, September 1999
11. Waltz FM (1994) Application of SKIPSM to grey-level morphology. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration III, Paper 2347–2440, Boston, November 1994
12. Miller JWV, Waltz FM (1997) Software implementation of 2-D grey-level dilation using SKIPSM. In: Waltz FM, Miller JWV (eds) An efficient algorithm for Gaussian blur using finite-state machines. Proceedings of the SPIE conference on machine vision systems for inspection and metrology VII, Paper 3205–3218, Pittsburgh, October 1997
13. Waltz FM, Miller JWV (1998) An efficient algorithm for Gaussian blur using finite-state machines. In: Proceedings of the SPIE conference on machine vision systems for inspection and metrology VII, Paper 3521–3537, Boston, November 1998
14. Waltz FM (1995) SKIPSM implementations: Morphology and much, much more. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration IV, Paper 2597–2614, Philadelphia, October 1995
15. Waltz FM (1998) The application of SKIPSM to various 3 x 3 image processing operations. In: Proceedings of the SPIE conference on machine vision systems for inspection and metrology VII, Paper 3521–3530, Boston, November 1998
16. Waltz FM, Miller JWV (1998) Fast, efficient algorithms for 3 x 3 ranked filters using finite-state machines. In: Proceedings of the SPIE conference on machine vision systems for inspection and metrology VII, Paper 3521–3531, Boston, November 1998
17. Hujanen AA, Waltz FM (1995) Pipelined implementation of binary skeletonization using finite-state machines. In: Proceedings of the SPIE conference on machine vision applications in industrial inspection, Paper 2423–2, San Jose, February 1995
18. Hujanen AA, Waltz FM (1995) Extending the SKIPSM binary skeletonization implementation. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration IV, Paper 2597–2612, Philadelphia, October 1995
19. Waltz FM (1994) Application of SKIPSM to the pipelining of certain global image processing operations. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration III, Paper 2347–2441, Boston, November 1994
20. Waltz FM (1998) Image processing operations in color space using finite-state machines. In: Proceedings of the SPIE conference on machine vision systems for inspection and metrology VII, Paper 3521–3533, Boston, November 1998
21. Shamos MI (1978) Robust picture processing operators and their implementation as circuits.

In: Proceedings of the fall 1978 workshop on image processing, Carnegie Mellon University

22. Narendra PM (1981) A separable median filter for image noise smoothing. IEEE Trans Pattern Anal Mach Intell 3(1):20–29. doi:10.1109/TPAMI.1981.4767047
23. Waltz FM (1996) Binary openings and closings in one pass using finite-state machines. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration V, Paper 2846, Boston, November 1996
24. Waltz FM (1997) Implementation of SKIPSM for 3-D binary morphology. In: Proceedings of the SPIE conference on machine vision applications, architectures, and systems integration VI, Paper 3205–3213, Pittsburgh, October 1997
25. Waltz FM (1998) Automated generation of efficient code for grey-scale image processing. In: Proceedings of the SPIE conference on machine vision systems for inspection and metrology VII, Paper 3521–3532, Boston, November 1998

21 QT – Prototyping Image Processing System

Bruce G. Batchelor[1] · *Simon J. Caton*[2]
[1]Cardiff University, Cardiff, Wales, UK
[2]Karlsruher Institut für Technologie (KIT), Karlsruhe, Germany

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_21,

Abstract: QT is the latest in a series of interactive image processing tool-kits for prototype development of Machine Vision systems. Experience gained over more than 35 years has shown that a command-line system with short mnemonic function names is an effective tool for developing vision algorithms. Parameter lists are also required for some functions. When the user types a command, such as *neg* (negate), *thr(89,127)* (threshold) or *lpf* (low-pass filter, blur), the operation is performed immediately and the result is displayed, together with the image before processing. When running on a modern computer, typical response times for processing an image of good resolution are rarely more that one second. Measurements are obtained in an obvious way. For example, to count the number of distinct objects (blobs) in a binary image, the user types $x = cbl$, while the command $y = avg$ computes the average intensity in a grey-scale image. Binary, grey-scale and colour images are all processed by QT, which is built on top of MATLAB™. The advantage of QT over MATLAB lies in its greater ease of use. The command names are short but most importantly, the user does not have to keep track of source and destination images. QT always processes the image displayed on the left of the computer screen. It then replaces this with the result of the operation selected by the user and displays the input image for the last processing step on the right. In addition, to image processing commands, QT provides facilities for a range of other operations, including image capture from local or remote cameras, controlling external devices, speech synthesis, on-line documentation, consulting the Catalogue of Lighting-Viewing methods (➲ Chap. 40). Any MATLAB function can be invoked from QT and programming new functions is straightforward. Many of the illustrations in the book were prepared using QT and several chapters exploit its ability to express image processing algorithms succinctly. Since it is a prototyping tool, QT is not fast enough for any but the least demanding factory applications. However, it is a powerful exploratory tool in the hands of an experienced user. On numerous occasions, QT or one of its predecessors, has been used to discover/invent algorithms very quickly in response to new visual inspection and control applications. QT has been interfaced to Prolog to produce an even more powerful language (PQT) that facilitates intelligent reasoning about images (➲ Chap. 23).

21.1 Introduction

QT is a software package for interactive image processing and was specifically designed to assist in the development of prototype Machine Vision systems. It allows rapid selection and testing of image processing algorithms and provides a large repertoire of easily executable operators: simply type the name of the appropriate function and, if necessary, a short list of control parameters. QT's antecedents were originally used exclusively in laboratory studies of tasks relating to Automated Visual Inspection and were successful in demonstrating feasibility in a very wide range of applications. Later, similar systems were used to control robotic devices, again in the laboratory. There has always been a clear distinction between target systems that are installed in a factory and prototyping systems, like QT, that are used to design them. Interactive image processing systems are also valuable for education, training and convincing potential investors in this technology. However, there is also a need for easily programmable, multi-function vision systems that can be used in factory-floor experiments. These are often required to convince sceptical managers who fear that this technology is unproven, unreliable, or impractical. While QT has the image processing capability for this purpose, the software must be able to control the appropriate hardware: lamps, cameras, solenoids, relays, lathes, machines for drilling, cutting, grinding, moulding, spraying, polishing, parts handling, etc. and even multi-axis robots. Ideally, we would like to provide QT with a "universal" interface module that would allow it to be connected immediately to any type of external device but clearly this is impossible. However, it is possible to provide it with a versatile interface module that allows it to operate a wide range of types of external equipment.

QT is the most recent in a family of similar systems, originating with SUSIE (Southampton University, 1976) [1] and including Autoview (British Robotics Systems Ltd.) VCS (Vision Dynamics Ltd.) [8], System 7 (3M Company), SuperVision (Image Inspection Ltd.), Prolog+ (Cardiff University) and PIP (Cardiff University) [4]. QT has been implemented using the MATLAB Image Processing Toolbox and employs MATLAB's programming syntax [MATLAB, Mathworks, Inc, URL http://www.mathworks.com/, accessed 26th November 2009]. QT employs the same function names as its predecessors, just mentioned. It is possible to extend QT indefinitely, simply by writing additional MATLAB M-files and placing them in the appropriate folder.

A copy of QT is supplied with this Handbook. The latest version of the software and a number of other utilities for Machine Vision system design, can be accessed by visiting http://bruce.cf.cs.ac.uk

21.2 Paradigm for Interactive Image Processing

Unlike many image processing systems, QT uses and displays only two images, called *Current* and *Alternate* (➋ *Fig. 21.1*). The *Current* Image is always displayed on the left and the *Alternate* Image on the right. When a typical command, such as *neg* (negate) is typed, the following happens:

1. The *Current* Image is copied into the *Alternate* Image. The original contents of the *Alternate* Image are discarded.
2. The specified operation is performed on the *Current* Image.
3. The result (negative image) is then stored in the *Current* Image.
4. The revised *Alternate* Image is also displayed alongside the Current Image.

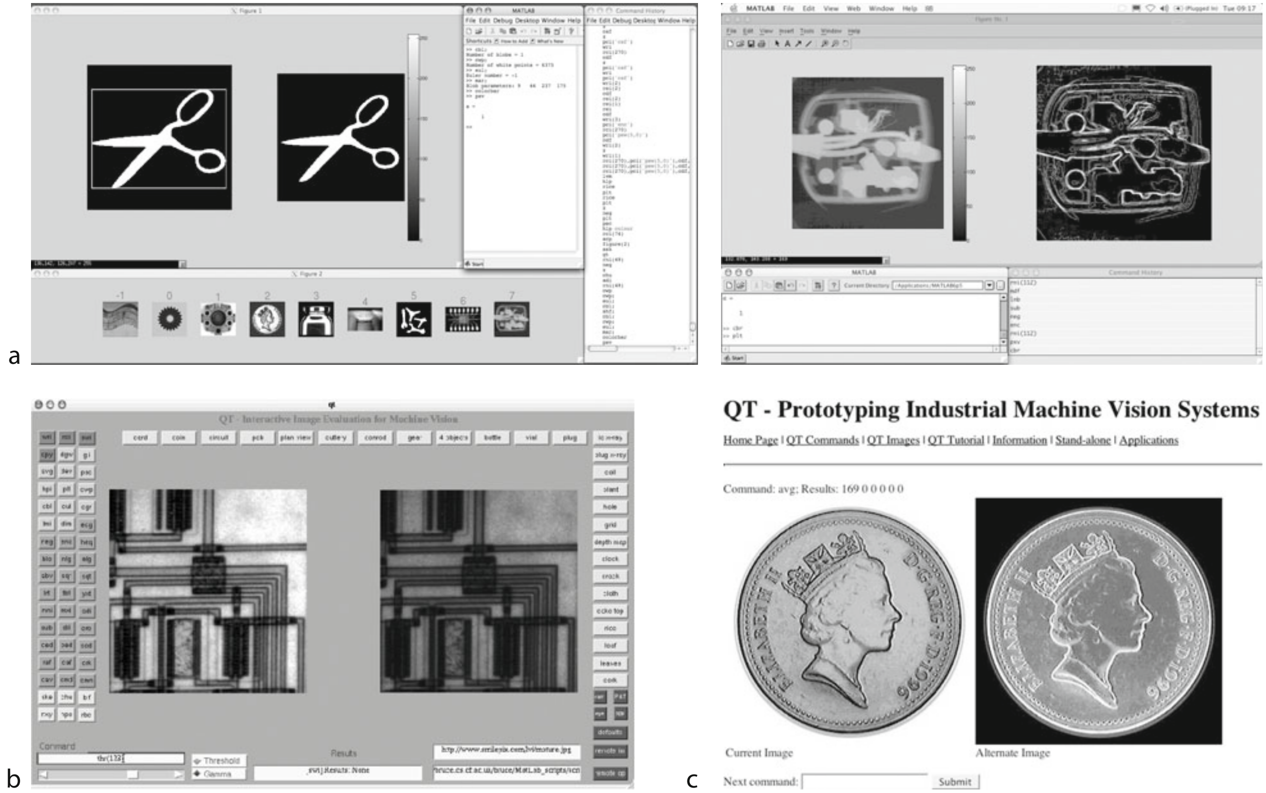
See ➋ *Fig. 21.2* Thesholding (function *thr*) requires zero, one or two parameters; the following are typical commands:

• *thr(123,234)*	Two arguments supplied
• *thr(156)*	Second argument is, by default, assumed to be 255 (white)
• *thr*	First argument default value is 128. Second is 255.

thr, with and without arguments, *neg* and most other QT commands, rely on the *Current-alternate* operating paradigm. However, there are some deviations from this model, reflecting the different types of operators. There are five other situations to consider:

(a) Some commands return values calculated from the Current Image but do not alter it, nor the Alternate Image. For example, *avg* computes the average grey level. Neither image is modified during this calculation.

(b) Some commands generate images. For example, *gsn* generates a random image from a Gaussian noise source. In this case, the *Current* Image is first copied to the *Alternate* Image. The *Current* Image is then replaced by the Gaussian noise image. This is, of course, compatible with the operation of the majority of commands, such as *neg*. Reading an image from a disc file is also performed in this way.



```
>> cbl;
Number of blobs = 1
>> cwp;
Number of white points = 6375
>> eul;
Euler number = -1
>> mar;
Blob parameters: 9   44  237  175
>> colorbar
>> pev

a =

     1

>>
```

thr(133)

,swi] Results: None

http://www.smileyis.com/bvi/mature.jpg

bruce.cs.cf.ac.uk/bruce/MatLab_scripts/scn

QT - Prototyping Industrial Machine Vision Systems

Home Page | QT Commands | QT Images | QT Tutorial | Information | Stand-alone | Applications

Command: avg; Results: 169 0 0 0 0 0

Current Image

Alternate Image

Next command: Submit

■ Fig. 21.1

QT control/display screens. (a) The Current Image is the large image on the *left*. The Alternate Image is on the *right*. The image stack is displayed, in part, as a set of thumbnails. (b) Window mode provides control buttons to read standard images and perform popular functions. Notice too the slide controls for thresholding and gamma adjustment

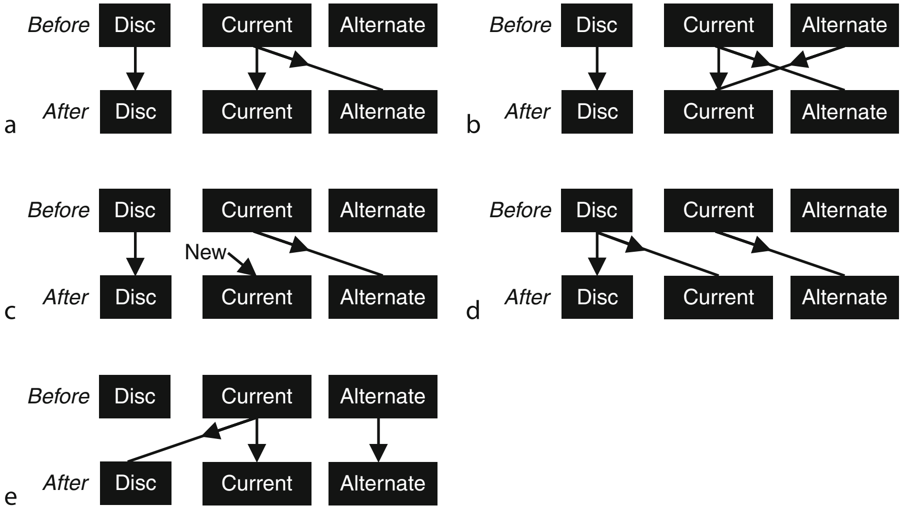


Fig. 21.2
Mapping the Current and Alternate Images during execution of QT functions. (a) Most commands behave in this way (e.g., *neg, thr, caf, chu*). (b) Image generators. (e.g., *gsn, hic, wgx*). (c) Dyadic operators (e.g., *adi, sub, mxi, mni*). (d) Reading images (e.g., *rei, rni, rci*). (e) Saving images (e.g., *wri*)

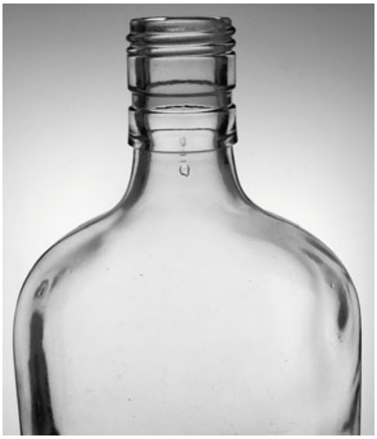


Fig. 21.3
Glass bottle, viewed with back lighting

(c) Commands that write images to a disc file do not alter either image.
(d) A few commands, such as *scp* (plot RGB scattergram) generate a new display window (MATLAB refers to this as Figure 4), without altering either the *Current* or *Alternate* Image.

> *Note*: The user should close MATLAB's Figure 4 before typing the next command, otherwise the results will be placed there, rather than the customary display window: MATLAB's Figure 1.

(e) A few commands combine the *Current* and *Alternate* Images. For example, function *adi* adds the intensities of pixels at corresponding addresses in the *Current* and *Alternate* Images. In such cases, the result is placed in the *Current* Image and the original *Current* Image is copied into the *Alternate* Image, whose original contents are discarded.

21.3 Command Syntax

QT commands can be concatenated as in the following example:

- ► wri(100), dil, ero, rei(100), sub, enc% Performs grey-scale edge detection using morphology

Alternatively, when writing new MATLAB scripts or functions, individual commands can be written on separate lines, with optional comments added as illustrated below:

```
wri(100)        % Write the Current Image to disc file 100.jpg
dil,            % Dilation using default parameters
ero             % Erosion using default parameters
rei(100),       % Read image file 100.jpg
sub,            % Subtract the Alternate Image from the Current Image
enc             % Enhance contrast
```

Notice that commas can be included at the end of each line but are not necessary. (Semicolons, which will be considered next, would have no effect in this example.)

Some commands generate numerical results. For example *avg* calculates the average intensity, which can be used by a later command. The following command sequence shows how this is achieved:

$$x = avg,\ \ thr(x)$$

avg calculates the average intensity and *thr* performs simple thresholding. Notice the comma separating these two commands. The effect of this comma is to allow MATLAB to display the value of the newly assigned variable (x). This is a design feature of MATLAB. To suppress the display of this variable, we must use a semi-colon instead:

$$x = avg;\ \ thr(x)$$

If we wish to place these commands on separate lines to make reading easier, the semi-colon again suppresses the display of the values for x:

```
x = avg;                 % The value of x is not displayed
thr(x)                   % Threshold at level x
```

Whole images can be assigned using a construction of the form

$$\text{current} = \text{original}$$

> *Warning*: This displays the intensities for the whole image, pixel by pixel. For a colour image, there are three values displayed for each pixel. This is a lot of data and the command takes a long time to execute!

To suppress the output display, use a semi-colon:

$$\text{current} = \text{original};$$

Some commands generate several output values, as the following example shows (This is the heart of the function *nxy*, which normalises the position of a blob so that its centroid lies at the centre of the image.)

```
[x1,y1] = cim          % No semi-colon, so the values of x1 and y1 are
                         displayed.
[x2,y2] = cgr          % No semi-colon, so the values of 2x and y2 are
                         displayed.
q = round(x1-x2);      % Standard MATLAB rounding functions. Output is
                         suppressed.
p = round(y1-y2);      % Standard MATLAB rounding functions. Output is
                         suppressed.
current = original;    % Take care to place a semi-colon here to avoid long
                         delays.
psw(p,q)               % Shift image horizontally by p pixels and
                         vertically by q pixels.
```

21.4 Image Formats

QT uses the following image representations:

Monochrome images (i.e., grey-scale images): 2-dimensional MATLAB arrays of 8 bit integers. Black is 0 (zero) and white is 255. Individual pixels in the *Current* Image can be accessed thus;

$$x = \text{current}(y - \text{address},\ x - \text{address})$$

> *Warning*: This is contrary to conventional mathematical notation. QT and its predecessors follow the normal convention that the *x*-address precedes the *y*-address. Hence, it is better to use the QT functions *civ* and *siv* when accessing individual pixels; we can use
> $x = \mathit{civ}\ (x - \text{address},\ y - \text{address})$
> and
> $\mathit{siv}\ (x - \text{address},\ y - \text{address},\ \text{intensity})$
> to read and set individual pixels in the Current Image. Also note that the image must be declared as being *global* in the *Command Window*, before we can access the pixels directly. Again, to avoid confusion, it is better to use *dgw* to determine the image size, instead of MATLAB's *size* function:
> $[\text{width},\ \text{height}] = \mathit{dgw}$

Binary images are represented in QT as special cases of monochrome images. Black is zero and white is 255. MATLAB has a special structure for binary images but this is not used in QT, to maintain consistency. (It is advantageous to use grey-scale operators to process binary images. For example, *neg* (grey-scale negation) performs logical inversion (*NOT*) on binary images.)

Colour images are represented by 3-dimensional arrays. The pixels can be accessed, using MATLAB's addressing convention, thus:

$$\text{current}\,(y-\text{address},\ x-\text{address},\ \text{colour\_plane})$$

where *colour_plane* is an integer in the range 1–3. Plane 1 is the *R*-channel; plane 2 is the *G*-channel and plane 3 is the *B*-channel. Again, it is preferable to use *civ* and *siv* to maintain a consistent addressing format.

Image files held on disc in JPEG and TIFF formats can be read using the function *rei*. (Also see functions *rea*, *rim*, *rni* and *rci*.) Other file formats, including BMP, GIF and BMP, can be read by MATLAB, so it is a straightforward matter to extend this range, as necessary.

> *Note*: The JPEG format is liable to produce artefacts if it is used to save images that are later compared to the original (➤ *Table 21.5*). The reason is that reading an image file with JPEG coding regenerates an approximation to the original image. Furthermore, when a binary image is saved in JPEG format, the image read back from the disc is not binary. Hence, it is necessary to apply the threshold function *thr* before applying any of the binary operators. The TIFF format, which is forced to be loss-less, regenerates an exact copy of the original image and hence avoids these problems altogether.

21.5 Writing M-Files

QT commands are implemented as MATLAB functions (*M-functions*), which are stored in M-files. It is quite easy for the user to extend the QT command repertoire, by adding his/her own M-functions. This section shows how this can be done and also explains how MATLAB scripts can be used. It is assumed that the reader is already familiar with MATLAB programming but not necessarily its Image Processing Tool-box.

21.5.1 Defining Simple QT Command Sequences (MATLAB Scripts)

Consider the following command sequence, which will form the body for a new script, called *crack_detector*.

```
rni(41)                                % Read file number 41 (Crack image)
crk(5)                                 % Apply the crack detector
thr(140)                               % Threshold
kgr(100)                               % Keep only those blobs with area
                                         > 100 pixels
```

```
x = cbl;                                    % Count the blobs
y = cwp;                                    % Count the white pixels
if x > 0                                    % Test number of blobs
  s = ['Crack detected; total no. of        % Construct message string
    white pixels = ', int2str(y)];
  display(s)                                % Display it
else
  s = ['No crack found'];                   % Construct a message string
  display(s)                                % Display it
end
```

First, the user defines a new M-file and enters the command sequence listed above. To incorporate the new script in QT, *crack_detector* can then be saved in any folder that lies on MATLAB's search path. The resulting script can then be executed by typing *crack_detector* in MATLAB's *Command Window.*

Notice that scripts cannot refer explicitly to the *Current* or *Alternate* Images. To do so, we must define a new MATLAB function. It is also better to use M-functions, rather than scripts, when we need to define input variables to control the processing, or calculate values that can be used in subsequent commands.

21.5.2 Adding New QT Functions

Here is an annotated listing of the function *ftm* (Log-magnitude of the Fourier transform), which shows the structure that is shared by many of the QT functions:

```
function ftm                        % Function name
% Fourier transform (log magnitude) [0/1]
global current                     % Defines the Current Image to be global
global alternate                   % Defines the Current Image to be global
original = current;                % Save Current Image until the end
a = log(abs(fft2(current)));       % Specific to this function
b = max(max(a))                    % Specific to this function
c = min(min(a))                    % Specific to this function
d = double(255/(b-c));             % Specific to this function
current = uint8(d*(a-c));          % Specific to this function
[e,f] = cim;                       % Specific to this function
psw(e,f)                           % Specific to this function
alternate = original;              % Original Current Image now in Alternate
                                     Image
subplot(1,2,1),                    % Display Current Image in frame of
  imshow(current)                    Figure 1
subplot(1,2,2),                    % Display Alternate Image in frame 2
  imshow(alternate)
```

Notice that QT functions (*cim* and *psw* in this case) can be incorporated within the definitions of other QT functions. QT functions can also be recursive. Also, note that MATLAB functions (e.g., *log*, *abs*, *min*, *max*, *double*, *uint8* in this example) can also be used freely.

Writing new QT functions that use control parameters and/or return values follows the MATLAB convention. Here is the listing of the QT function *eul*, which has one input (*b*) and one output (*a*).

```
function a = eul(b)               % b is the input parameter and a is the
                                    output
% Euler number for 4- or 8-connected blobs [1/1]
global current                    % Defines the Current Image to be global
global alternate                  % Defines the Current Image to be global
if nargin == 0                    % Tests number of arguments (nargin)
  b = 8;                          % Set b to value 8, Orders 8-connectivity
end
level = graythresh(current);      % Specific to this function
working = im2bw(current,          % Specific to this function
level);
a = bweuler(working,b)            % Assigns a value to the output variable a
```

Functions, such as *eul*, with both input and output parameters can be invoked in a number of ways:

- `eul`   Default value for input. Prints Euler number.
- `eul(4)`   Prints 4-connected Euler number.
- `a = eul`   Default value for input. Assigns a to the Euler number.
- `a = eul(8)` Assigns a to value of the 8-connected Euler number.

Functions can use several input arguments and several output arguments. Here is an example that has no inputs and four outputs.

```
function [a, b, c, d] = bbx
% BoundingBox – Smallest rectangle containing the blob [4/0]
q = blob_parameters        % Uses an auxiliary M-file ‘‘blob_parameters’’
  (‘BoundingBox’);
a = q(1);                  % Assigns a value to the output variable a
b = q(2);                  % Assigns a value to the output variable b
c = q(3) + q(1);           % Assigns a value to the output variable c
d = q(4) + q(2);           % Assigns a value to the output variable d
```

bbx may be called with zero to four outputs. The following function has five inputs and no outputs.

```
function vpl(x1,y1,x2,y2,v)
% Draw a straight line given two points [0/5]
global current              % Defines the Current Image to be global
global alternate            % Defines the Current Image to be global
original = current;         % Save Current Image until the end
[h,w] = size(current);
if nargin == 4              % Default: only four inputs
  v = 255;                  % Draw a white line
end
x1 = double(x1);
```

```
x2 = double(x2);
for i = 0:0.0001:1
  p = round(x1 + i*(x2 – x1));
  q = round(y1 + i*(y2 – y1));
  if p >= 1 & p <= w & q >= 1 & q <= h
    current(q,p) = v;
    working(q,p) = v;
  end
end
subplot(1,2,1),                  % Display Current Image in frame of
  imshow(current)                  ➜ Fig. 21.1
subplot(1,2,2),                  % Display Alternate Image in frame of
  imshow(alternate)                ➜ Fig. 21.1
```

21.5.3 Updating QT

QT uses four folders:

(a) *BGB_new_mfunctions* (Holds updates to standard QT functions, down-loaded from the author's web site.)
(b) *BGB_MATLAB_work_files* (The user can add new scripts/functions to QT or define new functions, by placing the appropriate M-files in this folder.)
(c) *BGB_MATLAB_Scripts* (Contains QT's standard library of M-files.)
(d) *BGB_MATLAB_Images* (Contains numbered image files in JPEG or TIFF format. Users can add extra files at will.)

MATLAB's search path should be set up, so that these folders are visited in the order listed above, viz *(a,b,c,d)*. Updated QT functions can be switched ON/OFF, using *qtp*:

qtp(0)	Updated functions in *BGB_new_mfunctions* are disabled
qtp(1)	Updated functions in *BGB_new_mfunctions* are enabled
qtp	Updates are toggled ON/OFF

With the search path set up as described above, any function defined in *BGB_new_mfunctions* is executed in preference to the standard function with the same name held in *BGB_MATLAB_Scripts*.

21.6 User Interface and Utilities

Several factors determine the effectiveness of an interactive image processing system. Of these, the ease of use is critical. The choice of short mnemonic commands, coupled with the *Current-Alternate* operating paradigm have been crucial in making this approach highly successful. QT is the latest development in a series of similar systems that has been evolving for over 30 years. QT possesses a number of operational features that make it significantly easier to use and hence more powerful than even its most recent antecedents.

21.6.1 Modes of Operation

QT has been provided with three modes of operation:

- *Window Mode* (➲ *Fig. 21.1b*)
  Novice users (Author uses this mode for teaching university students.)
  Single window, fixed format display
  Shows only current and alternate pictures
  Buttons, loading standard test images
  Buttons, executing commonly used functions
  Sliders for adjusting threshold and gamma
  Text box for command-line operation
  Restricted functionality (It is difficult to use with functions requiring additional windows)
  Unsuitable for editing M-files
- *Stack Mode* (➲ *Fig. 21.1a*)
  Intermediate level users
  User can reformat/resize image display
  Displays current and alternate pictures
  Displays thumbnails of images in the stack (User clicks on a thumbnail to read the corresponding image from the stack.)
  Buttons to load standard test images
  Buttons to execute commonly used functions
  Keyboard short cuts for common operations (e.g., *pop* and *push* the stack)
  Sliders for adjusting threshold and gamma.
  Most MATLAB features are available (e.g., History Window, cursor, 3D plots, etc.)
  Does not permit editing of M-files
- *Terminal Mode* (➲ *Fig. 21.1a*)
  Experienced users (used for software development)
  As Stack mode in most respects but more functions are available
  Stack display is used to show saved images (accessible at random)
  Permits M-files to be added/edited

➲ *Table 21.1* indicates the features available in these different modes.

21.6.2 Help

There are several ways to obtain help in QT.

- *hlp* lists the names and *H1* comment lines in all M-files in the folder *BGB_MATLAB_Scripts*. The list of QT commands given in Table 21.10 was created in this way.
- *hlp topic* lists the names and *H1* comment lines of all M-files that contain the character sequence *topic*. For example, typing
  hlp median
  produced the following output

```
cmd.m:% Column median [0/0]
mdf.m:% Median filter based on a window of size u*v [0/2]
```

Table 21.1
Comparing QT operating modes. Features marked with an asterisk (*) can be performed via command line control

Function	Window mode	Stack mode	Terminal mode
Cursor	Limited functionality	Yes	Yes
Interactive thresholding and gamma adjustment	Yes	Yes	Yes
Reading standard images	Buttons or command line	Command *rti* loads nine popular images into image stack. User then clicks on thumbnail to read image. Can also read images using command line	
Read images	Button* (single image store available)	CLICK on thumbnail, or via command line (nine thumbnails are displayed in the image stack)	
Save images		CNTRL/CLICK on thumbnail* (storage for eight images available)	
Extra windows (e.g., 3D plots and multi-image display)	No	Yes	Yes
Commands (single)	Buttons or command line	Command line only	Command line only
Command sequences	No (alphanumeric results not displayed)	Yes	Yes
Keyboard short cuts	No	Yes	No
Automatic saving of images during interaction (*push* onto stack)	No	Yes	No (stack can be operated via command line)
Sound playback	Yes	Yes	Yes
Error message reporting	Limited	Limited	Full
MATLAB History Window	Present but not visible	Yes	Yes
On-line help and documentation	No	Yes	Yes
On-line review of standard images	No	Yes	Yes
Editing M-files	No	No	Yes
MATLAB debug tools	No	No	Yes

MEDIAN Median value.
MEDFILT2 Perform 2-D median filtering.
MEDFILT1 One dimensional median filter.
NANMEDIAN NaN protected median value.
SIGNRANK Wilcoxon signed rank test of equality of medians.

The first two lines here refer to QT functions and the remainder to standard MATLAB M-files. Notice that the numbers within square brackets indicate the number of outputs and inputs for that function. So, *mdf* has no outputs and two inputs.

Updating help When writing a new QT function, insert a comment line immediately after the declaration of the function name. MATLAB refers to this as the *H1* line. When the user types *hlp*, the *H1* lines of all the M-files in the folder *BGB_MATLAB_scripts* are listed.

- *doc qt_ function* downloads and displays the PDF file containing the documentation for the QT function *qt_function*. A complete set of QT documentation files is provided on the CD provided with this Handbook and can also be found by visiting the author's web site:
- http://www.springer.com
- *img* displays a web page containing thumb-nail versions of each of QT's standard images. When the user clicks on one of these thumb-nails, the image is displayed at full resolution. This facility is useful when loading images with the QT command *rni*, which requires a number

21.6.3 Other Special Functions

QT also provides a series of other information and control functions

- *author* Details about the author of QT
- *dab* Do specified command for all blobs in a binary image
- *doc qtfunction* Read the documentation file describing QT's M-function called *qtfunction*
- *dtx* Toggle the text-display switch
- *esr* Evaluate a specified command sequence, save results for display via WWW
- *hub* Visit the author's main web page. This includes a lot of information about QT
- *img* View thumbnails of the standard images provided with QT
- *lvm* Illustrated catalogue of lighting-viewing methods (see ➔ Chaps. 8 and ➔ 40)
- *mmb* Operate the MMB interface unit
- *news* Latest news about QT updates, errors reported, etc.
- *prolog* Ask SWI-Prolog to satisfy a given goal returning a list of integers to QT
- *qt* Start the QT interactive image processing environment. (➔ *Fig. 21.1*)
- *qto* Go to the QT on-line web site. (Run QT on-line within QT stand-alone)
- *qtp* Add/remove path to QT update folder. Enable/disable updates
- *qtr* Evaluate remote command file and save results for display via world-wide web
- *qtu* Read a remote file and save it as an M-file
- *startup* Executed automatically when QT software is loaded

21.7 Applications

During a period of more than 30 years, QT and its precursors have been applied to a wide range of industrial vision tasks. A chapter such as this can do little more that discuss a few representative examples. In doing so, we will inevitably lose one essential feature of QT, the dynamics of the human-computer dialogue. One type of question can never be answered properly in a static paper-based description of QT: "*How did solution XXX emerge?*" The user of an interactive image processing system like QT can never explain how new algorithms were invented in response to a new inspection requirement. The reader is strongly urged to.

21.7.1 Analysing the Shape of a Glass Bottle

Inspection task: Examine the silhouette of a bottle, checking it for symmetry, radius of curvature of the neck-shoulder region, vertical posture (when standing on a horizontal surface), parallel sides. The physical strength of a bottle is determined in large part by the curvature of the neck-shoulder region.

Related tasks: Inspection of symmetrical components, made of glass, plastic, metal, wood, rubber, etc.

Lighting: Back illumination.

Sample image: ➋ *Figure 21.3* shows the image of a glass milk-bottle, with embossing on the front and rear faces. There is a crack in the neck and some surface imperfections which create dark smudge-like patches.

Measuring radius of curvature: To begin, three edge points on the neck are found, by scanning the binary image of the bottle from left to right, along the following rows, which were identified manually:

$$y = 49$$
$$y = 49 + 30 = 79$$
$$y = 49 + 30 + 30 = 109$$

The following command sequence converts this image to binary form. The function *fce* then locates the left-most white point on each of these rows and computes the circle parameters, *[x,y,r]:*

```
enc                              % Enhance contrast
thr(0,127)                       % Threshold
[x,y,r] = fce(49,30,30)          % Fit circle to three edge points
```

In a similar way, a circle can be fitted to the shoulder, simply by selecting three lower scan lines. These two circles are shown in ➋ *Figs. 21.4* and ➋ *21.5*.

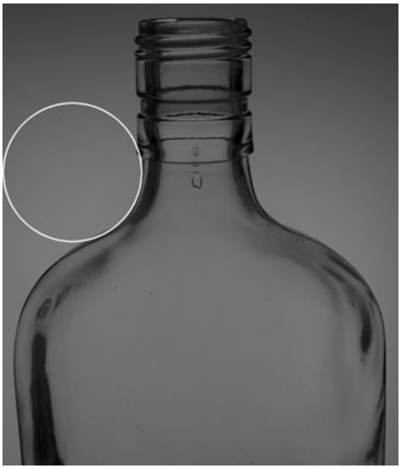


Fig. 21.4
***Circle* fitted to the neck of the bottle**

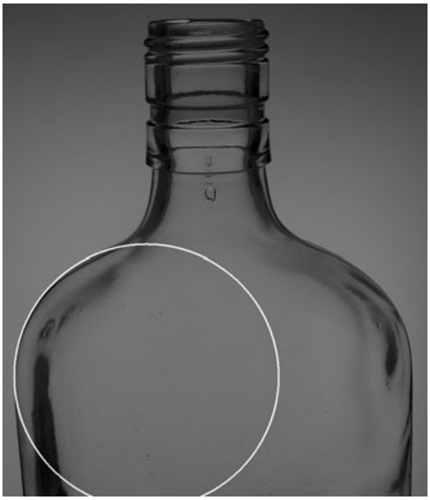


Fig. 21.5
***Circle* fitted to the neck of the bottle**

Fitting a polynomial curve: The following function fits a polynomial of order *n* to a set of edge points, found using *hzs*.

```
function p = fit_polynomial(y1,y2, delta_y, n)
% Fit a polynomial of order n (n ≥ 1) to a binary edge [1/4]
global current
global alternate
original = current;                    % Save image for later
[x,y] = hzs(y1,y2, delta_y);           % Pivotal points
p = polyfit(y,x,n);                    % Calculate polynomial parameters
q = polyval(p,y);                      % Evaluate polynomial
Zer                                    % Black image
[u,v] = size(y);                       % How many pivotal points?
for i = 2:v
vpl(round(q(i)),round(y(i)), round(q(i-1)),round(y(i-1)),255)
                                       % Piece-wise linear plot
end
alternate = original;                  % Recover original image
subplot(1,2,1), imshow(current)        % Draw Current Image
subplot(1,2,2), imshow(alternate)      % Draw Alternate Image
```

The result of applying *fit_polynomial* to the image of the bottle is shown in ➋ *Fig. 21.6*.

21.7.2 Calibrating a Spirit Level

21.7.2.1 Spirit Levels

A spirit level (or bubble) level, is a device that indicates whether a surface is truly horizontal, or vertical. Spirit levels are commonly used by building-trade workers and are often incorporated

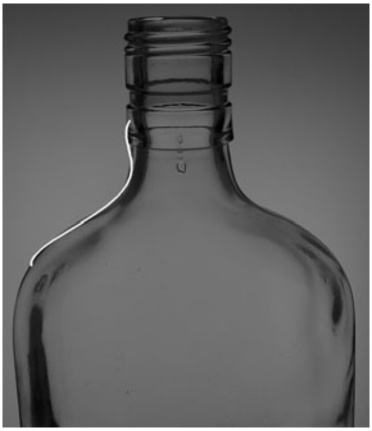


Fig. 21.6

A fourth-order polynomial fitted to the neck–shoulder region, by *fit_polynomial*. The original bottle image was superimposed to emphasise the good fit. Fitting a higher-order polynomial does not improve the fit

into surveying instruments and camera tripods. A common form consists of a slightly curved glass tube that is nearly, but not completely, filled with spirit or alcohol, creating a single bubble, which appears as an ellipse when viewed from above. The ends of the bubble coincide with two calibration lines when the instrument rests on a perfectly level surface. When the device is inclined slightly, the bubble moves away from the centre position. Other forms of spirit levels exist but, for this demonstration of QT, we shall limit ourselves to considering only tube spirit levels.

To make viewing easier, fluorescene is often added to the liquid, making it yellow-green. Preliminary experiments showed that colour filtering (➲ Chap. 16) can assist in separating the bubble from the background. Front lighting, with a localised light source, produces bright highlights by reflection from the tube. This may cause local optical overload in the camera and will certainly complicate the image processing. The bubble acts as an ellipsoidal “hole” in the water. The bubble therefore acts as a crude lens and also creates highlights by reflection. Back illumination produces high contrast images that do not require colour analysis.

21.7.2.2 Locating the Bubble and Calibration Lines

➲ *Figure 21.7* shows a sample grey-scale image, obtained using back-lighting. The back-lit image serves our present purpose, which is to demonstrate the use of QT, rather than offering a solution to a pressing industrial problem. The task we set ourselves is to locate the bubble and calibration lines (A, B, C, D in ➲ *Fig. 21.7*). The “ghost” is the image of calibration line B from the rear of the glass tube. Its displacement is due to the lensing effect of the bubble. ➲ *Figure 21.7* was obtained using a camera placed well away from the spirit level. As a result, the calibration marks appear as single lines, unless the bubble produces a ghost. If a wide angle lens were to be used on a camera located close to the tube, lines A, C and D would split. A telecentric lens would avoid this and allow the camera to be placed closer to the spirit level.

A useful first step when confronted with a new image is to examine its intensity profile along relevant scan lines (➲ *Fig. 21.8*). QT command *plt* does this rapidly but the profile trace is

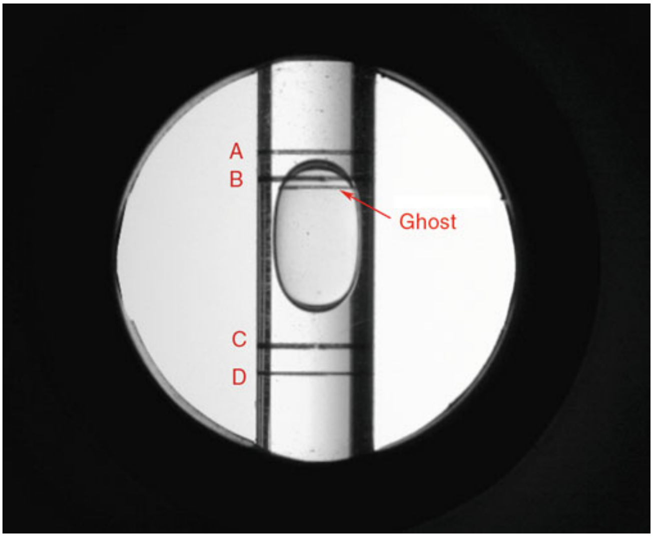


Fig. 21.7
Spirit level, viewed with back illumination. Lines A, B, C and D are genuine calibration marks, while the line labelled "ghost" is an artifact caused by the lensing effect of the bubble. It is an image of line B at the rear of the glass tube. Text was added using Photoshop™

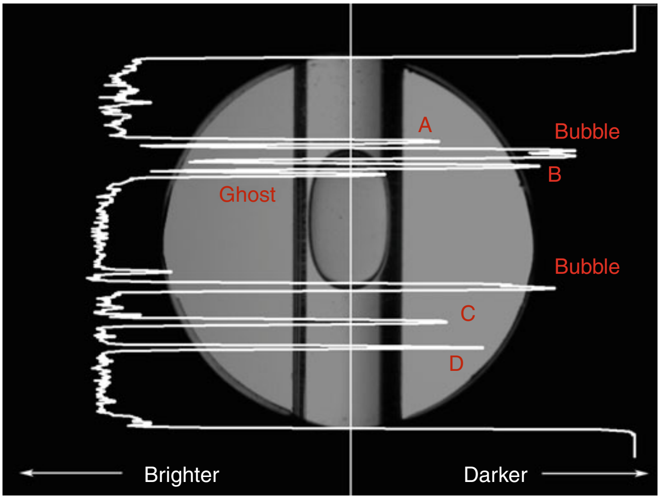


Fig. 21.8
Intensity profile along a vertical line through the centre of the bubble. Notice that the intensity of the ghost is about mid-way between the intensities of the bright back-ground and the dark edge of the bubble and the calibration lines. Text was added using Photoshop™.
[QT: *psh(680,0), csh, wgx, sub, thr, bed, vpl(680,1,680,960,255, lrt, dil(2), rei(original), swi, adi, mxi*]

not permanent; it is not possible to process it as an image. An alternative is to use the following QT command sequence. The parameter *p* controls the position of the vertical scan line.

```
[w,h] = dgw;     % Width height of image
psh(w-p,0)       % Shift right so desired scan line is on RHS of image
csh              % Make all columns same as RHS
wgx              % Intensity wedge
sub              % Subtract Current & Alternate Images
thr              % Threshold at mid grey (level 128)
bed              % Binary edge
dil(2)           % Optional: dilates the trace slightly to improve clarity
```

The resulting image is the same size as the original and hence can be combined with it, as was done in ❯ *Fig. 21.8*.

In almost applications studies, another useful preliminary step is to draw the intensity histogram. (❯ *Fig. 21.9*). There are two strong peaks, corresponding to the bright circular disc and the dark surround but the calibration marks and bubble contribute very few pixels whose contribution to the histogram is not obvious.

As we shall see a little later, it is advantageous to separate the bright central disc from the very dark surrounding area For this reason, the histogram happens to be useful, after all. The centre of the "valley" in a bimodal histogram is often a useful indicator for the threshold parameter needed to separate a figure from its background. An easy way to do this is to use the following QT command sequence:

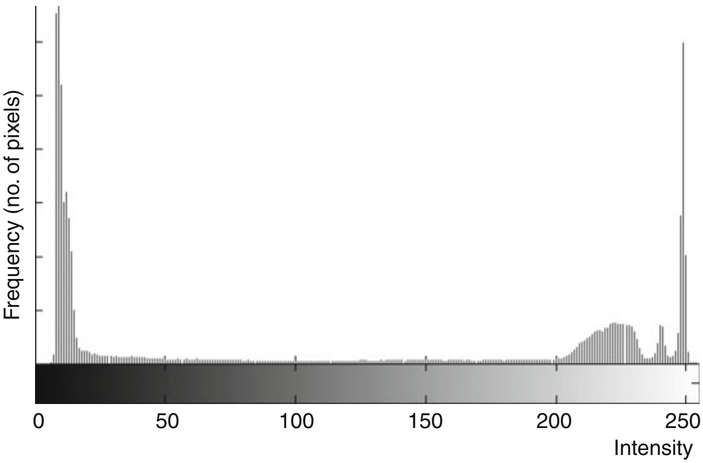


Fig. 21.9
Intensity histogram for the image in ❯ *Fig. 21.7*. Notice that there are two distinct peaks, corresponding to the bright central disc and the very dark background. A bimodal histogram, such as this, often indicates that it is reasonable to apply thresholding to separate a figure from its background. However, in this application, the features of interest make only a very small contribution to the form of the histogram [QT: *hpi*. Axes were relabelled using Photoshop™]

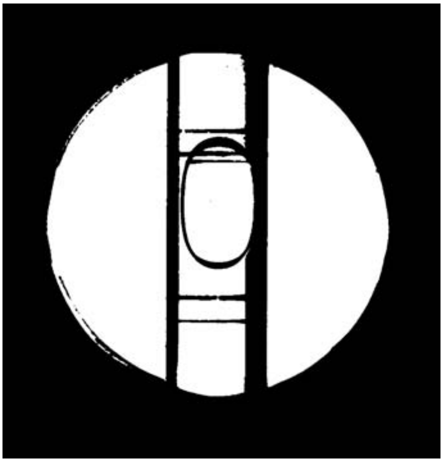


Fig. 21.10
Thesholding applied to the image in *Fig. 21.7*. *[QT: enc, thr]*

```
enc            % Enhance contrast. Darkest pixel becomes black. Brightest
                 becomes white
thr            % Theshold at mid-grey
```

This effectively thesholds at a level corresponding to the mid-point between the brightest and darkest pixels. (See *Fig. 21.10*)

It is now a simple matter to define a Region of Interest (ROI) All further operations will be performed on pixels within the ROI, while those outside will be ignored. This has two advantages. Firstly, it reduces the computation load and thereby improve the execution time. (*Fig. 21.11*(a)) Secondly, unimportant features outside the ROI will be ignored and hence cannot confuse an analysis program, by distracting it from the important parts of the image. The interesting parts of the spirit level image occupy only a very small part of the original image. Hence, it is desirable to make the ROI even smaller, by using the dark vertical stripes to define its left and right limits. (*Fig. 21.11*(b)) For obvious reasons, a spirit level is made to close tolerance. Since the glass tube is held in a well-defined position relative to the frame, it is not unreasonable to expect that the tube can be placed accurately relative to the camera during calibration. This means that it is likely that we can define a small ROI by "dead reckoning". Another ploy is to use well-defined image features in pairs, to locate a ROI whose limits lie between them. In *Fig. 21.11*, the top and bottom limits of the bright disc were used to calculate the height of a ROI that is much smaller than the gap between them. In this case, the processing load is reduced to less than 8% of that originally envisaged. Of course, such a saving is only worthwhile if the processing necessary to calculate the ROI position is straightforward.

From this point on, we shall process only ROI images. It happens that simple thresholding produces a binary image that can then be analysed in a satisfactory manner using binary morphology. To make the thresholding more tolerant of lighting variations, we again use the QT sequence *[enc, thr]*. Binary erosion with vertical line structuring element then detects the calibration lines, top and bottom of the bubble. (*Fig. 21.12*) Here is the command sequence used so far (excluding ROI selection)

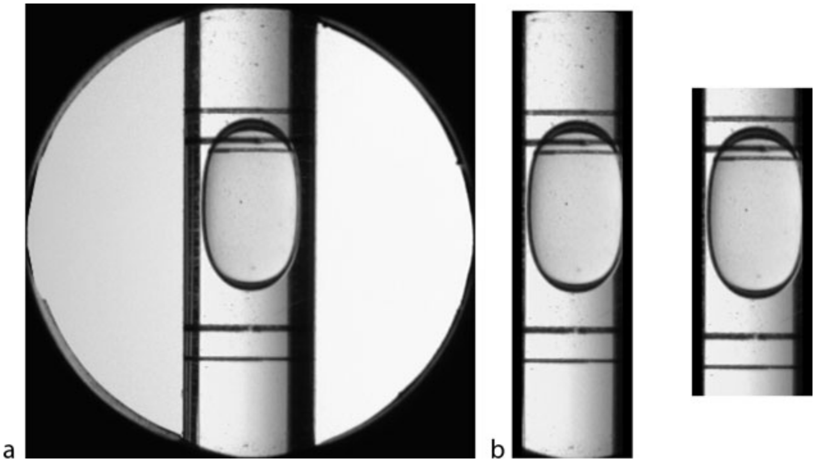


Fig. 21.11

Region-of-interest (ROI). (**a**) ROI defined by the minimum-area box bounding the central bright disc. The computational load for point-by-point calculations (e.g. negation, thresholding and convolution filtering) is reduced to 43.5%. [QT: *wri, enc, thr, [a,b,c,d] = exm(0), rni(126), crp(a,c,b-a, d-c)*] (**b**) The left and right edges of the ROI are defined by those of the vertical dark bands. The computational load is reduced to 11.5%. [QT: *wri, enc, thr*, *big(2), x = cwp; swi, kgr(x), exr, kgr(100), [a,b,c,d] = exm(0), rei, crp(a,c,b-a,d-c), pad(16,0,0)*] (**c**) Minimal ROI, allowing for some uncertainty in the vertical position. The computational load is reduced to 7.9%

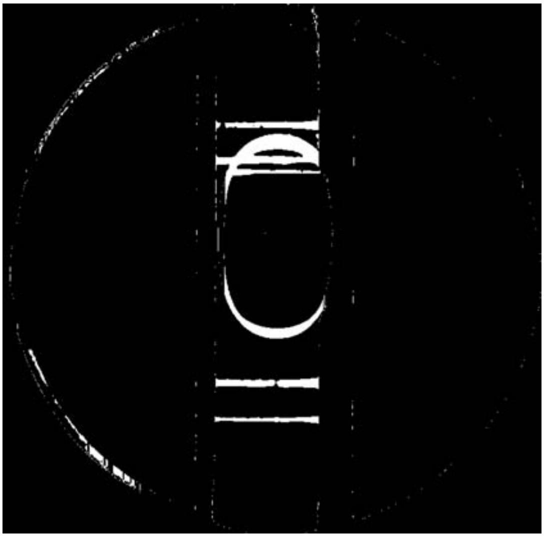


Fig. 21.12

Thresholding (See ➋ *Fig. 21.15b*) and binary erosion with a vertical line structuring element

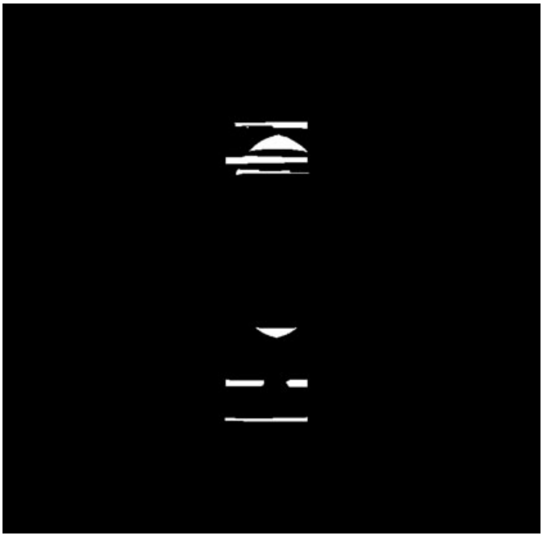


Fig. 21.13
Binary erosion applied to *Fig. 21.12*. The structuring element was a horizontal line, 30 pixels long

```
enc                % Enhance contrast
thr                % Threshold at mid-grey
x = cse(2,50)      % Construct a vertical line structuring element, x, 1*50
                     pixels
gbd(x)             % Dilation. This following operator together implement
                     closing with SE x
gbe(x)             % Erosion. This preceedingoperator together implement
                     closing
rei(original)      % Recover original ROI image
exr                % Exclusive OR of Current and Alternate Images
```

Notice that the top edge of the bubble produces a thick white arc that overlaps calibration line B and the ghost. Horizontal features in this image can be detected using binary erosion with a horizontal line structuring element. (Chap. 19) Here is the command sequence used to generate *Fig. 21.13*:

```
y = cse(1,30)     % Construct a horizontal line structuring element, y,
                    30*1 pixels
gbe(y)            % Erode with structuring element y
```

The y-coordinates of the centroids of these blobs provides the numeric result that we need to complete the calibration. Intepreting this data is an issue that need not bother us here. Let it suffice to say that some intelligent reasoning is needed, since the bubble moves unpredictably. Instead, we shall show visually that the results appear to be reasonable, compared to what we expect. Instead of using the centroid, we use an approximation based on the thinning operator, *thn*.

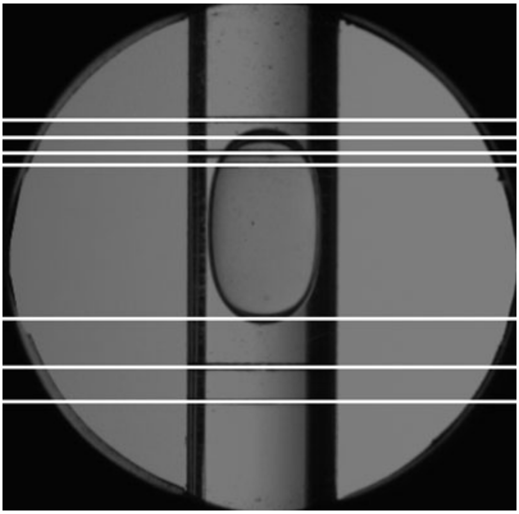


Fig. 21.14

Locating features of interest on the spirit level. Clearly, the bubble can move to overlap one or both of the lower pair of calibration lines (C and D), or not be visible at all. As the bubble moves, the ghost will also move, or not be present at all

```
rox               % Row maximum
csh               % Make all columns the same as the RHS (produces horizontal
                    stripes)
thn               % Thin the stripes to be horizontal lines one pixel wide
dil(2)            % Dilate slightly, to make the resulting lines more obvious
rni(original)     % Read the original ROI image
swi               % Swtch images
adi               % Add Current and Alternate Images
mxi               % Point-by-point maximum intensity between Current
                    Alternate Images
```

The result is shown in *Fig. 21.14*

21.7.2.3 Tracing the edge of the bubble

Finally, we explain how the edge of the bubble can be traced. (*Fig. 21.15*) While this task is not strictly part of the calibration process, it does show that QT can implement complex scene analysis procedures. Since the following QT command sequence was developed interactively, it is not possible to justify why every processing step was chosen. As with many other manifestations of human intuition, rationalisation fails to explain why the mystery remains. An important point worth noting is that the sequence listed below has been shown to produce the desired result on a single image; it has not yet been tested and refined using a large set of images, derived from a range of spirit levels.

```
crp(305,105,731,721)   % Crop parameters were chosen as described earlier
wri('a')               % Save image - see Figure 21.15(a)
```

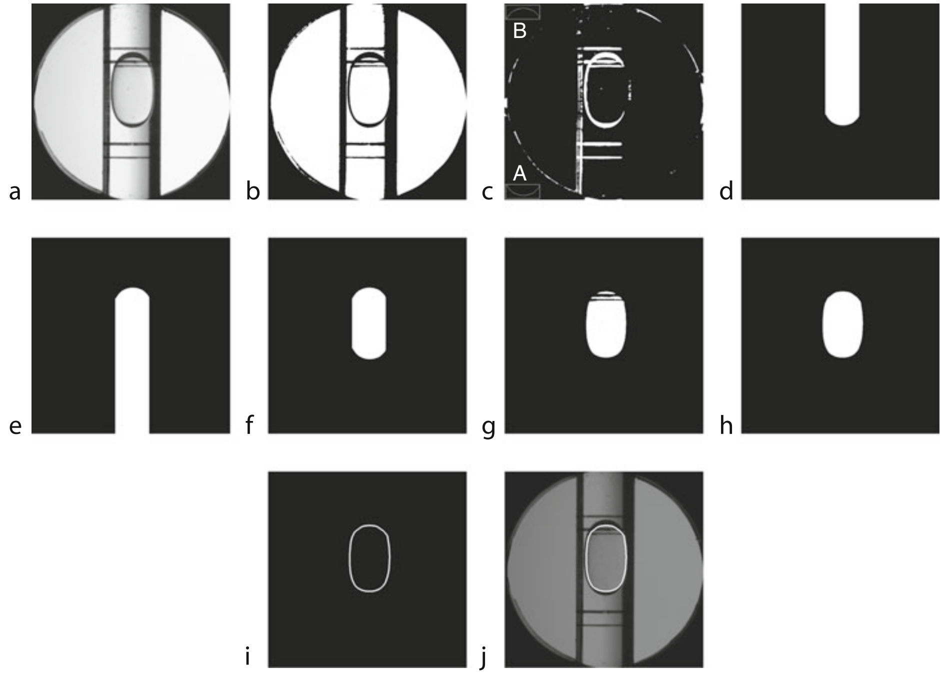


Fig. 21.15

Tracing the edge of the bubble. (a) Original image after cropping. *[crp]* (b) Contrast enhancement and thresholding. *[enc, thr]* (c) Crack detection and thresholding applied to *(b)*. Inserts *A* and *B* are structuring elements. (d) Structuring element A was used to erode image *(c)*. (Structuring element A is represented by *y* in the QT program.) This resulted in single very small blob, which was then shrunk to a single point *[shk]* and dilated [*gbd(y)*], using structuring element *A*. This creates a white arc identical to the structuring element *A*. Following this, a "shadow" (upwards from this arc) was created. *[tbt, cox, tbt]* (e) Similar processing to (d) but using structuring element *B* and creating a "shadow" going downwards. [cox] (f) Images (d) and (e) ANDed together. *[mni]* (g) Blobs in *(b)* that overlap the blob in *(f)*. *[kob]* (h) The previous image was closed. (i.e. dilation followed by erosion) The convex hull *[chu]* was drawn around the resulting blob. (i) Edge of the previous image. *[bed]* (j) Edge contour superimposed on the original image. *[rei('a'), swi, adi, mxi]*

```
enc                     % Enhance contrast
thr                     % Threshold at mid-grey
wri('b')                % Save image - see Figure 21.15(b)
rei('a')                % Save ROI image - see Fig. 21.15
crk(11)                 % Crack detector
enc                     % Enhance contrast
thr(32)                 % Theshold. Parameter chosen interactively
wri('c')                % Save image - see Figure 21.15(c)
gbe(y)                  % Erosion. Structuring element (y) is shown in
                          Fig. 21.15(c)
shk                     % Shrink each blob to a single point
gbd(y)                  % Dilation. Structuring element (y) is shown in
                          Fig. 21.15(c)
```

```
tbt                 % Invert vertical axis
cox                 % Column maximum
tbt                 % Invert vertical axis
wri('d')            % Save image - see Figure 21.15(d)
rei('c')            % Recover image saved earlier (Fig. 21.15(c))
tbt                 % Invert vertical axis
gbe(y)              % Erosion. Structuring element (y) is shown in
                      Fig. 21.15(c)
shk                 % Shrink each blob to a single point
gbd(y)              % Dilation. Structuring element (y) is shown in
                      Fig. 21.15(c)
tbt                 % Invert vertical axis
cox                 % Column maximum
wri('e')            % Save image - see Figure 21.15(e)
rei('d')            % Recover image saved earlier (Fig. 21.15(d))
mni                 % Point-by-point minimum int'y in Current
                      Alternate Images
wri('f')            % Save image - see Figure 21.15(f)
kgr(1000)           % Ignore small blobs. . Parameter chosen
                      interactively
rei('b')            % Recover image saved earlier (Fig. 21.15(b))
kob                 % Keep blobs in Current Image that overlap blob in
                      Alternate Image
wri('g')            % Save image - see Figure 21.15(g)
dil(5)              % Dilate, to join blobs
chu                 % Convex hull
ero(5)              % Erode. Restore blob to original size
wri('h')            % Save image - see Figure 21.15(h)
bed                 % Binary edge
dil(2)              % Dilate. Make edge contour more obvious for
                      display purposes
wri('i')            % Save image - see Figure 21.15(i)
rei('a')            % Recover oriiginal ROI image (Fig. 21.15(a))
swi                 % Switch Current and Alternate Images
adi                 % Add Current and Alternate Images
mxi                 % Point-by-point maximum int'y in Current
                      Alternate Images
wri('j')            % Save image - see Figure 21.15(j)
```

21.7.3 Robot Vision

In the limited space available, it is only possible to introduce a few of the important issues encountered when designing robot vision systems. Hence, our aim in this section is simply to show that QT posseses appropriate tools for this type of task. We begin by considering one of the simplest: locating and picking up rigid objects lying in random position and orientation on a flat featureless table. An overhead camera views the object. *Figure 21.16a* shows a typical scene: a set of objects lying on an $[X, Y, \theta]$-table. To simplify matters, we shall assume that each of the objects presents a high visual contrast, when viewed against the table surface. This enables the digital image to be segmented easily, perhaps using a simple QT command sequence such as

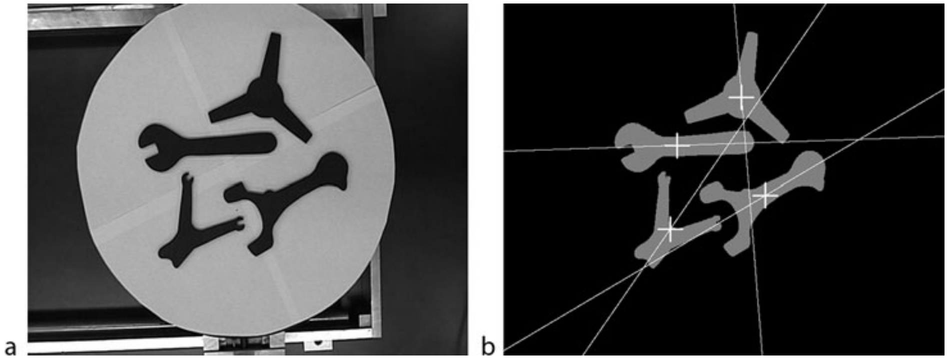


Fig. 21.16
Simple robot vision scene. A common requirement is to isolate each object in turn, then calculate its position and orientation coordinates. These parameters are then translated into robot coordinates. The robot will pick up each object in turn, then place it in a fixed position and orientation, perhaps for packing, assembly, or machining. (a) *Original image*: four objects lying on an [*X,Y,θ*]-table, as seen by an overhead camera. (b) Each object has been isolated, its centroid (cross) and principal axis (*white line*) have been calculated. Notice that the orientation of the principal axis for the three-pointed star is not a reliable indicator of its orientation but is for the other three objects. The centroid of the Y-shaped object is not a good place to position a suction or magnetic gripper, as it lies too close to the object edge to ensure secure lifting

[enc, thr]. For now, we shall assume that a binary image is available and that any background detail, such as the frame of the [X,Y,θ]-table, has been eliminated. Initially, we shall assume that there is only one object in the camera's field of view, although we shall remove this restriction later.

For the sake of brevity, we shall ignore the essential task of calibrating a robot vision system (often referred to as *hand-eye coordination*). Few general principles can be given, since the process depends on the type of robot and gripper, how the system is used, where and how the camera is set up. Let it suffice to say that, by observing the robot as it performs a set of pre-defined movements, QT can calculate the transformation matrices needed to convert the vision system coordinates into the robot coordinates, and vice versa [3]. Calibration is almost always necessary, particularly when the camera and robot are set up independently of each other. Normally, neither "knows" exactly where the other is. Since, the camera's optical magnification (zoom) and focus are best adjusted interactively when it is *in situ*, prior calculations, based upon exact design drawings of the robot-camera cell, are unlikely to provide the necessary accuracy.

21.7.3.1 Picking up Isolated Rigid Objects

Locating most objects is performed easily using the centroid. (➋ *Fig. 21.16*) In QT,

$$[x, y] = cgr$$

calculates the centroid position, [x,y]. However, this may not be best place to put the robot gripper, which we shall, for the moment, ssume is of suction type. A U-shaped object cannot be

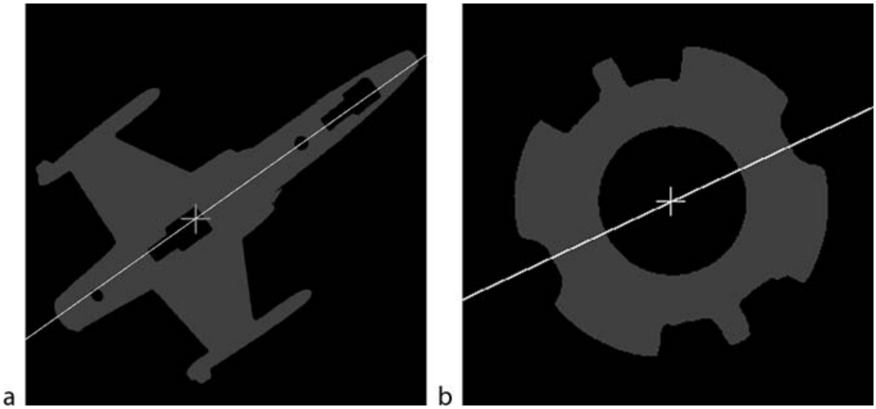


Fig. 21.17

Positioning the gripper when the centroid does not lie within the object bondary. (a) Casting for a model aeroplane. The centroid lies within a lake and is therefore not a good place to put a suction or magnetic gripper. However, placing the gripper at some point along the principal axis, part-way towards the second largest lake will ensure secure lifting. (b) Cam. Although the centroid lies within a lake, this is an ideal place to put a three-finger gripper that expands to grasp the component

lifted at the centroid, since it lies outside the object boundary. The metal casting in *Fig. 21.17a* clearly cannot be lifted by placing the gripper at its centroid. *Figure 21.17b* emphasises the point that the type of gripper is important when calculating grasping positions. The cam cannot be lifted by a suction, or magnetic, gripper placed at the centroid, although it could be held securely by a self-centering 3-finger *expanding* gripper. (Like a drill chuck except that it grips when the fingers are separated.) However for the moment, we shall continue to use a suction gripper. If it is not sensible to place the gripper at the centroid, it may be reasonable to place it at some point defined with reference to the centroid and some measurement of the orientation. A simple rule might be applied in the case of the Y-shaped object in *Fig. 21.16d* or the casting in *Fig. 21.17a*:

- Starting at the centroid, move Q millimetres/pixels along the principal axis, travelling towards/away from the centroid of the largest bay.

Finding the orientation is even more problematical. The principal axis works well for the conrod, spanner (wrench) and even the Y-shaped object in *Fig. 21.16*. It is also satisfactory for the cam, despite there being no obvious "long axis" (*Fig. 21.17b*). However, it does not produce a useful measure of orientation for certain objects, so other methods are sometimes needed. One possibility is to use a line joining two bays, or two lakes. For example, the orientation of a pair of scissors can be derived from the line joining the centroids of the finger-holes (*Fig. 21.18a*). Here is the command sequence for this calculation:

```
blf                             % Fill lakes
exr                             % Exclusive OR – isolates the lakes
big(1)                          % Choose the largest lake
[x1,y1] = cgr;                  % Centroid of the largest lake
```

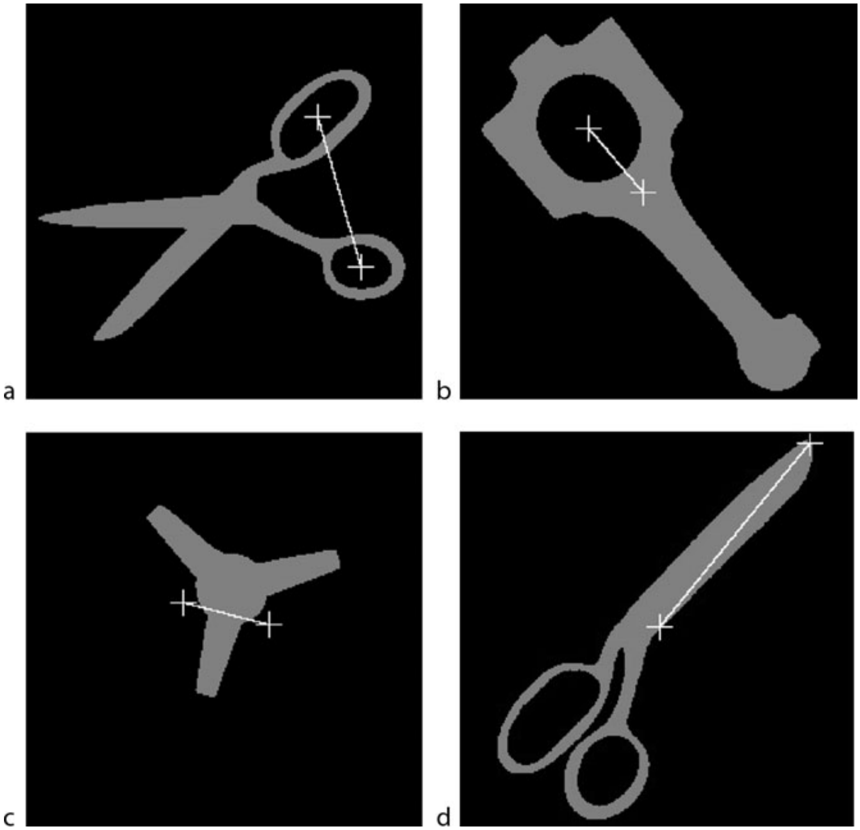


Fig. 21.18

Alternative methods for finding orientation. (a) Scissors (*open*). The lakes (*finger holes*) provide a robust way to determine the position, orientation and the angle of opening. (b) Unfinished forging for an automobile conrod. The centroid and the centroid of the lake determine the orientation. (c) Metal stamping (*three pointed star*). The centroids of two bays determine the orientation. (d) Scissors (*closed*). The centroid and the furthest edge point from the centroid provide a method for determining the orientation

```
swi                              % Switch images
big(2)                           % Choose the second largest lake
[x2,y2] = cgr;                   % Centroid of the largest lake
z = atan2((y1-y2),(x1-x2))       % Orientation (radians)
```

This method work well, whether the scissors are open wide or fully closed.

The following sequence finds the orientation of an object using the line joining its centroid and the centroid of its Nth largest lake (➋ *Fig. 21.18b*, $N = 1$).

```
[x1,y1] = cgr;                   % Centroid of the object
blf                              % Fill lakes
exr                              % Exclusive OR - isolates the lakes
big(N)                           % Select the N-th largest lake
```

```
[x2,y2] = cgr;                    % Centroid of the N-th largest lake
z = atan2((y1-y2),(x1-x2))        % Orientation (radians)
```

Sometimes we might use two bays instead (➲ *Fig. 21.18c*).

```
[x1,y1] = cgr;                    % Centroid of the object
blf                               % Fill the lakes first
chu                               % Convex hull
exr                               % Exclusive OR – isolates the bays
big(N)                            % Select the N-th largest bay
[x2,y2] = cgr;                    % Centroid of the N-th largest bay
z = atan2((y1-y2),(x1-x2))        % Orientation (radians)
```

Another possibility is to find the edge point that is furthest from the centroid. This method for finding orientation is appropriate when the object has a single, long, sharply pointed "limb" (➲ *Fig. 21.18d*).

21.7.3.2 Finding Orientation by Pattern Matching

The methods described above do not produce accurate results when applied to convex shapes, or those that have only small bays, no lakes and no well-defined principal axis. For objects like this, a completely different approach is necessary. The operator *fbl*, defined below, is based on the following procedure (see ➲ *Fig. 21.19*):

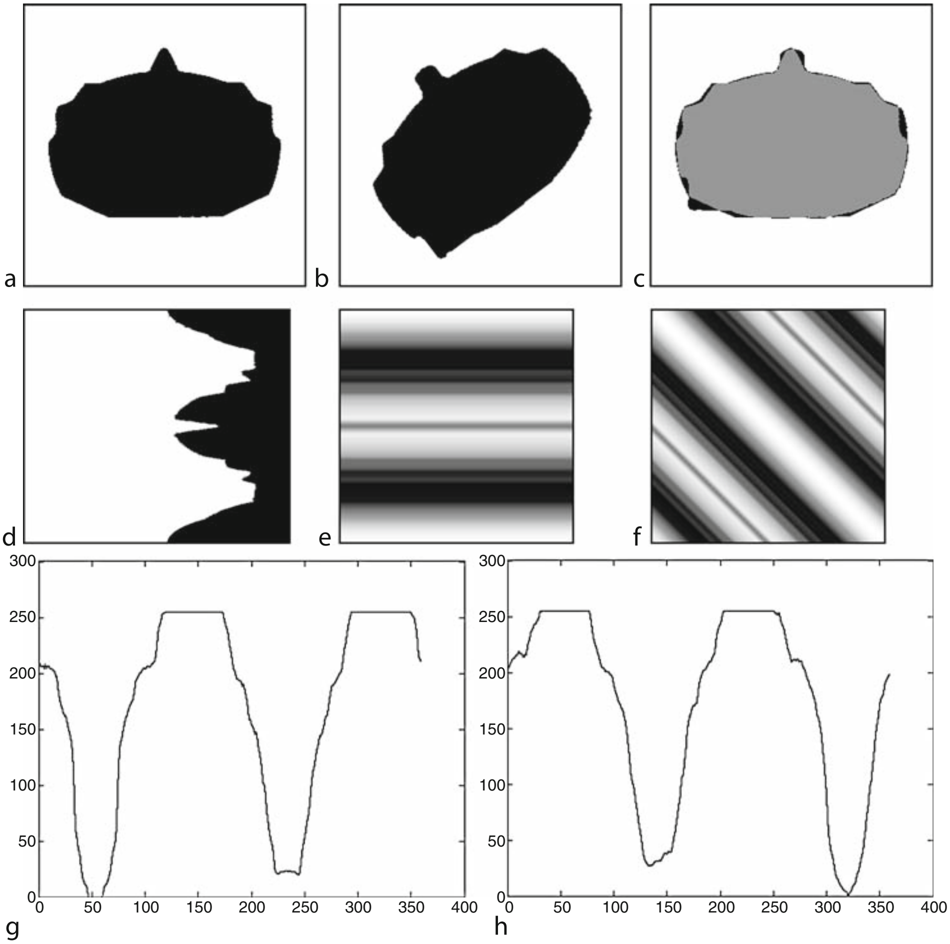
1. Analyse the Current Image and calculate the distribution of "mass" along a series of 360 lines radiating from the object centroid. This is achieved using the operator sequence *[ptc,rin]*. This generates a 360-element, 1-dimensional array, *x*, describing the object shape. (The shape cannot be reconstructed exactly from array *x*.)
2. Contruct an image containing 360 copies of *x*. The operator sequence *csh, crp(1,1,360,360)* does this (➲ *Fig. 21.19e*). Call this image *X*.
3. Repeat step (i) for the Alternate Image, thereby generating another 360-element, 1-dimensional array, *y*.
4. Generate an image in which column *i* is a copy of *y* that has been subjected to a circular shift of *i* pixels (see ➲ *Fig. 21.19f*). Call this image *Y*.
5. Find which column of image *Y* most accurately fits the pattern stored in each column of *X*. This implements a simple pattern matching algorithm based on minimising the quantity.

$$D(i) = \sum_{j=1}^{\text{height}} |(X(i,j) - Y(i,j)|$$

The summation is performed along the columns of the image. *D(i)* is called the *City Block Distance* between two column vectors, one representing the test object (the original Current Image) and the other a reference object (Alternate Image), after it has been rotated by *i* degrees. Hence, *n* where

$$D(n) \leq D(i) \quad \text{for} \quad i = 1, 2, \ldots, 360$$

indicates how far the object in the Current Image has to be rotated to match the reference object best.



◘ **Fig. 21.19**

Finding orientation by matching a vector describing the radial distribution of mass to a set of similar vectors derived from a standard object in different rotations. (a) ***Binary image*****: metal base-plate for an automobile brake pad. Call this** ***Part A*****. (b) Binary image from another slightly different object,** ***Part B*****. (c) After applying** ***[rei('B'), rei('A'), fbl]*****, Part B has been rotated and shifted so that it ''fits'' Part A. Black areas indicate the differences between the two objects. (d) [*****ptc*****] applied to part A. (e) [*****ptc,neg,rin*****] applied to part A. (f) Result of the shear warping. (This is image** ***Y*** **mentioned in the text.) (g) Array of distance scores, [*****D(1), D(2),...*****], when Part A is compared to another image of Part A rotated by 55°. Notice that the minimum value of D_i occurs when i = 55. (h) Array of distance scores, [*****D(1), D(2),...*****], when Part A is compared to an image of Part B. The minimum value of D_i (*****i*** **=321°) indicates how far the image of Part B must be rotated so that it fits Part A. When Part B is rotated by this amount, it fits well onto Part A (➋** ***Fig. 21.19c*****)**

Here is the listing for *fbl*:

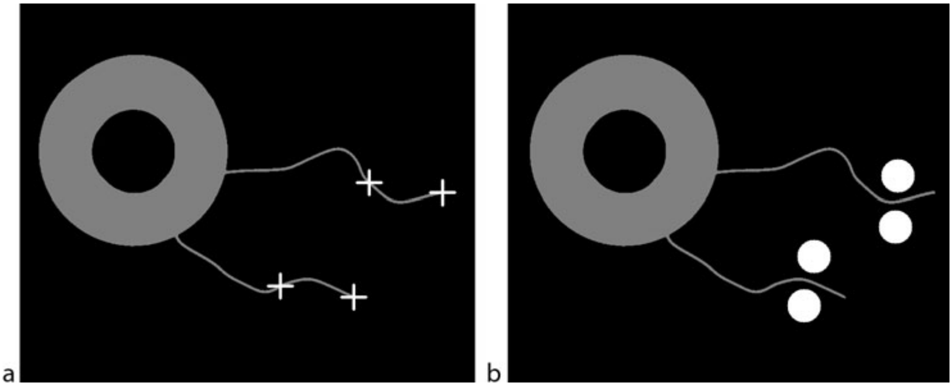
```
function n = fbl
% Fit one blob onto another [1/0]
global current
global alternate
cur_image =           % See ➔ Fig. 21.19b
current;
alt_image =           % See ➔ Fig. 21.19a
alternate;
nxy;                  % Normalise the position of the centroid (Current Image)
cur_image =           % Save image for later
current;
ptc;                  % Polar-to-Cartesian mapping: see ➔ Fig.
                        21.19d
neg;                  % Negate
rin;                  % Row integration
csh                   % Make all columns same as RHS; see ➔ Fig.
                        21.19e
crp(1,1,360,360)      % Crop image to 360*360 pixels
temp = current;       % Save image for later. This is image
                        X mentioned above
current =             % Now we are going to analyse the Alternate
alt_image;              Image
nxy;                  % Normalise the position of the centroid
alt_image =           % Save image for later
current;
ptc;                  % Polar-to-Cartesian mapping: similar picture to
                      ➔ Fig. 21.19d
neg;                  % Negate
rin;                  % Row integration
csh                   % Similar picture to ➔ Fig. 21.19e
for i = 1:360;
  y = circshift       % Circular shift each column by different amount
     (current
     (:,i),i);
  temp2(:,i) = y;
end                   % For result see ➔ Fig. 21.19f
current = temp2;      % Keep result for further processing.
                        Image Y mentioned above
alternate = temp;     % Recover image saved earlier
sub                   % Subtract
abv                   % ''Absolute value'' (''folds'' intensity
                        scale around mid-grey)
heq                   % Histogram equalisation – heuristic that works!
cin                   % Column integration
yxt                   % Interchange the image axes
csh                   % Make all columns same as RHS
yxt                   % Interchange the image axes
w = lsc(1,1,360,1);   % Sample intensities along the line joining [1,1]
                        and [360,1]
                        % w is the array of distance values (D mentioned
                          above.)
➲ Figure 21.5           % Switch drawing to new figure
plot(w)                 % Vector of ``distances'' as object is rotated.
                          See ➲ Fig. 21.19h
➲ Figure 21.1           % Revert to normal figure for drawing images
[a,b] = gli;            % Get limits of intensity
thr(a,a)                % Select pixels at lowest intensity
[n,m] = cgr;            % Find centre band. We ignore vertical position (m)
current =               % Recover normalised original Current Image
  cur_image;
tur(–n)                 % Rotate image by –n degrees; see ➲ Fig. 21.19c
alternate =             % Recover the original Alternate Image
  alt_image;
subplot(1,2,1)          % Getting ready to display an image
imshow(current)         % Now actually displaying the new Current Image
subplot(1,2,2)          % Getting ready to display another image
imshow(alternate)       % Finally, display the new Alternate Image
```

No attempt has been made to optimise this program. To improve the speed of execution, image Y (➲ *Fig. 21.19e*) should be computed once and then stored for future use.

Notice that *fbl* is invariant of position changes for both objects. Scale invariance is also possible by normalising the object size before *fbl* is invoked. Finally, notice that, when *fbl* is executed, the Alternate Image (i.e. the reference image) remains unchanged, while the Current Image is rotated. Hence, *fbl* does not follow the convention established for QT operators.

21.7.3.3 Gripping Flexible Wires

➲ *Figure 21.20* shows the silhouette of a small inductor (coil) with flying leads. Our task is to construct a visual guidance system for a robot that will, in turn, grasp the two free ends of the



■ Fig. 21.20
Grasping flexible wires. (a) Original image, coil with two flying leads. The points marked with crosses indicate the ends of the wires and, on each wire, a point N pixels from its end. N is determined by the size of the gripper. (b) Gripper finger positions in the "open" configuration

wires and then solder them onto connecting pins. This is quite a complicated task, as it requires the calculation of three positional and two angular coordinates for each wire. In practice, the wires can overlap, or even become twisted together. To achieve a complete solution to this problem, we require at least four cameras: one overhead and three providing side views. Moreover, a high level of intelligent image interpretation is needed, particularly to resolve situations where the wires are entwined. For the sake of clarity, we shall consider the simplified task of analysing plan-view images like that shown in *Fig. 21.20a*, considering only two dimensions. This yields three values: *[X,Y]* position coordinates and one angle. We shall explain the processing steps required, by concentrating on just one wire.

Once the image of the coil has been reduced to binary form, a skeletonisation/thinning operator is applied. (*ske* or *thn*) The pixel at the very end of the skeleton limb is then isolated, using the operator *lmb* and its position, $[x_1,y_1]$, is found using *gap*. Next, the skeleton is "pruned", by removing the pixel from the tip of the limb. This process is repeated iteratively, say, *N* times. (*N* is chosen to suit the size of the robot gripper.) The position of the last pixel removed is then found. Call this $[x_2,y_2]$. Here is the code to calculate these values.

```
t2 = current; zer, t1 = current; swi;
for i = 1:N
   current = t2; lmb, alternate = t1; mxi, t1 = current;
   current = t2; lmb, exr,temp2 = current;
end
current = t1; big(1), lmb, [x1,y1] = gap;
current = t1; big(2), lmb, [x2,y2] = gap;
```

The points $[x_1,y_1]$ and $[x_2,y_2]$ enable us to calculate the position coordinates *[x,y]* and the angular coordinate in the XY plane, *theta*, required by the robot:

```
x = (x_1 + x_2)/2;             % [x,y] is approximately N/2 pixels from end of
                                 the wire
y = (y_1 + y_2)/2;
theta = atan2((y_1-y_2)        % Angle of wire at end
  /(x_1-x_2));
```

(Notice that *atan2(.)* is MATLAB's four-quadrant inverse tangent function.)

21.7.3.4 Ensuring Safe Lifting

Consider the scene shown in *Fig. 21.16b*, which shows four objects lying close to each other but not overlapping. Suppose we wish to direct the robot to pick up the conrod. In order to do so, the robot must place its fingers on opposite sides of the object. The robot lowers its gripper onto the work table in the "open" configuration. How can we be sure that the robot will not bump into another object lying nearby. One solution is to superimpose a diagrammatic representation of the robot's "fingerprint" onto the (binary) image. If the number of binary objects is increased by exactly two, it is safe to lower the gripper. If the gripper fingerprint overlaps either the object that is to be picked up, or some other nearby object, the Euler number will not be increased by 2. In this case, it is not safe to lower the gripper, as a crash would occur.

Here is a QT program which can be used to check the safety for a 2-finger gripper used to grasp one of the flying leads of a coil. The input parameters are as follows:

```
function [x,y,z] = grip2(x1,y1,x2,y2,N1,N2)
% [x1,y1]                    Cartesian coordinates of the end of the
                               end of the wire
% [x2,y2]                    Cartesian coordinates of a point N pixels
                               from the end of the wire
% N1                         Length of the gripper finger
% N2                         Width of the finger
global current
global alternate
original = current;
x = (x1 + x2)/2;
y = (y1 + y2)/2;
z = atan2((y1-y2),(x1-x2)) + pi/2;
s = sin(z);
c = cos(z);
u1 = x1 + N1*c;
v1 = y1 + N1*s;
u2 = x1 + (N1 + N2)*c;
v2 = y1 + (N1 + N2)*s;
u3 = x2 + N1*c;
v3 = y2 + N1*s;
u4 = x2 + (N1 + N2)*c;
v4 = y2 + (N1 + N2)*s;
vpl(u1,v1,u2,v2,255)         % Draw a line from [u1,v1] to [u2,v2]
vpl(u3,v3,u4,v4,255);        % Draw a line from [u3,v3] to [u4,v4]
vpl(u3,v3,u1,v1,255);        % Draw a line from [u3,v3] to [u1,v1]
vpl(u4,v4,u2,v2,255);        % Draw a line from [u4,v4] to [u2,v2]
u1 = x1-N1*c;
v1 = y1-N1*s;
u2 = x1-(N1 + N2)*c;
v2 = y1-(N1 + N2)*s;
u3 = x2-N1*c;
v3 = y2-N1*s;
u4 = x2-(N1 + N2)*c;
v4 = y2-(N1 + N2)*s;
vpl(u1,v1,u2,v2,255)         % Draw a line from [u1,v1] to [u2,v2]
vpl(u3,v3,u4,v4,255);        % Draw a line from [u3,v3] to [u4,v4]
vpl(u3,v3,u1,v1,255);        % Draw a line from [u3,v3] to [u1,v1]
vpl(u4,v4,u2,v2,255);        % Draw a line from [u4,v4] to [u2,v2]
blf                          % We now have two open rectangles, so we
                               fill them
alternate = original;
subplot(1,2,1), imshow(current)
subplot(1,2,2), imshow(alternate)
```

The situation shown in Fig. 21.20b is safe.

To save programming effort, the author used another M-function (*grip1*) to prepare data for *grip2*, so that it can accept arguments in a form that is more suitable for handling discrete parts. The input parameters for *grip1* are then as follows:

```
function grip1(x,y,z,d,N1,N2)
% [x,y]                         Centroid coordinates
% z                             Orientation (normally use lmi to find)
% d                             Separation of the fingers when the gripper
                                  is ``open''
% N1                            Length of the gripper finger
% N2                            Width of the finger
global current
global alternate
original = current;
z = pi*z/180;
s = sin(z);
c = cos(z);
x1 = x + d*c;
y1 = y + d*s;
x2 = x−d*c;
y2 = y−d*s;
grip2(x1,y1,x2,y2,N1,N2);
alternate = original;
subplot(1,2,1), imshow(current)
subplot(1,2,2), imshow(alternate)
```

Although grip1 is computationally inefficient, it does the necessary calculation with minimal reprogramming effort (➜ *Fig. 21.21*). Notice that we can model a gripper with circular or rectangular fingers. The principle for modelling a vacuum gripper is simple: to avoid air leakage, which would lead to unsafe lifting, the suction pad must not overlap the edge of the object silhouette (➜ *Fig. 21.21d*). A similar principle applies to a magnetic gripper, although a small amount of overlap might be tolerated.

21.8 Controlling External Devices

Practical Machine Vision systems do not operate in isolation from the real world; factory-based visual inspection/control systems require an image processing engine that can be interfaced easily to a variety of external devices, such as:

- *Lamps* (filament bulbs, strobed discharge tubes, LED arrays, etc.)
- *Optical systems* (lens aperture, zoom, focus, camera, pan and tilt, indexed filter wheel)
- *Cameras* (synchronising scanning and image acqusition, as well as setting internal control parameters)
- *Transport mechanisms* (ranging from simple indexed/continuous parts-transport mechanisms to multi-axis robots)
- *Actuators* (relays, solenoids, pneumatic/hydraulic valves, stepper-motors, DC motors, programmable logic controllers (PLCs), heaters, fans, machine tools lathe, drilling, milling machine, etc.)

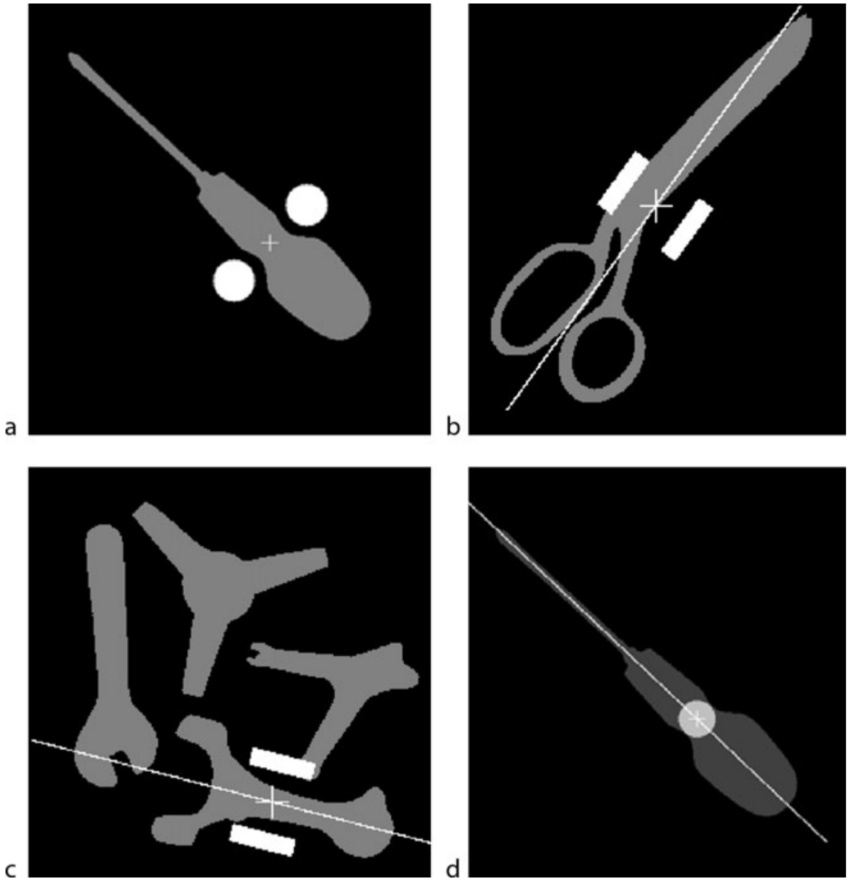


Fig. 21.21
Ensuring safe lifting. In each case, the centroid and principal axis were used to calculate the position and orientation of the gripper. (a) Screwdriver. The gripper has two circular fingers. This is a safe lifting configuration. (b) Scissors (*closed*). The gripper has two rectangular fingers. This is not safe as the upper finger collides with the scissors. (c) Picking up the conrod with other objects nearby. This is not safe as the upper finger collides with the Y-shaped object. (d) Picking up a screwdriver with a circular suction/magnetic gripper. The grass-fire transform (*gft*) was used to estimate the maximum size of gripper that can be used safely

QT's primary function is that of prototyping tool, so, it must be able to operate such devices, if it is to be useful in practical demonstrations and feasibility studies. It should be understood, however, that real-time operation is not an issue that we can address seriously. QT was designed as a multi-function development tool and hence speed has often been sacrificed for versatility.

MATLAB provides several tools for building an interface to external hardware:

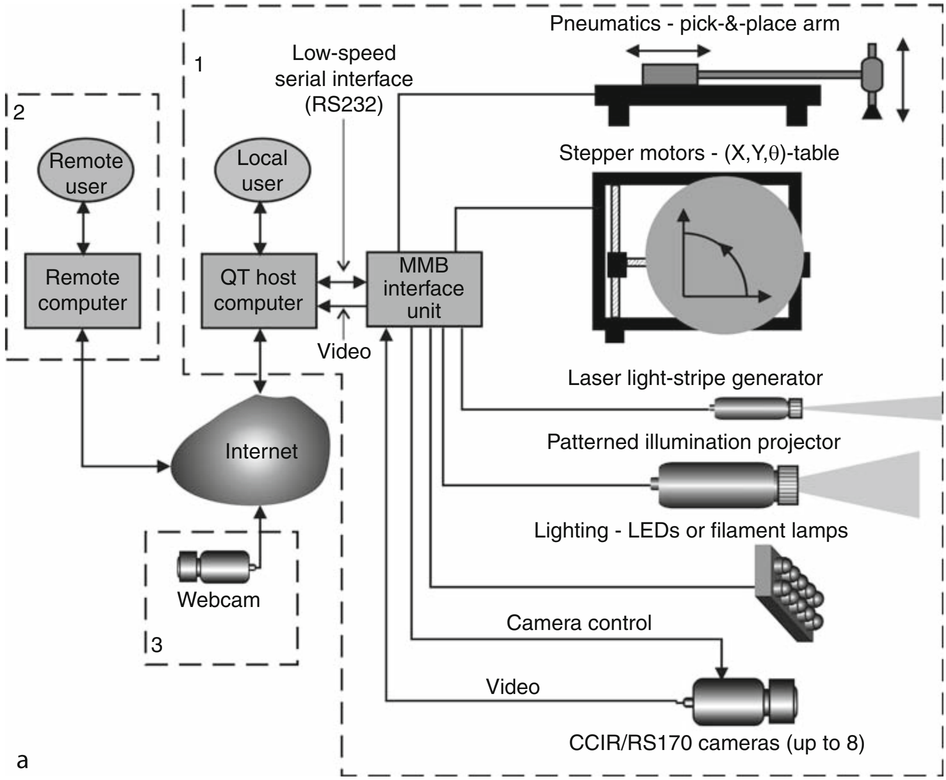
(a) Serial port
(b) Reading/writing files on the host computer
(c) Reading files on a remote computer, given an appropriate URL

(d) Communicating with external hardware via custom C or Java programs, running on the same host computer as QT

To date, QT has been provided with facilities for (a), (b) and (c) but not (d). The last mentoned remains an option for improving the speed of dedicated control systems but does not represent a serious loss for our present requirements.

Since MATLAB can operate a serial port, an obvious approach is to connect the host computer to another machine that has the required I/O capability (➲ *Fig. 21.22*). This is very convenient but inherently slow, since all I/O traffic flows via the serial port. However, we argue that this is not normally important in QT's intended role. Suppose, for example, that we want to set up a prototype automated visual inspection system in a factory. Clearly, we cannot expect QT to perform 100% inspection in any but the least demanding of applications. It might, for example, be possible to perform the appropriate processing on only 1% of all product samples. If the limited I/O capabilities reduce this to, say, 0.5%, the experiment will simply take a little longer! It is much more important that the equipment and software can be set up and adjusted quickly and easily, so that the customer is able to assess the benefits of using vision, whilst incurring minimal cost and causing little disturbance to the smooth running of the factory.

Serial port There are several available options for the I/O control processor, including. The authors have used a low-cost system designed and built by one of their colleagues,


1
2
3
Low-speed serial interface (RS232)
Pneumatics - pick-&-place arm
Stepper motors - (X,Y,θ)-table
Remote user
Local user
Remote computer
QT host computer
MMB interface unit
Video
Laser light-stripe generator
Internet
Patterned illumination projector
Lighting - LEDs or filament lamps
Webcam
Camera control
Video
CCIR/RS170 cameras (up to 8)
a

■ **Fig. 21.22**
(Continued)

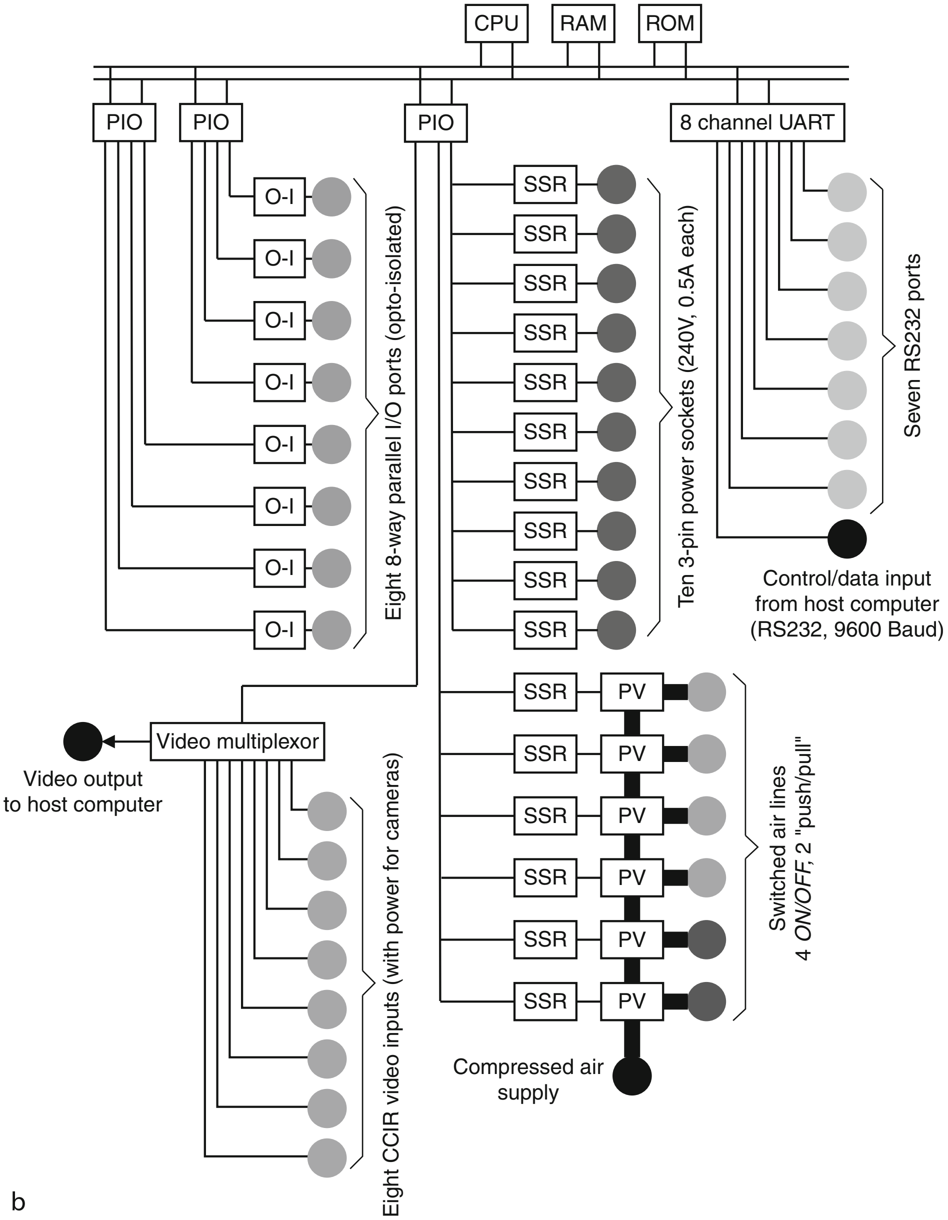

CPU
RAM
ROM
PIO
PIO
PIO
8 channel UART
O-I
Eight 8-way parallel I/O ports (opto-isolated)
SSR
Ten 3-pin power sockets (240V, 0.5A each)
Seven RS232 ports
Control/data input from host computer (RS232, 9600 Baud)
Video multiplexor
Video output to host computer
Eight CCIR video inputs (with power for cameras)
PV
Switched air lines 4 *ON/OFF*, 2 "push/pull"
Compressed air supply
b

Fig. 21.22
QT controlling external hardware via a serial interface. (a) System organisation. This does not normally create a communications bottle-neck, since the opto-electro-mechanical devices are all slow. However, these must be located close to QT's host computer (site 1). Since it is possible to operate QT remotely via a web browser (site 2) and to digitise images from a remote webcam (site 3), this arrangement can be limiting. (b) Details of the MMB architecture (Designed by Michael Daley, Cardiff University)

Michael Daley. This is called MMB [9, 12] (➜ *Fig. 21.22b*). This is built around a slow processor (Intel 8086). There is only a very small amount of RAM, one ROM chip for firm-ware written in assembly code. MMB provides the following I/O facilities:

(a) One serial port, which exercises control over the function of the MMB unit and passes all I/O traffic to/from QT.
(b) Seven serial ports, any of which can be multiplexed to receive data from the control port. These can operate at different speeds, with run different protocols.
(c) Eight 8-way parallel I/O ports.
(d) 8-way video multi-plexor.
(e) Four *ON/OFF* compressed air lines.
(f) Two complementary-pair compressed-air ports, each allowing forward/reverse operation of a pneumatic cylinder.
(g) Ten *ON/OFF* mains-power lines, each capable of supplying 2.5 A at 240 V, 50 Hz. (Total load must not exceed 13 A.)

Notice that the MMB unit and other devices interfaced via the serial port must be located close to the QT host processor. That is, QT must be running on a computer that is physically close to the object being inspected. Therefore, we cannot use the serial-port interfacing option for running QT on a computer that is far removed from the point of inspection.

USB Port The Velleman *K8055 USB Experiment Interface Board* [Velleman K8055 USB Interface Board, URL http://www.velleman.eu/, accessed 13th December 2010] is a low-cost micro-controller device that is able to control the following input/output channels, via a single USB port:

Five digital input channels
Eight digital output channels

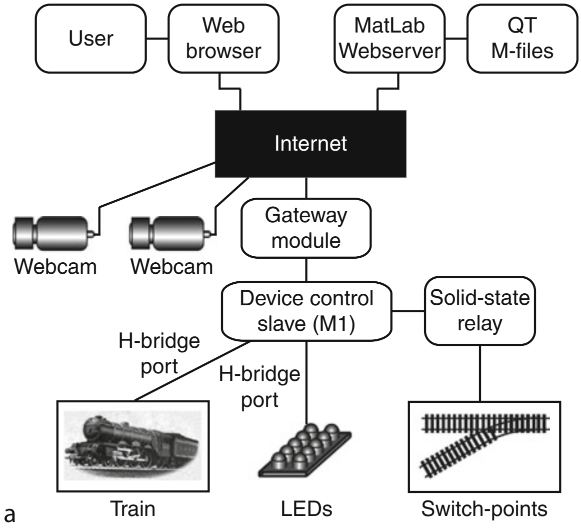


Fig. 21.23
(Continued)

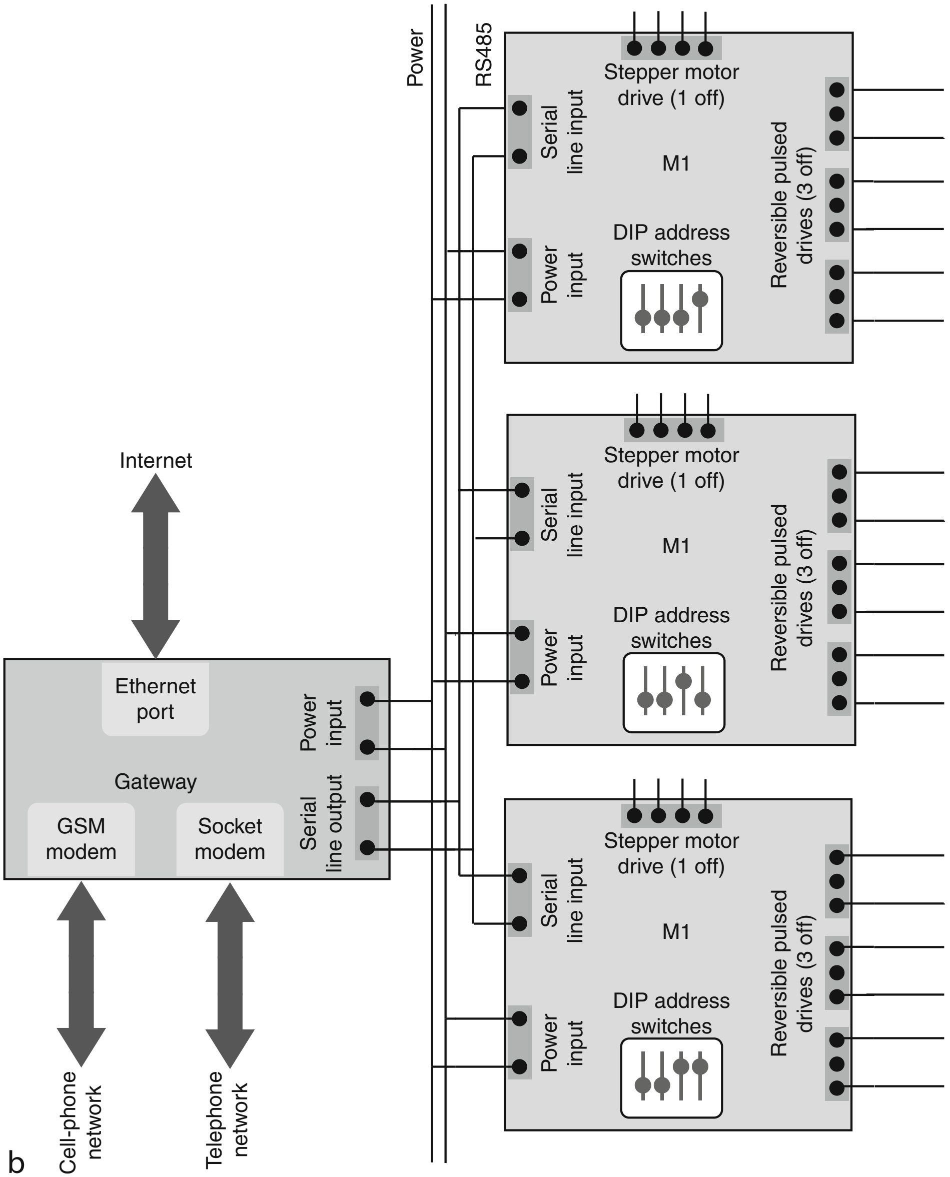


Fig. 21.23

QT controlling external hardware via the Internet. (a) System organisation using the Gateway and M1 boards. Up to 16 M1 boards can be connected in a daisy-chain and controlled from a single Gateway board. Each M1 board provides three H-bridge ports, one stepper-motor controller and four parallel TTL output lines. (b) Connecting the Gateway and M1 boards. Other types of boards, providing alternative I/O control facilities, can be inserted into the daisy-chain. The author used this arrangement to operate a simple model railway layout

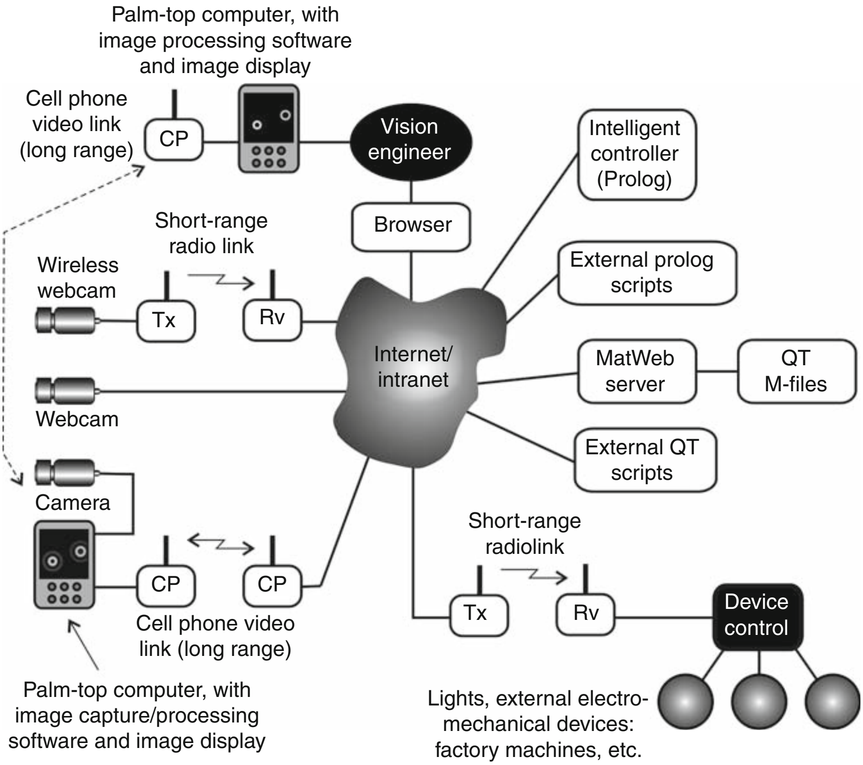

Palm-top computer, with image processing software and image display
Cell phone video link (long range)
CP
Vision engineer
Intelligent controller (Prolog)
Browser
Short-range radio link
Wireless webcam
Tx
Rv
External prolog scripts
Internet/ intranet
MatWeb server
QT M-files
Webcam
External QT scripts
Camera
Short-range radiolink
CP
CP
Tx
Rv
Device control
Cell phone video link (long range)
Palm-top computer, with image capture/processing software and image display
Lights, external electro-mechanical devices: factory machines, etc.

Fig. 21.24
Operating QT via the Internet

Two pulse-width modulated (PWM) digital outputs
Two analogue inputs
Two analogue outputs

The board derives its power from the host computer's USB port. Details of the driver software for the Windows XP version of QT are given in Appendix J. Up to a maximum of four K8055 boards can be connected together, thereby increasing the number of I/O channels available. The K8061 module is similar to the K8055 board and provides more I/O channels, with a higher resolution. A similar range of devices is available from Arduino.

Ethernet/Web-based I/O Control Web-friendly hardware is now readily available for controlling devices such as those listed above. For example, a flexible family of devices is available from [J43 *Designs*, Neath, West Glamorgan SA10 7TG, UK] (➔ *Fig. 21.23*). Overall control is exercised via the so-called *Gateway* board, which can be controlled from its Ethernet port. Each *Gateway* board is assigned a unique URL and can control up to 16 slave modules, which are connected directly to the external equipment. The company provides slave modules with the following output facilities:

(a) Parallel ports.
(b) Serial ports. (RS485/RS232)

(c) Switched *H-bridge* ports. (These provide a succession of reversible, variable pulse-width, fixed-level pulses. An *H-bridge* port is ideal for controlling the speed/direction of a DC motor and can also be used to control the brightness of low power lighting modules.)
(d) Stepper motor driver.

Since the Gateway board is web-based, an unlimited number of them, with their attendant slave modules, can be controlled by single QT processor. These do not have to be located close to the QT-host computer. This provides an important advantage over the serial-port and USB interfaces and allows us to build a distributed prototyping system (➋ *Fig 21.24*).

21.9 Concluding Remarks

The authors remain convinced that a command-line system, with abbreviated mnemonic names and short argument lists provides the best way to study Machine Vision applications interactively. Some other worker taken a different approach. (See, for example, ➋ Chap. 22.) While a certain investment of time is needed to master QT's cryptic mnemonics, the present author asserts that this is well worthwhile, in the long term. He was an early and enthusiastic advocate of the Macintosh operating environment (long before MS-WINDOWS appeared) but he never felt that a GUI was the appropriate way to operate an interactive image processor. Pull-down menus were made available in PIP, QT's immediate predecessor, but were never found to provide an adequate substitute for command-line control. Similarly, a set of MATLAB short-cuts was established for QT. (A short-cut is a labelled icon residing at the top of the Command Window and executes a predefined command sequence with a single click.) Short-cuts have never been found to be really useful, except as a means of selecting standard images by name, rather than index number.

MATLAB, with its Image Processing Tool-box (IPT), provides a very versatile system for processing and analysing images. The fact that QT is implemented using MATLAB provides testimony to this. MATLAB is capable of handling a wide variety of image and data types and it has a very rich command repertoire. As a result, it has found application in a wide range of engineering and scientific disciplines. It is therefore worth asking why the authors felt it was necessary to add the QT M-functions to it; why not use "raw" MATLAB instead? (This term includes "core" MATLAB, plus the IPT, but without the M-files that constitute QT.) After all, everything that QT can accomplish, MATLAB can also do. There are several reasons:

- Individual MATLAB commands are often quite long and demand far greater mental effort on the part of the user than QT does. This is distracting from the real task: analysing images/designing algorithms.
- The source and destination images must both be specified for each operation. (QT always processes the *Current* Image.)
- Full path names are required when images are read from, or written to disc. (QT allows the use of image files to be specified more simply, by numbers or single-letter names, without recourse to file-name extensions, or path names.)
- Certain MATLAB functions provide unnecessary options. (Many operations, or command options, provided by MATLAB have no foreseeable use in Machine Vision applications.)
- Machine Vision can make good use of special image processing operators that have been programmed in MATLAB but which do not exist in its "raw" form.

- It is necessary to order the display of images explicitly. (QT displays results automatically.)
- There is no facility for normalising image intensity after each operation. (➜ Chap. 14. Normalisation is an important and integral feature of QT.)
- The distinction between grey-scale and binary images often hinders user interaction. (It is often helpful to be able to regard, and process, a binary image as if it it were an ordinary grey-scale image.)
- MATLAB encourages new images to be generated. These are very easily forgotten, leading to progressive increase in memory usage. (QT retains only two "permanent" images (*Current* and *Alternate*). In addition, it occasionally generates a few (usually no more than two) working images that are deleted immdiately after use.)
- Many MATLAB functions can be applied to only a limited group of image types. There are several different types of image and it is not obvious which type is currently being processed/displayed. (This makes the operation of "raw" MATLAB intellectually challenging for the user. When processing grey-scale images, QT has only two data types, so this problem does not arise.)
- Writing valid and effective MATLAB function sequences is far from straightforward, as each operation is liable to change the data type. QT always leaves the result in a form that can be processed by *any* other QT function. (Colour image processing operators do not conform to this convention.)
- Input and output images are sometimes of different sizes. (Certain functions, such as *conv2*, generate images that are of a different size from the original. Some other functions crop the output image, to remove "redundant" pixels. QT avoids this awkward situation.)

MATLAB is an excellent platform on which to build more specialised facilities. In its "raw" form, its versatility mitigates against rapid user interaction.

21.10 QT User Guide

21.10.1 What Is QT?

QT is a software package for interactive image processing. It provides an experimental tool-kit for application analysis and algorithm selection and was specifically designed to assist in the development of prototype industrial Machine Vision systems. It allows rapid selection and testing of image processing algorithms. QT has a large repertoire of easily executable operators: simply type the name of the appropriate function and, if necessary, a short list of control parameters. (➜ section 21.11 provides a list of QT commands.) This section concludes with an explanation of how a typical algorithm is developed using QT (➜ Sect. 21.10.7) and some graded exercises (➜ Sect. 21.10.8).

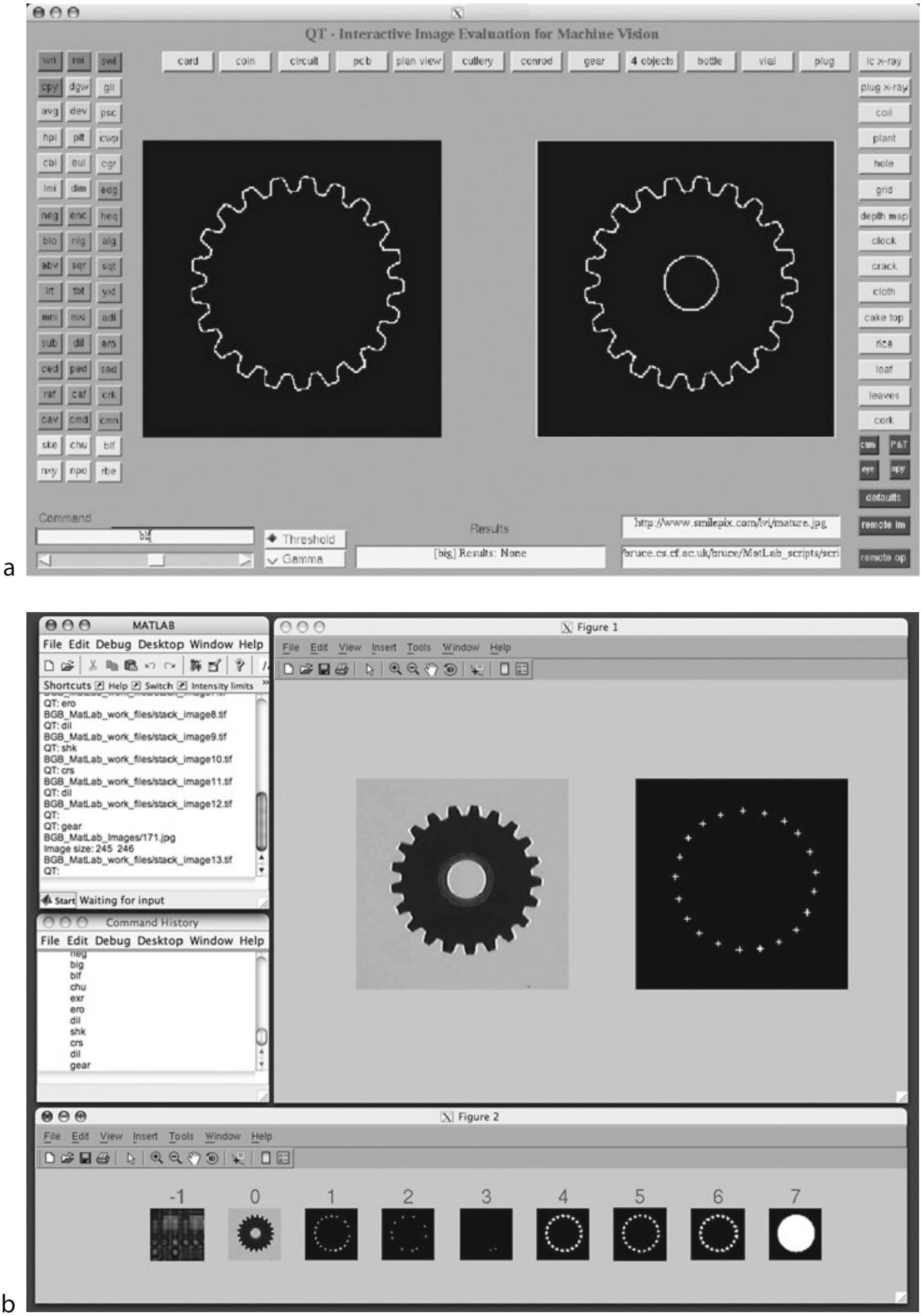
The QT software, a set of standard images and a number of other utilities for Machine Vision system design, can be accessed by visiting the author's web site at

http://bruce.cs.cf.ac.uk/index.html

21.10.2 Choose the Right Version

The following versions of QT are currently available:

- Single-user version, written and executed as a series on MATLAB M-files.


a
QT - Interactive Image Evaluation for Machine Vision
wri rei swi
cpy dgw gli
avg dev psc
hpi plt cwp
cbl eul cgr
lmi dim edg
neg enc heq
blo nlg alg
abv sqr sqt
lrt tbt yxt
mni mxi adi
sub dil ero
ced ped sed
raf caf crk
cav cmd cmn
ske chu blf
rdy npo rbe
card coin circuit pcb plan view cutlery conrod gear 4 objects bottle vial plug
ic x-ray plug x-ray coil plant hole grid depth map clock crack cloth cake top rice loaf leaves cork
cam P&T eye spy
defaults remote im remote op
Command
Threshold
Gamma
Results
b
MATLAB
File Edit Debug Desktop Window Help
Shortcuts Help Switch Intensity limits
Start
Command History
Figure 1
File Edit View Insert Tools Window Help
Figure 2
-1 0 1 2 3 4 5 6 7

Command: blf

[big] Results: None

http://www.smilepix.com/lvl/mature.jpg

bruce.cs.cf.ac.uk/bruce/MatLab_scripts/scri

```
QT: ero
BGB_MatLab_work_files/stack_image8.tif
QT: dil
BGB_MatLab_work_files/stack_image9.tif
QT: shk
BGB_MatLab_work_files/stack_image10.tif
QT: crs
BGB_MatLab_work_files/stack_image11.tif
QT: dil
BGB_MatLab_work_files/stack_image12.tif
QT:
QT: gear
BGB_MatLab_Images/171.jpg
Image size: 245  246
BGB_MatLab_work_files/stack_image13.tif
QT:
```

Waiting for input

```
neg
big
blf
chu
exr
ero
dil
shk
crs
dil
gear
```

Fig. 21.25
Screen layout, (a) Window mode. (b) Terminal and Stack Modes

- Single-user stand-alone executable file, running under LINUX or WINDOWS.
- Multi-user, multi-processor version.
- PQT using SWI-Prolog [URL: www.swi-prolog.org/] as the top-level controller.

21.10.3 Getting Started

Visit the following web site:

http://www.springer.com
News and updates are made available from time to time at the author's web site:
http://bruce.cs.cf.ac.uk/bruce/

21.10.4 Modes of Operation

QT has been provided with three modes of operation:

- Window Mode (➔ *Fig. 21.25a*)
  Novice users (The author uses this mode for teaching degree-level and post-graduate students.)
  Single window, fixed format display
  Shows only current and alternate pictures
  Buttons, loading standard test images
  Buttons, executing commonly used functions
  Sliders for adjusting threshold and gamma
  Text box for command-line operation
  Restricted functionality (It is difficult to use with functions requiring additional windows)
  Unsuitable for editing M-files
- Stack Mode (➔ *Fig. 21.25b*)
  Intermediate level users
  User can reformat/resize image display
  Displays current and alternate pictures
  Displays thumbnails of images in the stack (User clicks on a thumbnail to read thecorresponding image from the stack.)
  Keyboard short cuts for common operations (e.g., *pop* and *push* the stack)
  Most MATLAB features are available (e.g., History Window, cursor, 3D plots, etc.)
  Does not permit editing of M-files
- Terminal Mode (➔ *Fig. 21.25b*)
  Experienced users (used for software development)
  As Stack mode in most respects but more functions are available
  Stack display is used to show saved images (accessible at random)
  Permits M-files to be added/edited

➔ *Table 21.2* summarises the features available in these different modes.

Table 21.2
QT operating modes. Features marked with an asterisk (*) can be controlled via command line

Function	Window mode	Stack mode	Terminal mode
Cursor	Limited functionality	Yes	Yes
Interactive thresholding and gamma adjustment	Yes	Yes	Yes
Reading standard images	Buttons or command line	Command *rti* loads nine popular images into image stack. User then clicks on thumbnail to read image. Can also read images using command line	
Read images	Button* (single image store available)	CLICK on thumbnail, or via command line (nine thumbnails are displayed in the image stack)	
Save images		CNTRL/CLICK on thumbnail* (storage for eight images available)	
Extra windows (e.g., 3D plots and multi-image display)	No	Yes	Yes
User prompt	Flashing vertical line in *Command* box	QT:	>>
Commands (single)	Buttons or command line	Command line only	Command line only
Command sequences	No (results not displayed)	Yes	Yes
Keyboard short cuts	No	Yes	No
Automatic saving of images during interaction (*push* onto stack)	No	Yes	No (stack can be operated via command line)
Sound playback	Yes	Yes	Yes
Error message reporting	Limited	Limited	Full
MATLAB History Window	Present but not visible	Yes	Yes
On-line help and documentation	No	Yes	Yes
On-line review of standard images	No	Yes	Yes
Editing M-files	No	No	Yes
MATLAB debug tools	No	No	Yes

21.10.5 Operating QT

Except where stated, the notes in this section refer to Stack and Terminal modes.

21.10.5.1 Basics

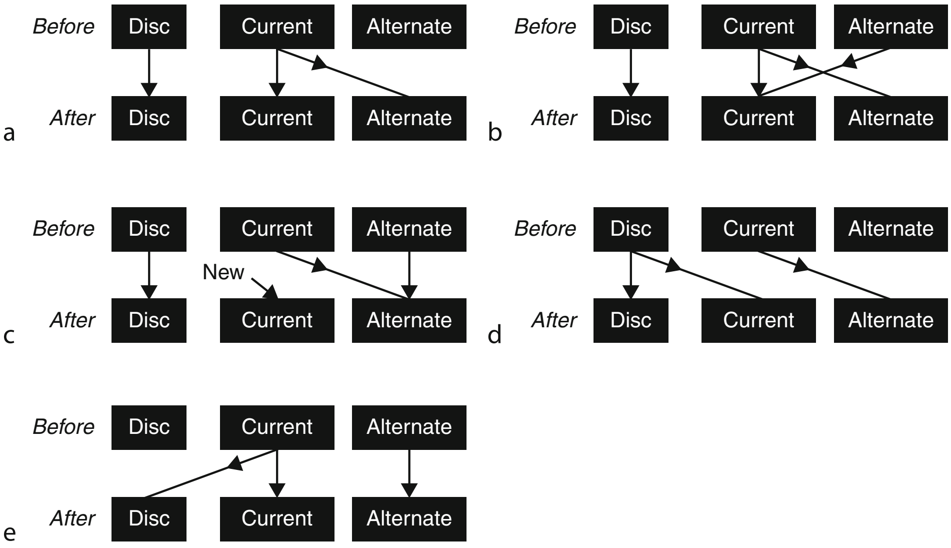
QT uses and displays only two images, called the Current and Alternate Images. The Current Image is always displayed on the left and the Alternate Image on the right. When a typical command, such as *neg* (negate) is typed, the following happens:

1. The Current Image is copied into the Alternate Image. The original contents of the Alternate Image are discarded.
2. The specified operation is performed on the Current Image.
3. The result (negative image) is then stored in the Current Image.
4. The revised Alternate Image is also displayed alongside the Current Image.

For example, thesholding (function *thr*) requires zero, one or two parameters; the following are typical commands:

• *thr(123,234)*	Two arguments supplied
• *thr(156)*	Second argument is, by default, assumed to be 255 (white)
• *thr*	First argument default value is 128. Second is 255.

Functions *thr* (with and without arguments), *neg* and most other QT commands employ the Current and Alternate Images in the same way. However, there are some deviations from this model, reflecting the different types of operators. There are five other situations to consider. (See ➲ *Fig. 21.26*.)



◘ Fig. 21.26

Mapping the Current and Alternate Images during the execution of QT commands. (a) Monadic operators, local operators and other functions that use a single input image behave in this way. (b) Dyadic operators. (c) Image generators. (d) Reading images from a disc file. Functions that down-load images from the web behave in a similar way. (e) Writing images to disc

(a) Some commands return values calculated from the Current Image but do not alter it, nor the Alternate Image. For example, *avg* computes the average grey level in the Current Image. The result(s) can be used to define variables that will be used later to define input values for other commands, or simply printed.

(b) Some commands generate images. For example, *gsn* generates a random image from a Gaussian noise source. In this case, the Current Image is first copied to the Alternate Image. The Current Image is then replaced by the Gaussian noise image. Notice that this process is compatible with the operation of the majority of commands, such as *neg*. Reading an image from a disc file is also performed in this way. Popping an image from the Image Stack also behaves in a similar manner. (Operations involving the Image Stack will be discussed later.)

(c) Writing an image to disc and pushing an image onto the Image Stack do not alter either image.

(d) A few commands, such as *rgb*(show RGB channels separately), *hsi* (show HSI channels separately) and *plot3d* (plot intensity as a function of X and Y coordinates) generate a new display window (MATLAB refers to this Figure 4), without altering either the Current or Alternate Image.

Note: After executing *rgb*, *hsi*, *hpi*, plot3d, click anywhere on Figure 4, to close it

(e) A few commands combine the Current and Alternate Images. For example, function *adi* adds the intensities of pixels at corresponding addresses in the Current and Alternate Images. In such cases, the result is placed in the Current Image and the original Current Image is copied into the Alternate Image, *w*hose original contents are discarded.

21.10.5.2 Combining Commands

QT commands can be concatenated as shown in the following example:

wri(100), *dil*, *ero*, *rei*(100), *sub*, *enc*

Alternatively, when writing new MATLAB scripts or functions, individual commands can be written on separate lines, with optional comments added as illustrated below:

```
wri(100)       % Write the Current Image to disc file 100.jpg
dil,           % Dilation using default parameters
ero            % Erosion using default parameters
rei(100),      % Read image file 100.jpg
sub,           % Subtract the Alternate Image from the Current Image
enc            % Enhance contrast
```

Notice that commas can be included at the end of each line but are not necessary. (Semicolons, which will be considered next, would have no effect in this example.)

Some commands generate numerical results. For example, *avg* calculates the average intensity, which can be used by a later command. The following command sequence illustrates this is.

$$x = avg,\ \ thr(x)$$

avg calculates the average intensity and *thr* performs simple thresholding. Notice the comma separating these two commands. The effect of this comma is to allow MATLAB to display the value of the newly assigned variable *(x)*. This is a design feature of MATLAB. To suppress the display of this variable, we must use a semi-colon instead:

$$x = avg;\ \ thr(x)$$

If we wish to place these commands on separate lines to make reading easier, the semi-colon again suppresses the display of the values for *x*:

```
x = avg;                         % The value of x is not displayed
thr(x)                           % Threshold at level x
```

Whole images can be assigned using a construction of the form

$$\text{current} = \text{original}$$

Warning: This displays the intensities for the whole image, pixel by pixel. For a colour image, there are three values displayed for each pixel. This is a lot of data and the command takes a long time to execute!

To suppress the output display, use a semi-colon:

$$\text{current} = \text{original};$$

Some commands generate several output values, as the following example shows (This is the heart of the function *nxy*, which normalises the position of a blob so that its centroid lies at the centre of the image.)

```
[x1,y1] = cim        % No semi-colon, so the values of x1 and y1
                       are displayed.
[x2,y2] = cgr        % No semi-colon, so the values of 2x and y2 are displayed.
q = round(x1-x2);    % Standard MATLAB rounding functions. Output is
                       suppressed.
p = round(y1-y2);    % Standard MATLAB rounding functions. Output is
                       suppressed.
current =            % Take care to place a semi-colon here to avoid
  original;            long delays.
psw(p,q)             % Shift image horizontally by p pixels and vertically
                       by q pixels.
```

21.10.5.3 Special Features

- Switching Current and Alternate Images
  Terminal Mode — *swi* or *z* (single letter command)
  Window and Stack Modes — *swi*, *z* or simply press the ENTER key
- Interactive investigation of features (position and intensity/colour)
  pxv

- Dynamic thresholding
  - Window mode
    - Slider (Click on radio button labelled "*Threshold*" first)
    - Note: Result is left in the *Alternate Image*
  - Stack and Terminal modes
    - Use command *dth*
- Dynamic gamma adjustment
  - Window mode
    - Slider (Click on radio button labelled "*Gamma*" first)
    - Note: Result is left in the Alternate Image
  - Stack and terminal modes
    - Use command *dgm*
- Display standard images (Stack and terminal modes)
  - *img* Displays all of QT standard images via MATLAB web browser
  - *rti* Loads most popular images into the image stack. (Click on the thumbnail to load into the Current Image)
- Help (Stack and terminal modes)
  - *hlp* Lists all QT functions with a brief description of each one
  - *hlp text* List QT function "text" if it exists

  Otherwise search for all occurences of "*text*" in the abbreviated QT descriptions
- Documentation of individual image processing functions
  - *doc* Same as *hlp*
  - *doc cmd* Display documentation file for QT command "*cmd*"
- Presentation of results
  - *dtx* Switch between minimal and verbose presentation of results
- Mode switching
  - *qt*/*qt(0)* Switch to Terminal Mode
  - *qt(1)* Switch to Stack Mode

  Note: *end*, *halt*, *quit*, etc. switch from Stack to Terminal Mode
  - *qt(2)* Switch to Window Mode. Close QT window to switch to re-enter Terminal Mode
- Interactive drawing of a polygon
  - *pgn* Double click to exit. (Useful for drawing masks.)
- Cursor
  - *cur* Double click to exit
  - *pxv* Fast investigation of position and intensity. Does not return usable numeric results
  - *cob* Choose one blob using cursor
- Calibration of grey-scale
  - *cbr* Display colour bar. Dispappears when next command is executed
- Pseudo-color
  - *psc* Result is an RGB image
- Getting web command sequence
  - *ersf* Execute a remote script file. (Repeats process with time limit.)
- Acquiring image from web camera
  - *gwi* Argument is URL of the camera image

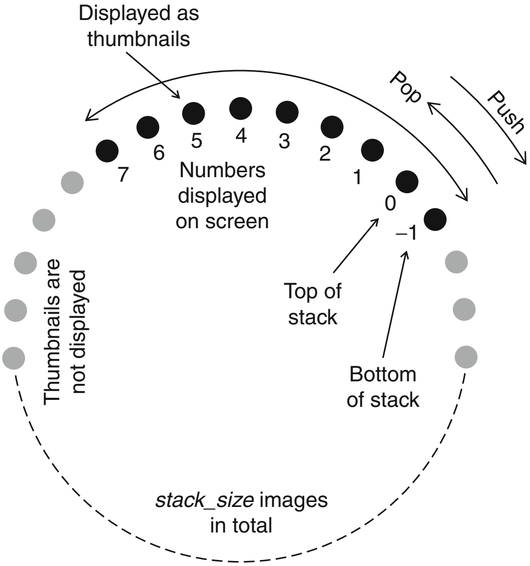

Displayed as thumbnails
Pop
Push
7
6
5
4
3
2
1
0
−1
Numbers displayed on screen
Thumbnails are not displayed
Top of stack
Bottom of stack
stack_size images in total

Fig. 21.27
Organisation and operation of the Image Stack. (Stack Mode)

Table 21.3
Relating thumbnail labels to the Image Stack

Screen label	Image no.	Image
−1	*stack_pointer* – 1 (mod *stack_size*)	Image at the bottom of the stack. This is the image that was previously at the top of the stack, i.e., just before the most recent *pop* operation
0	*stack_pointer*	Image at the top of the stack
1	*stack_pointer* + 1 (mod *stack_size*)	Image next to the top of the stack
2	stack_pointer + 2 (mod *stack_size*)	Second image from the top of the stack
N (1 ≤ N ≤ 7)	stack_pointer + N (mod *stack_size*)	Nth image from the top of the stack

- Controlling simple electronic hardware
  See Appendix J
- Viewing/searching the Catalogue of Lighting-Viewing methods
  lvm View the catalogue or perform a simple search
  lvs Multi-term search of the catalogue

Table 21.4

Stack and Terminal Modes. The behaviour of some standard QT commands is slightly different in Stack and Terminal Modes. (Indicated by "*")

Command/Action	Mode	Function
*end**	S	Exit *Stack Mode*; return to *Terminal Mode*
exit		
stop		
*quit**		
halt		
qt(0)	S	Enter Terminal Mode
qt(1)	T	Enter Stack Mode
*qt(2)**	S/T	Enter Window Mode
spon/spof	S	Stack enabled/disabled
sdon sdof	S	Stack display on/off
Null *(CR only)*	S	Interchange Current and Alternate Images. Current Image is put on the stack
pop/push	S	Pop/push the image stack
b	S	Rotate the stack array back one step.
c	S	Grab an image from same the camera as was last used by the *grb* command
f	S	Rotate the stack array forward one step
h	S	Help: display command list
r	S	Read image from temporary file. (See command "w".)
s	S	Read same standard QT image as the last time *rni* was used
t	S	Get image from the top of the stack but do not alter the stack
w	S	Write image to a temporary file. (See command "r". No option which one.)
w	T	*w(N)* writes an image to the Nth stack image. Display the thumbnail in Figure 2.
z	S/T	Interchange Current and Alternate Images. Stack is not changed
*csk**	S	Clear the stack. Fill it with images generated by a Gaussian noise source
dtx	S/T	Display text ON/OFF
img	S/T	Display QT's standard images
rti	T	Load image stack with most popular test images
ssk	S/T	Show (part of) the stack array
*Other QT commands**	S	Execute command as normal, then push Current Image onto the stack
Failed command	S	*Warning*: That command was not recognised – try again
Click on thumbnail in Figure 2	S/T	Selected image is copied from the stack to Current Image
CNTRL/click on thumbnail in Figure 2	T	Copy image to selected stack image

21.10.6 Image Stack

The Image Stack display is used in different ways in Stack and Terminal Modes. The stack consists of *stack_size* images arranged in a circular array. The position of the top of the stack is indicated internally by the software by an integer variable, called *stack_pointer*, which is incremented/decremented (modulo *stack_size*) during *pop* and *push* operations. Eight images residing at or near the top of the stack on the screen are displayed as thumbnails, in a window labelled "➋ *Figure 21.2*" (see ➋ *Fig. 21.27*). In addition, the image at the bottom of the stack is displayed on the screen and is labelled "−1". Its use will be come clear later. The numeric labels in this window should be interpreted as shown in ➋ *Table 21.3*.

21.10.6.1 Stack Mode (➋ *Table 21.4*)

As each command is entered, the result (i.e., the Current Image) is pushed automatically onto the stack and the screen display updated. QT allows the user to revert quickly and easily to an earlier stage in the algorithm development process by

(a) Using command *rsi(N)*, where N ∈ {1,0,2,3,. . .,7}.
(b) Using command *r(N)*, where N ∈ {1,0,2,3,. . .,7}.
(c) Using command *pop* to pop the stack. The screen display is updated automatically.
(d) Using the single-letter command '*b*' (for backwards) to pop the stack. (Use the single-letter "*f*" to correct a mistake if you type "*b*" too often.)
(a) Clicking on a thumbnail in the stack display window (➋ *Fig. 21.25b*) copies the corresponding (full-size) image into the Current Image.

Functions *pop*, *push* and *tsk* (top of the stack), csk (clear stack) and *rsk*(fill stack with images chosen at random) are also available. *rti* fills the stack with popular test images. This is done automatically at start-up.

It is possible to pop the stack using a single-letter command (*b*). Hence, it is easy to go too far and move *stack_pointer* past the desired image, by mistake. Since the bottom of the stack (thumbnail "−1") is visible, the user can easily remedy the error by reversing the last pop operation, using command *f*.

21.10.6.2 Terminal Mode (➋ *Table 21.4*)

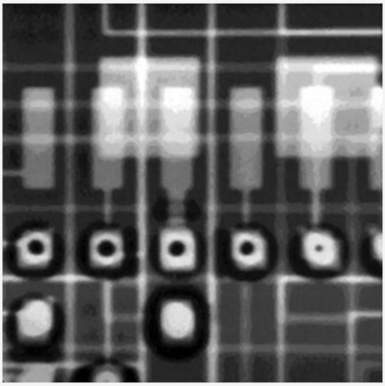
In this mode, the stack is not normally updated automatically, as it is in Stack Mode. (This is possible but the user must work without the benefit of the visual display of the stack contents.) Instead, the stack display can be used to see (some of) the images that have been saved to disc. To save an image so that a thumbnail is visible in the stack display window (labelled "Figure 2"), use one of the following:

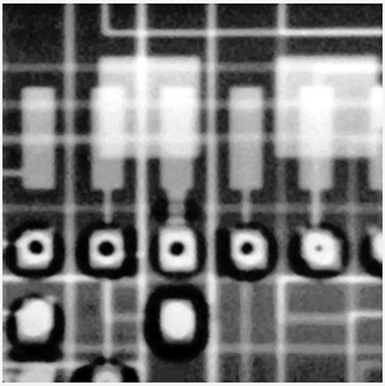
(a) Command *wis(N)*, where N ∈ {1,0,2,3,. . .,7}.
(b) Command *w(N)*, where N ∈ {1,0,2,3,. . .,7}. (N.B. *w(N)* and *wis(N)* are synonymous.)
(c) Clicking on the thumbnail while holding the CONTROL key down.

Notice that *wis/w* serves the same purpose as *wri*, except that the latter does not provide a visual display of the saved image. Hence, the stack display effectively provides a 9-picture

Table 21.5

Developing an algorithm for locating the centres of the holes in the component mounting pads

QT	Function	Result
rni(135)	Read the original JPEG image from disc. The intensity histogram is shown below	

rni(135);
heq;
mdf
	Reduce camera noise.
	Composite image, showing the effects of the median filter.
	Histogram equalisation was applied to make the differences appear more pronounced to the reader; it is not used elsewhere in the processing sequence.
	Top-left: *heq* (i.e., no filtering)
	Bottom-right: *heq; mdf* (i.e., after filtering)
		

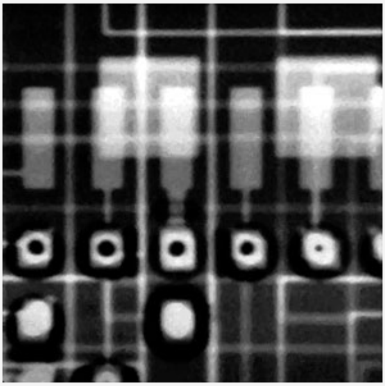
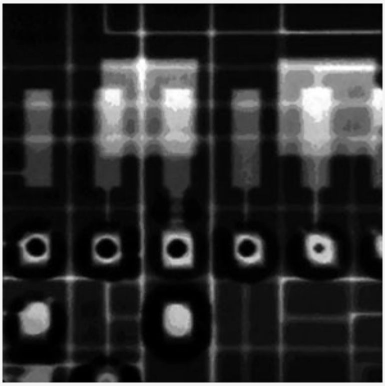
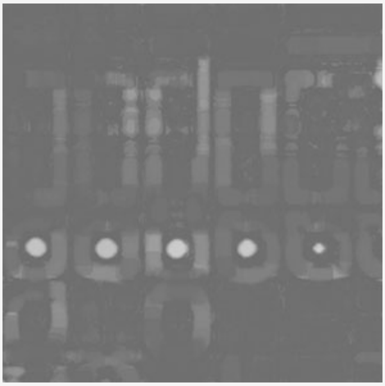
rni(135);
mdf;
sqr
	Square all intensities. Provides greater contrast in dark regions.	

Table 21.5 (continued)

QT	Function	Result

rni(135);
mdf;
sqr;
ero
	Grey-scale erosion: shrink all bright regions	

rni(135);
mdf;
sqr;
ero;
crk(7)
	Crack detector: grey-scale morphological closing operator	

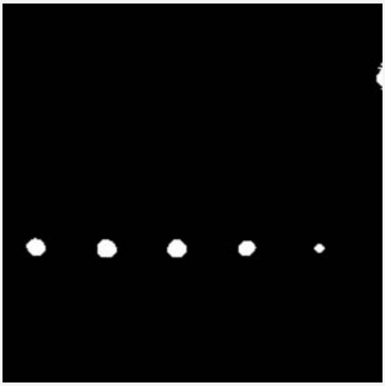
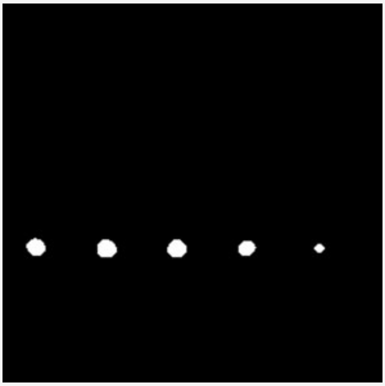
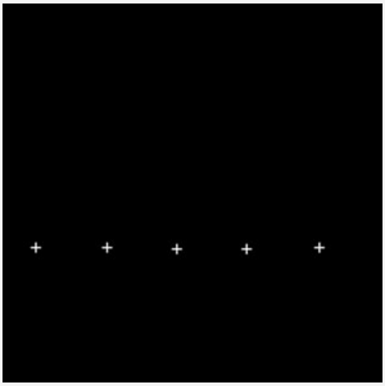
rni(135);
mdf;
sqr;
ero;
crk(7);
thr(180)
	Threshold at level 180 (light grey). The value of this parameter was chosen by examining the intensity histogram of the image above. This histogram is shown below.	

Table 21.5 (continued)

QT	Function	Result

rni(135);
mdf;
sqr;
ero;
crk(7);
thr(180);
rbe
	Remove all blobs that touch the edge of the image. (Erosion and closing operators produce anomolous effects around the border.)	

rni(135);
mdf;
sqr;
ero;
crk(7);
thr(180);
rbe;
shk;
crs
	Shrink all blobs to a single point (shk).
	Then, draw a white cross at each white pixel.
	The XY-coordinates of the pads can be obtained using the following command sequence. (The array *z* is redundant.)
	rni(135);
	mdf;
	sqr;
	ero;
	crk(7);
	thr(180);
	rbe;
	[x,y,z] = gap
		

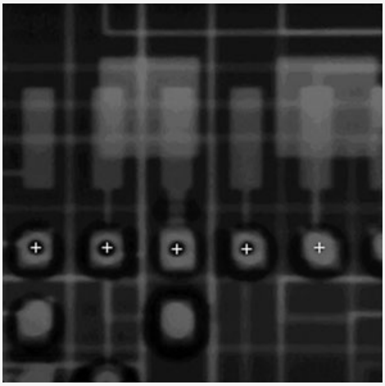
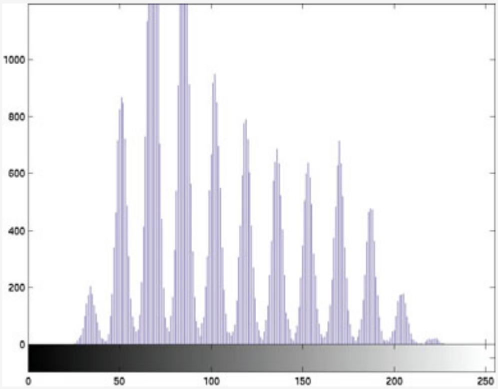
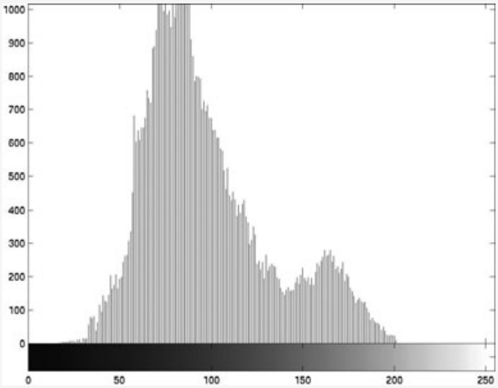
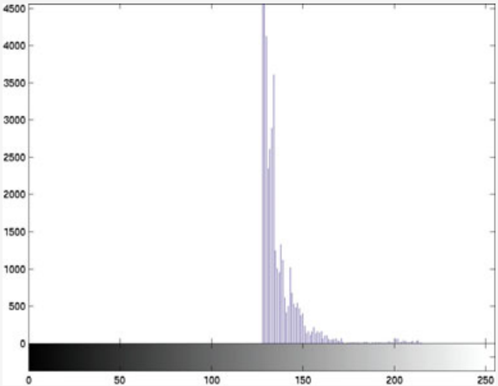
rni(135);
mdf;
sqr;
ero;
crk(7);
thr(180);
rbe;
shk;
crs;
rni(135);
swi;
adi;
mxi
	Superimpose the crosses onto the original image. This is helpful, so that the user can verify that the processing produces sensible results	

Table 21.5 (continued)

QT	Function	Result
rni(135); *hpi*	Intensity histogram of the original image. This comb-like structure is an artifact of the JPEG coding process. The histogram derived from the image obtained directly from the camera resembles that shown below	

rni(135);
caf;
hpi
	Intensity histogram of a low-pass filtered (i.e., blurred) image	

rni(135);
mdf;
sqr;
ero;
crk(7);
hpi
	Histogram of the difference image (i.e., after *sub*) is useful for determining an appropriate threshold value	

random access store with visual feed-back. The command sequence *[wis(n),ssk]* can be used to provide a dynamic display during computation.

- ► Notice that clicking on a thumbnail in the stack display window does not update MATLAB's History file.

21.10.7 Developing an Algorithm Using QT

The development of a simple algorithm is illustrated in ➋ *Table 21.5*. In practice, the user normally explores a multi-branch search tree. For clarity, dead-end branches have been eliminated here.

21.10.8 Exercises

21.10.8.1 Locating Pads on a Printed Circuit Board

Perform the steps in ➋ *Table 21.5*, one at a time. Check that you obtain the same images.

21.10.8.2 Crack Detection

Command	Explain the action and the significance of the result Take account of the Current Image and results in the Command Window
rni(41)	
crk(5)	
enc	
thr(220)	
kgr(100)	
cbl	
rni(41)	
mxi	

21.10.8.3 Telling the Time

Command	Explain the action and the significance of the result Take account of the Current Image and results in the Command Window
rni(141)	
rni(142)	
mxi	
dil	
rni(143)	

sub	
enc	
thr(64)	

- Confirm that the centre of the clock face is at [280.1380, 234.5614]. (*Hint:* first fill the circular contour around the outer edge and then find the centroid.)
- Estimate the radius of the clock face using:
  (a) mar
  (b) cwp
- Explain how *scc* might be used to locate the hour and minute hands.
- Draw a small cross at the centre of the face automatically and then superimpose it onto image # 141.

21.10.8.4 Analysing the Image of a Cog

Execute the following commands one at a time but ignoring the first line (i.e., *function [x,y] = cog*).

Then, execute *cog*.

Command	Explain the action and the significance of the result Take account of the Current Image and results in the Command Window
function [x,y] = cog	How many inputs? How many outputs?
% Analyse the image of a cog [2/0]	This is the H1 line. What is the significance of that? (Read MATLAB documentation.)
global current	Needed to gain direct access to the Current and Alternate Images
global alternate	
rni(210)	
original = current;	Why is the semi-colon there?
enc	
thr	
kgr(50)	
wri(1)	
blf	
chu	
exr	
ero	
shk	
[x,y,z] = gap;	What happens if we replace the semi-colon by a comma?
crs	
wri(2)	
rei(1)	

blf	
exr	
big	
a = cwp;	Will processing take much longer if we remove the semi-colon?
b = round(0.9*a);	Is the semi-colon needed?
swi	
kgr(b)	
str	
rei(2)	
mxi	
alternate = original;	What happens if we replace the semi-colon by a comma?
adi	
swi	
mxi	
alternate = original;	
subplot(1,2,1)	MATLAB functions for image display
imshow(current)	
subplot(1,2,2)	
imshow(alternate)	

21.10.8.5 Analysing the Creasing of Fabric

Obtain a plain white handkerchief and crease it in your hands. Place it gently on a flat-bed scanner. (If you do not have a scanner to hand, use image number 309.) Then, apply the processing steps described in ❯ Chap. 38 to the resulting image. They may need to be adapted to take account of the different image size.

21.10.8.6 Counting Teeth on a Comb

Analyse QT standard image 317 or, to count the teeth and place a cross at the tip of each one. Assume first that the orientation of the comb could be fixed to within a few degrees before image acquisition? Then, repeat the exercise without this benefit.

21.10.8.7 Inspecting a Coding Disc

QT standard image 162 is derived from a shaft-angle coding disc. This is to be assembled using a visually guided robot. Devise a QT command sequence for

(a) Checking that all holes (lakes) have been correctly punched.
(b) Verifying that the outer edge is circular.
(c) Finding the position of the centre of the disc.
(d) Determining the orientation of the disc.

21.10.8.8 Examining a Slice of Bread

Standard image number 45 shows the silhouette of a slice of bread from a non-lidded tin loaf. Assume that the orientation of the slice can be fixed approximately by mechanical means before image acquisition and that its nominal size is known.

Measure the radius of curvature of the top surface.

Measure the angles of the two straight sides.

Measure the area of the overspill (The dough expands during baking and bulges out, overspilling the top rim of the baking tin.)

21.11 QT: M-Files for Image Processing and Analysis

The following is a list of the M-files currently available in QT. Further functions are made available from time to time. These and documentation revisions are available from the authors' web site (URL: bruce.cs.cf.ac.uk). The author welcomes contributions by other workers.).

		Arguments	
Name	Function	In	Out
aab	Area of all blobs in a binary image	0	1
abd	Absolute value of difference in intensities twixt Current and Alternate Images	0	0
abv	Absolute value of the intensity in the Current Image	0	0
acf	Autocorrelation applied to an image	0	1
acn	Add a constant to all intensities in the Current Image	1	0
adc	Add the RGB channels of two colour images separately	0	0
adi	Add the Current and Alternate Images	0	0
aeb	Area of each blob in a binary image	0	1
aia	Write an image to an archive folder	1	0
aic	Adjust intensity range for RGB channels	2	0
aid	Aid the user to set up QT	0	0
air	Rescale intensities:	2	0
alg	Antilogarithm of the intensity of each pixel	0	0
ani	Anisotropic diffusion	2	0
apple	Read standard image: apple (colour)	0	0
asi	Average several images derived from a given webcam	2	0
author	Learn about the author of QT	0	0
avg	Average intensity	0	1
bay	Find the bays for a binary object	0	0
bbx	Bounding box, smallest rectangle containing the blob	0	4
bcl	Bit set	1	0
bcp	Bit complement	1	0
bdg	Bridge operator (morphology)	1	0

beans	Read standard image: coffee beans	1	1
bed	Binary edge detector	1	0
bgb	Information about Bruce Batchelor, the author of QT	0	0
big	Select the blob of given rank size	1	0
birds	Read standard image: birds (colour)	0	0
blf	Fill holes in a binary image	0	0
blo	Expand central part of intensity range; [64,192] is mapped to [0, 255]	0	0
blp	Find the bottom-left-most white (i.e., non-black) pixel	0	2
bnb	Bridge between the two closest blobs	0	3
bottle	Demonstration: Fit third order polynomial and circle to the outline of a bottle	0	1
bpa	Calculate shape/size parameters of all blobs in a binary image	1	1
brp	Find the bottom-right-most white (i.e., non-black) pixel	0	2
brx	Barrel rotate by shifting along the X-axis	1	0
bry	Barrel rotate by shifting along the Y-axis	1	0
btt	Bottom hat morphological filter	1	0
bxy	Barrel rotation along X- and Y-axes	2	0
cab	Centroids of all blobs in a binary image	0	1
cae	Find and count points where the intensities at corresponding points in the Current and Alternate Images are similar	1	1
caf	Circular averaging filter	0	1
caliper	Measure object diameter using a virtual caliper gauge	2	1
cam	Read standard image: cam	0	0
car_park	Read standard image: car park	1	0
card	Demonstration: Find suit and value of non-picture playing card	2	0
cav	Column average	0	0
cbd	Chessboard pattern	1	1
cbl	Count 4- or 8-connected blobs	1	1
cbr	Add color bar next to Current Image Does not alter either image	0	0
cct	Read standard image: thin film circuit	0	0
ccv	Get RGB colour values in the Current Image at a defined address	2	3
cdf	Colour difference between two images, applied to RGB channels separately	0	0
ceb	Use the least-squares method to fit a circle to the edge of a blob	1	3
ced	Canny edge detector	1	1
cen	Contrast enhancement applied to the RGB channels separately	0	0
ceo	Edge detector applied to the RGB channels separately	0	0
cfp	Closest white point [u,v] from a given point [x,y]	2	3
cga	Gamma adjustment applied to the RGB channels separately	3	0
cgp	Count points thta have a given intensity	1	1
cgr	Centroid coordinates, calculated from a single blob	0	2
cgr_all	Centroid coordinates, calculated from all blobs	0	2

chk	Generate check pattern	1	1
chq	Histogram equalise the RGB channels separately	0	0
chu	Convex hull – operates on only one blob at a time	0	0
chu_all	Convex hull of all blobs	0	0
cig	Count points with intensity greater than or equal to a	1	1
cim	Centre of the image	1	2
cin	Column integrate	0	0
cir	Draw an open circle	4	0
civ	Get intensity value in the Current Image at a defined address	2	1
cln	Remove isolated white pixels	0	0
clocks	Read standard image: clock face	1	0
cls	Closing with one of four pre-defined structuring elements	2	0
cmd	Column median	0	0
cmn	Column minimum	0	0
cmx	Column maximum	0	0
cnr	Corner detection: dog-on-a-lead, edge-following algorithm	2	0
cnw	Count white neighbours	0	0
cob	Choose one or more a-connected binary objects with the cursor (a = 4 or 8)	1	0
cog	Demonstration: Analyse the image of a cog	0	2
coil	Demonstration: Locate the ends of the flying leads on a small cylindrical coil	0	6
coin	Read standard image: coin	0	0
con	Same as *glc*. (Mac OS X. Causes an error with MATLAB running under Windows)	0	0
cox	Column maximum	0	0
cpy	Copy the Current Image to the Alternate Image	0	0
crack	Demonstration: Detect a crack in a forged ferrous component	0	0
cracked	Read standard image: cracked ferrous component	0	0
crease	Read standard image: creased fabric	0	0
crk	Crack detector with variable structuring element	2	0
crp	Crop the image interactively or by specifying four parameters	4	4
crs	Dilate so that an isolated white point becomes a vertical cross	1	0
csd	Column standard deviation	0	0
cse	Creating structuring elements for *gbd* and *gbe*	2	1
csg	Define polygon within colour image. Plot 3D scattergram of its RGB values	1	0
csh	Copy given column to all other columns	1	0
csi	Crop the image to create a 3-level mask and convolve this with whole image	4	3
csk	Clear the image stack. The stack size is also defined	0	0
csp	Generate a sine-wave intensity pattern that is rotationally symmettrical	3	0

ctp	Convert from Cartesian to polar coordinates	2	0
ctr	Draw intensity contours in the Current Image	0	0
cur	Get position of a point defined by the cursor	0	3
cwp	Count white points	0	1
cxa	Convex area for all blobs	0	1
dab	Do specified command for all blobs in a binary image	1	0
dbn	Direction of brightest neighbour	0	0
dcd	Draw a solid circular disc of radius a	1	0
dcg	Draw a cross at the centroid (of one blob)	0	0
dci	Draw a cross at the centre of the image	1	0
dcl	Draw a cross at a given point	3	0
dcr	Dilate so that an isolated white point becomes a diagonal cross	1	0
dcw	Draw a circular intensity wedge	3	0
dec	Draw equivalent circle (i.e., same area, located at the centroid)	0	0
delay	Delay of a given duration	1	0
demon	Demonstrations: a collection of demonstrations of QT	1	0
dev	Standard deviation of the intensity	0	1
dfc	Shade image so that intensity indicates distance from centroid (for all blobs)	0	0
dfp	Intensity is equal to the Minkowski r-distance from a given point, where r = 0,1,2,3,. . .	3	0
dga	Dynamic gamma adjustment	0	1
dgr	Diagonal gradient	0	0
dgw	Find the size of the image,	0	2
dic	Draw an intensity cone	2	0
did	Dynamic image display	0	2
dil	Dilate with one of four pre-defined structuring elements	2	0
dim	Find the size of the image	0	2
din	Double all intensities	0	0
div	Divide Current by Alternate Image	0	0
dla	Draw a line at a given angle from a given point	3	0
dlr	Draw one line identified from a peak in the Radon transform image	4	0
dls	Find the closest point along that line at a given angle from a given point	3	0
dmg	Direction of maximum gradient	0	0
doc	Read the documentation file describing one of QT's M-functions	1	0
dor	DOLP filter, based on rectangle-averaging filters	4	0
dpa	Draw the principal axis (i.e., axis of minimum second moment)	0	3
dps	Picture shift with wraparound Shift parameters are set using the cursor	0	2
drd	set the intensities of all pixels within a defined rectangle to a given value	5	0
dri	Dilate with a rectangular structuring element	2	0
drp	Dominant Radon projection	0	0
dsc	Draw a white disc of radius r centred at a given point [x,y]	3	0

dsq	Dilate so that an isolated white point becomes a square	1	0
dsz	Dynamic size thresholding (Works with blob sizes $<$ 256 pixels)	0	1
dth	Slightly modified version of cur, omitting colorbar generator	0	3
dtx	Toggle the text-display switch	0	0
eab	Euler number for all blobs	0	1
ecc	Exchange colour channels	3	0
ecy	Eccentricity – ratio of and the lengths of the major and minor axes	0	2
edd	Morphological edge detectors	1	0
edg	Set the image border to a predefined level	3	0
edx	Functions lbx and rbx are applied Then, the results are merged using mni	0	0
emn	Extended-minima transform	1	0
emx	Extended-maxima transform	1	0
enc	Contrast enhancement	0	0
epl	Find the extremal white points along a given line	4	4
eqd	Equivalent diameters of all blobs in a binary image	0	1
eri	Erode with a rectangular structuring element	2	0
erl	Erode with an L-shaped structuring element	2	0
ero	Erode with one of four pre-defined structuring elements	2	0
ersf	Repeated execution of a remote script file, with a specified time limit	3	0
esr	Evaluate a specified command sequence Save results for display via the web	2	2
eul	Euler number for 4- or 8-connected blobs	1	1
evo	Grab the latest image saved by the EvoCam image capture software from the iSight camera	1	0
exm	Extremal points in a binary image	1	4
exr	Exclusive OR	0	0
ext	Extent – proportion of pixels in minimum area rectangle that are also in blob	0	1
fap	Set pixels at a list of given addresses to intensity values defined by list z	3	0
far	Filled area – no of white pixels after holes have been filled	0	1
fbl	Fit one blob onto another	0	1
fcc	Freeman or chain code the edge of a solid blob	0	3
fcd	Draw a circle intersecting three given points	6	3
fce	Fit a circle to intersect three edge points found by scanning horizontally	3	3
fcf	Fixed colour filter	0	0
fci	Read Current Image (see *sci*)	0	0
fcl	Fit a circle using the least-squares fit	0	3
fib	Fill isolated black pixels	0	0
fil	Image filtering with one of MatLab’s predefined image filters	1	0
fish	Read standard image: fish	1	3
fll	Fill isolated black pixels	1	0
fov	Field-of view-calculator	0	0

fsl	Find maximum length straight line using the Hough transform	0	0
fsz	Display image at full size Rescale it, if it is too big to fit on screen	0	0
ftd	Fourier transform (direct)	1	0
ftm	Fourier transform (log magnitude)	1	0
fwp	Furthest white point from a given point	2	3
gap	Get position coordinates and intensity values for all non-black pixels	0	3
gbd	General binary dilation	1	0
gbe	General binary erosion	1	0
gbi	Generate a new black image of a given size	3	0
gcc	Gamma adjustment applied to the RGB channels separately	1	0
gear	Read standard image: gear	0	0
gft	Grass-fire transform	1	0
girls	Read standard image: three girls (colour)	0	0
gis	Grab and save a series of images from a webcam given its URL	4	0
git	General intensity transform using a look-up table	1	0
glass	Read standard image: glass bottle	0	0
glc	General linear convolution operator	1	2
gli	Lower and upper intensity limits	0	2
gma	Gamma adjustment	1	0
go	Initialise QT	0	0
gpt	Find the intensity at a given point	2	1
gra	Simple (cross) edge detector	0	0
grb	Grab an image from one of the remote cameras	2	0
grd	Grid pattern	1	1
grip1	Demonstration: Draw robot fingerprint given size and orientation parameters	6	0
grip2	Demonstration: Calculate grasping parameters for a 2-finger gripper	6	3
grip3	Demonstration: Calculate grasping parameters with round fingers	1	4
grip4	Demonstration: Calculate grasping parameters for a suction or magnetic gripper	0	4
grp	Calculate and display gripping points for 2-finger robot gripper	1	4
grs	Calculate and display gripping point for suction or magnetic robot gripper	0	4
gry	Convert a colour image to grey	0	0
gsi	Get image from the stack simply by clicking on the Image Stack display	0	0
gsn	Add Gaussian noise to an image for testing purposes	1	0
gwc	grab web image, retaining colour information	1	0
gwe	Get a sample colour image from the web	1	0
gwi	Get image from a given web site	1	0
gwp	Get window position	1	4
hbt	Local averaging along the Hilbert curve	2	0
hbt1	Maximum intensity along the Hilbert curve	2	0

hbt2	Minimum intensity along the Hilbert curve	2	0
hcd	Harris corner detector	3	2
hcf	Programmable colour filter for detecting variation in hue	1	0
hcp	Plot the cumulative intensity histogram. Display only; neither image is altered	0	0
heq	Equalise the intensity histogram and plot the mapping function	0	0
hgm	Plot the intensity histogram Display only; neither image is altered	0	0
hgr	Horizontal gradient	0	0
hil	Highlight pixels whose intensity is within a given range	3	0
hin	Halve all intensities	0	0
hlp	Help	1	1
hpf	High-pass filter	0	0
hpi	Plot the intensity histogram. Display only; neither image is altered	0	0
hpk	Find and draw the peaks in the Hough transform	1	1
hqx	Apply histogram equalisation mapping function for Alternate Image to Current Image	0	0
hs4	Hysteresis smoothing applied four times: right, left, up, down. Results are added	1	0
hsb	Scan horizontally along a given row until a bright pixel is found	2	2
hsd	Hyteresis smoothing then subtract the original	1	0
hsi	Separate the HSI planes	1	0
hsp	Render the HS plane in glorious colour	0	0
hsr	Hue-saturation reference chart	0	0
hub	Visit the author’s main web page. Upgrades to QT are held here	0	0
hue	Extract hue channel from a colour image	0	0
huf	Hough Transform	3	0
hys	Hysteresis smoothing	1	0
hzs	Scan horizontally along a given set of rows	3	2
icc	Correlation coefficient between the Current and Alternate Images	0	0
idn	Display the Current and Alternate Images immediately	0	0
ihm	Intensity histogram maximum	1	2
imf	Plot scattergram of A(i,j) versus C(i,j), for all (i,j)	0	0
img	View the standard QT images held on the author’s web site	0	0
imn	H-minima transform	1	0
imx	H-maxima transform	1	0
ine	Find pixels that are equal or very nearly equal in intensity	1	1
iob	Isolate that blob which overlaps a given point	3	0
ipc	Intensity profile	1	0
ipl	Intensity profile along a line	4	0
ipp	Find 100*a centile intensity	1	1
ipr	Intensity profile along a selected row	1	0
irt	Inverse Radon Transform	0	0

isight	Read an image obtained from the iSight camera and iStill software (Macintosh only)	1	0
iso	Plot isophotes in a new figure	0	0
itl	Find the point of intersection of two lines, each defined by two points	8	2
itx	Insert text in an image	1	0
ity	Extract intensity channel from a colour image	0	0
jas	Find a skeleton junction and the area surrounding it	3	1
jnt	Find the joints on a match-stick figure, produced by thn or ske	0	0
joy	Play one of several pre-recorded sounds	0	0
kbb	Keep the blob of rank size a and those blobs larger than it	1	0
kgp	Keep all blobs whose area is greater than b times the area of ath largest blob	2	0
kgr	Keep all b-connected blobs (b = 4 or 8) that contain at least a pixels	1	0
lak	Find the lakes in a blob	0	0
las	Find a skeleton limb-end and the area surrounding it	3	1
lbp	Find the left-bottom-most white (i.e., non-black) pixel	0	2
lci	Linear combination (weighted sum) of the Current and Alternate Images	2	0
led	Laplacian of Gaussian edge detector	1	1
lenna	Read standard image: Lenna (History of Lenna: http://en.wikipedia.org/wiki/Lenna)	0	0
ley	Local entropy of the intensities within a rectangle of given size	2	0
lhq	Local histogram equalisation	2	0
lic	Largest inscribed circle	0	3
linescan	Read standard image: derived from a line-scan camera	0	0
lln	Longest line found using the Hough Transform	3	0
lmb	Find the limb ends on a match-stick figure, produced by thn or ske	0	0
lmi	Centroid and orientation of the axis of minimum second moment, single blob	0	1
lmi_all	As *lmi*, calculate from multiple blobs	0	1
lnb	Largest neighbour in each neighbourhood of given size	1	0
lpc	Laplacian	0	0
lpf	Local averaging low-pass filter, 3*3 kernel	0	0
lrt	Invert the horizontal axis	0	0
lrx	L-R row maximum(rox) and R-L row maximum Results are then merged with mni	0	0
ls_cone	Read standard image: loud-speaker cone with loose wires	0	1
lsc	Linear scan; sample the intensity along a line	5	1
lsd	Local standard deviation of the intensities within a rectangle of given size	2	0
lsi	Variable frequency sine-wave generator	4	0
ltp	Find the left-top-most white (i.e., non-black) pixel	0	2
ltw	Left-top white point	0	2
lvm	Browse database of lighting-viewing methods on author’s web site	1	0

lvm_search	Search the Catalogue of Lighting-Viewing Methods for a given string	1	0
lvs	Search the LVM Catalogue for a set of terms	1	0
mab	Take multiple measurements (13) on all blobs	0	1
maj	Majority-vote edge smoothing on a binary image	1	0
mar	Minimum-area enclosing rectangle	0	4
mbc	Minimum bounding circle	1	3
mch	Find the mid-point of each vertical chord	0	0
mci	Mask a coloured image	0	0
mcn	Multiply all intensities by a constant	0	0
mdf	Median filter based on a window of given size	2	0
mdp	Most distant edge points	0	4
mgw	Magic wand operator	1	0
mic	Apply lic recursively; fit many circular discs inside a blob	1	3
mnc	Minimum of the RGB channels of two colour images separately	0	0
mni	Point by point minimum of current and alternate intensities	0	0
moment	Calculate the [p,q]-th moment for small integers p and q	2	4
mrc	Crop image by finding the smallest enclosing rectangle	0	0
mst	Thin white lines, to create a "match-stick" figure	1	0
mul	Multiply two images	0	0
mxi	Point by point maximum of current and alternate intensities	0	0
mxs	Maximum and minimum of several images derived from a given webcam	2	0
nargs	Utility function: No. of arguments and data I/O requirements for given function	0	6
ndo	Numerate distinct binary objects	0	0
neg	Negate	0	0
news	Read the latest news about QT from the author's web site	0	0
nlf	Non-linear filters Function is selectable by first parameter	3	0
nlg	Natural logarithm of the intensity of each pixel	0	0
nlp	Non-linear low-pass filter with one of four pre-defined structuring elements	2	0
npo	Reposition binary object at the centre of the image and normalise orientation	0	0
npo_all	As *npo* but calculated from all blobs	0	0
npv	Find number of pattern vectors	0	1
nrf	Neighbourhood intensity range filter	2	0
nwp	Nearest white point to a given point	2	3
nxy	Reposition binary object at the centre of the image	0	2
nxy_all	Reposition centroid of multiple binary objects at the middle of the image	0	2
oab	Orientation parameters of all blobs in a binary image	0	1
opn	Opening with one of four pre-defined structuring elements	2	0
packer	Demonstration: Packs rectangles into an odd-shaped area	1	0

pad	Pad the image by adding a border of defined width and intensity	3	0
panasonic	Grab an image from a Panasonic Lumix camera	1	0
parts	Read standard image: four metal stampings	0	0
pbi	Perpendicular bisector of the line joining two given points	4	4
pcb	Read standard image: printed circuit board	0	0
pcd	Apply given dyadic operator to RGB channels separately	2	0
pcf	Programmable colour filter	0	0
pci	Process each colour channel separately	3	0
ped	Prewitt edge detector	1	1
pencils	Read standard image: pencils (colour)	0	0
pex	Rescale the spatial resolution along both axes, by a given amount	1	0
pft	Fit a polynomial curve to the edge of a binary object	0	0
pgn	Draw a polygon using the cursor	0	0
pgt	Find the intensity at a given point	2	1
phs	Process each H, S and I channels separately	3	0
pis	Process all image files	3	0
pki	Find coordinates and value of peak intensity and plot it	0	3
plan	Read standard image: plan view of an hydraulics component	0	0
plot3d	Plot intensity in the Current Image as a 3D surface	0	0
plt	Intensity profile, overlaid on the Current Image without changing it	1	0
plv	Plot a vector in the Current Image	1	0
plw	Parallelogram warping	1	0
pmf	Process selected frames of a non-compressed colour AVI movie	4	0
pon	Normalise position and orientation, determined from direction of greatest radius	0	3
pop	Pop a file from the image stack	0	0
pout	Read standard image: girl pouting	0	0
prc	Plot intensity profiles along a row and column selected using the cursor	0	2
prolog	Ask SWI-Prolog to satisfy a given goal, returning a list of integers	1	1
psc	Pseudo-colour display of the Current Image	0	0
psf	Process each image in a given folder and construct an AVI movie	2	0
psh	Picture shift, without wrap-around	2	0
psw	Picture shift with wrap-around	2	0
ptc	Convert from polar to Cartesian coordinates	2	0
pth	Threshold so that a defined proportion of the image is white	1	0
push	Push a file onto the image stack	0	0
pxv	Investigate position and intensity of image features interactively, using the cursor	1	0
qt	Initialisation and starting one of the standard interactive environments	0	0
qtp	Add/remove path to QT update folder. Enables/disables updates	1	0
qtr	Evaluate sequence specified by the command file and save results for web	2	0

qts	Operate QT using the image stack (The stack size is set within qtstart)	0	0
qtu	Read a remote file and save it as an M-file	1	0
r	Synonym for rsi	1	0
raf	Local averaging low-pass filter of given kernel size	2	0
rbe	Remove blobs that touch the edge of the Current Image	2	0
rbi	Restore Current and Alternate Images from disc	0	0
rbk	Remove background intensity by opening and then subtracting the original image	1	0
rbp	Find the right-bottom-most white (i.e. non-black) pixel	0	2
rcf	Rank convolution filter	1	0
rci	Read a numbered image file and do not convert from colour to monochrome	1	0
rdc	Relate distance from centroid of a blob to distance around its circumference	0	2
rdf	Rank difference filter	3	0
rdt	Radon transform	0	0
rea	Read an image file	1	0
red	Roberts edge detector	0	0
rei	Read the image saved previously by wri in a TIFF or JPEG file	2	1
rev	Bit reversal for each intensity value; MSB – > LSB, LSB – > MSB, etc	0	0
rgb	Separate RGB colour planes	1	0
rgb2xyz	Convert RGB to XYZ colour coordinates, single data triple	3	3
rice	Read the standard image file showing a few grains of rice	0	0
rii	Read an image created by the iSight camera and sitting on the desk-top	1	0
rim	Read a numbered or named image available formats: TIF, TIFF, or JPG	1	0
rin	Row integrate	0	0
rip	MATLAB process control, interrupt on timer signal	2	0
rkf	Rank filter Find c-th largest intensity value within each a*b window	3	0
rlc	Run-length code along columns	0	0
rlr	Run-length code along rows	0	0
rmn	Find regional minima	0	0
rmv	Remove interior pixels (Binary edge)	0	0
rmx	Find regional maxima	0	0
rni	Read a numbered RGB or gray image file, with file-name extension "jpg" or "tif"	1	0
road	Read standard image: road scene	1	0
rot	Rotate the image by a given amount (a), with no cropping	1	0
rox	Row maximum	0	0
rpc	"Reversible" pseudo-colour mapping using standard image number #999 as LUT	0	0
rsb	Make all rows same as bottom row of the image	0	0
rsf	Resize the image display windows (labelled Figure 1 and Figure 2)	0	0

rsi	Read numbered stack image	0	0
rsk	Fill the stack with randomly chosen images	0	0
rsz	Resize image/adjust aspect ratio	2	2
rti	Fill the image stack with popular images	0	0
rtp	Find the right-top-most white (i.e., non-black) pixel	0	2
rtx	Convert text file to speech.	1	0
rwo	Repeated web operation	2	0
rwp	Read one of the standard images from the author's web site	2	0
rxm	Relaxed median filter	2	0
sai	Sum all intensities	0	1
sat	Extract saturation channel from a colour image	0	0
saturn	Read standard image: Saturn and its moons	0	0
say	Convert a text string to speech (Macintosh only)	1	0
sbc	Select pixels based on the number of white 8-neighbours of each white pixel	1	0
sbi	Save both Current and Alternate Images on disc	0	0
sca	Bit set	1	0
scc	Scan around a circular path	3	1
sci	Set image to a constant intensity	1	0
scissors	Read standard image: scissors (JPEG not true binary)	0	0
scp	Draw polygon in colour image and plot scattergram of RGB values in 3D space	0	0
scr	Scramble the image by sampling along the Hilbert curve	1	0
scv	Set RGB colour values in the Current Image at a defined address	2	3
sdm	Grey-level spatial dependency matrix	2	0
seb	Shade the edge of a blob according to the distance around the circumference	0	3
sed	Sobel edge detector and variations thereon	1	0
seg	Segment the input image	2	1
sgt	Initiate a given operation. After a given time, the user is given the option to terminate execution.	2	0
shf	Area, perimeter and shape factor	0	3
shh	Shift the Current Image by a given number of rows	1	0
shk	Shrink each white region to a point or to loops, if it has lakes	1	0
shp	Vertical gradient	0	0
shr	Shear warping	1	0
shs	Plot scattergram of saturation versus hue	0	0
sia	Save all image files in the named folder to the folder holding QT's standard images (BGB_MatLab_Images)	1	0
side	Read standard image: side view of an hydraulics component	0	0
sim	Read one of the standard images. The user selects which one by clicking on a menu.	0	0
sir	Set the intensities of all pixels within a defined rectangle to a given value	5	0

sit	Sigmoid function intensity mapping	0	0
siv	Get intensity value in the Current Image at a defined address	3	1
ske	Skeletonisation	0	0
slf	Normalise the orientation of the strongest linear feature	0	1
sly	Solidity – proportion of pixels in convex hull that are also in blob	0	1
sme	Edge smoothing: dog-on-a-lead, edge-following algorithm	1	0
sml	Keep the ath smallest blob	1	0
snb	Smallest neighbour in each square neighbourhood of a given size	1	0
sob	Select one blob with the cursor	0	2
sol	Draw line identified by a single spot in the RT (Radon transform) image	4	0
sony	Grab an image from a Sony Cybershot camera	1	0
speaker	Use the speech synthesiser to play a pre-recorded message (Macintosh only)	1	0
spi	Timed interruption of processing	2	0
spn	Add salt and pepper noise to an image for testing purposes	1	0
spr	Shrink each white region to a point or to N loops, if it has a holes	1	0
sqi	Pad the image to make it square	1	0
sqr	Square of all intensities	0	0
sqt	Square-root of all intensities	0	0
srgb	2D colour scattergram	2	0
ssk	See the images on the image stack	0	0
startup	Initialise the stand-alone version of QT. (MATLAB automatically executes the macro startup.m on start-up.)	0	0
stp	Generate a series of intensity steps	0	0
str	Draw a straight line between the overall centroid and the centroid of each blob	0	0
street	Read standard image: street scene (Times Square, New York City, USA)	0	0
sub	Subtract two images	0	0
suc	Subtract the RGB channels of two colour images separately	0	0
swi	Switch images – interchange the Current and Alternate Images	0	0
swipl	Ask SWI-Prolog to satisfy a given goal. This expected to return a list of integers.	1	1
tav	Threshold at some defined point within the band of occupied intensity values	1	0
tbt	Invert vertical axis of the image	0	0
tbx	T-B column maximum (cox) and B-T column maximum Results are merged with mni	0	0
tha	Threshold at some defined point within the band of occupied intensity values	1	0
thb	Threshold at some defined point within the band of intensity values defined by two centiles	3	0
thc	Threshold at the average intensity plus some defined constant intensity	1	0

thd	Threshold at the most popular intensity value plus some defined constant	1	0
the	Threshold to show those pixels that have the lowest intensity	0	0
thf	Threshold to show those pixels that have the highest intensity	0	0
thg	Threshold to show pixels at the popular intensity value	0	0
thh	Threshold using Otsu's method for choosing the parameter	0	2
thn	Thin white lines	1	0
thr	Threshold	2	0
tim	Operation timer	1	1
tkn	Thicken white areas but do not join them	1	0
tlp	Find the top-left-most white (i.e., non-black) pixel	0	2
tlw	Left-top white point	4	4
tpt	Top-hat morphological filter	1	0
trp	Find the top-right-most white (i.e., non-black) pixel	0	2
tsk	Get a file from the top of the image stack, or	0	0
tur	Rotate the image by a given amount	1	0
txt	Superimpose text temporarily on the Current Image	2	0
udc	Draw circle through two points defined by the user	1	0
udr	Draw rectangle through two points defined by the user	1	0
uds	Move up/down the image stack	0	0
ult	Ultimate erosion of a binary image	0	0
uns	Unsharp masking	0	0
usc	Reverse the scrambling perfomed by sampling along the Hilbert curve	1	0
use	User defines a structuring element using the cursor	1	1
vfs	Variable frequency sine-wave generator for testing 1-dimensional filters	1	0
vgr	Vertical gradient	0	0
vial	Read standard image: glass vial (dark-field illumination)	0	0
vpl	Draw a straight line given two points	5	0
w	Synonym for wis	1	0
wgx	Intensity wedge varying along the X-axis	0	0
wgy	Intensity wedge varying along the Y-axis	0	0
wires	Read standard image: coil with loose wires	0	0
wis	Write the Current Image to the image stack	0	0
wnf	Wiener filter	2	0
wri	Save the Current Image in a TIFF or JPEG file	2	1
wrw	Save the Current and Alternate Images (JPEG files), ready for web	1	0
wsh	Watershed	1	0
wti	Save the Current Image in a TIFF or JPEG timed file	1	1
wtx	Create a new text file/add text to exiting file. Prepare files for "rtx"	2	0
xcf	Cross correlation applied to two images.	1	2
xray	Read the standard image x-ray of a mains plug	0	0
xyl	Minimum-area enclosing rectangle for a single blob	0	4

xyz	Reserved for testing QT software		
xyz2rgb	Convert XYZ to RGB colour coordinates	3	3
yxt	Interchange the vertical and horizontal axes	0	0
z	Switch images; interchange the Current and Alternate Images	0	0
zed	Zero-crossing edge detector	1	1
zer	Set all pixels to black	0	0

Controlling the Velleman K8055 Interface Board (WINDOWS only, See Appendix I)			
		Arguments	
Name	Function	In	Out
vel_start	Start	1	0
vel_close	Close	0	0
vel_set_dig_out	Set given digital output to defined level	2	0
vel_clear_dig_out	Clear given digital output	1	0
vel_read_digital	Read given digital input	1	1
vel_set_analog_out	Set given analogue output to defined volatage	2	0
vel_read_analog	Read given analogue input	1	1
vel_test	Test the board	1	0
vel_wait_dig_input	Wait for given digital input to go "high"	1	1
vel_wait_any_dig_input	Wait for any digital input to go "high"	0	2
vel_wait_analog_input	Wait for given analogue input to go above defined voltage	2	2

Commercial Interactive Image Processing Products

Several commercial products have been based on the same paradigm as QT's progenitor, SUSIE. In addition, several similar systems have been devised for in-house use in companies and universities.

Intelligent Camera, previously a product of Image Industries/Ltd, now part of Cognex Ltd., Epsom, Surrey, England, U.K.

SuperVision, interactive image processing software, previously a product of Image Industries/Ltd, now part of Cognex Ltd., Epsom, Surrey, England, U.K.

VCS (Vision Control System), interactive image processing software, previously a product of Vision Dynamics Ltd., Hemel Hempstead, Herts, U.K.

Autoview Viking, previously a product of British Robotic Systems Ltd., London, U.K.

System 7, previously a product of The 3M Company, St.Paul Minnesota

MATLAB

QT was programmed using the following software:

Image Processing Toolbox, MATLAB, Mathworks, Inc, URL http://www.mathworks.com/, accessed 26th November 2009.

Further Reading

1. Batchelor BG (1979) Interactive image analysis as a prototyping tool for industrial inspection. IEE PROC-E 2(2):61–69
2. Batchelor BG (1986) Merging the autoview image processing language with prolog. Image Vision Comput 4(4):189–196
3. Batchelor BG (1991) Intelligent image processing in prolog. Springer, Berlin. ISBN 0-540-19647-1
4. Batchelor BG, Hack R (1995) Robot vision system programmed in Prolog. In: Proceedings of the Conference on Machine Vision Applications, Architectures and Systems IV, Philadelphia, October 1995, SPIE, Bellingham, vol 2597, ISBN 0-8194-1961-3, pp 239–252
5. Batchelor BG, Waltz FM (1993) Interactive image processing. Springer, New York. ISBN 3-540-19814-8
6. Batchelor BG, Waltz FM (2001) Intelligent machine vision. Springer, New York. ISBN 3-540-76224-8
7. Batchelor BG, Whelan PF (1997) Intelligentvision systems for industry. Springer, London. ISBN 3-540-19969 1
8. Batchelor BG, Mott DH, Page GJ, Upcott DN (1982) The Autoview interactive image processing facility. In: Jones NB (ed) Digital signal processing. Peter Peregrinus, London, pp 319–351. ISBN 0 906048 91 5
9. Batchelor BG, Daley MW, Griffiths EC (1994) Hardware and software for prototyping industrial vision systems. In: Proceedings of the Conference on Machine Vision Applications, Architectures and Systems Integration III, Boston, October 1994, pub. SPIE, Bellingham, vol 2347, ISBN 0-8194-1682-7, pp 189–197
10. Batchelor BG, Griffiths EC, Hack R, Jones AC (1996) Multi-media extensions to prototyping software for machine vision. In: Proceedings of the Conference on Machine Vision Applications, Architectures and Systems VI, Boston, November 1996, SPIE, Bellingham, vol 2908, pp 259–273
11. Batchelor BG, Hack R, Jones AC (1996) Prolog-based prototyping software for machine vision. In: Proceedings of the Conference on Machine Vision Applications, Architectures and Systems VI, Boston, November 1996, SPIE, Bellingham, vol 2908, pp 234–248
12. Batchelor BG, Daley MW, Hitchell RJ, Hunter GJ, Jones GE, Karantalis G (1999) Remotely operated prototyping environment for automated visual inspection. In: Proceedings of the IMVIP99 – Irish Machine Vision and Image Processing Conference 1999, Dublin City University, Dublin, September 1999, pub Irish Pattern Recognition and Classification Society, ISBN 1 872 327 22 2, pp 300–320
13. Caton SJ, Rana OF, Batchelor BG (2009) Distributed image processing over an adaptive campus grid. Concurrency Comput-Prac Ex 21(3):321–336
14. Chatburn LT, Daley MW, Batchelor BG (2004) Myriad: a framework for distributed and networked vision systems, two- and three-dimensional vision systems for inspection, control, and metrology. In: Batchelor BG, Hügli H (eds) Proceedings of the SPIE, vol 5265, pp 98–109
15. Jones AC, Hack R, Batchelor BG (1996) Software organisation for a Prolog-based prototyping system for machine vision. In: Proceedings of the Conferecne on Application of Digital Image Processing XIX, Denver, August 1996, pub. SPIE, Bellingham, vol 2847, ISBN 0-8194-2235-5, pp 16–27
16. Karantalis G, Batchelor BG (1998) Prototyping machine vision software on the World Wide Web. In: Proceedings of the SPIE Conference on Machine Vision Systems for Inspection and Metrology VII, Boston, November 1998, vol 3521, pp 3014–31, ISBN 0-8194-2982-1
17. Suau P, Pujol M, Rizo R, Caton SJ, Rana OF, Batchelor BG, Pujol F (2005) Agent-based recognition of facial expressions. In: Proceedings of the 4th International Joint Conference on Autonomous Agents and Multi-agent Systems, Utrecht, 2005, ISBN: 1-59593-093-0, pp 161–162
18. Whelan PF, Batchelor BG, Lewis MRF, Hack R (1998) Machine vision design and training aids on the World Wide Web. Vision 14(2):1–6

22 NeatVision: Development Environment for Machine Vision Engineers

Paul F. Whelan · Robert Sadleir · Ovidiu Ghita
Dublin City University, Glasnevin, Dublin, Ireland

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_22,

Abstract: This chapter will detail a free image analysis and software development environment for machine vision engineers. The environment provides high-level access to a wide range of image manipulation, processing and analysis algorithms (over 300 to date) through a well-defined and easy to use graphical interface. Users can extend the core library using the developer's interface, a plug-in, which features automatic source code generation, compilation with full error feedback, and dynamic algorithm updates. The chapter will also discuss key features associated with the environment and outline the advantages in adopting such a system for machine vision application development. (http://www.cipa.dcu.ie/)

22.1 Introduction

For many novices to the world of machine vision, the development of automated vision solutions may seem a relatively easy task, as it only requires a computer to understand basic elements such as shape, colour and texture. Of course this is not the case. Extracting useful information from images is a difficult task and as such requires a flexible machine vision application development environment. The design of machine vision systems requires a broad spectrum of techniques and disciplines. These include electronic engineering (hardware and software design), engineering mathematics, physics (optics and lighting), mechanical engineering (since industrial vision systems deal with a mainly mechanical world) as well as the system engineering aspects of developing reliable industrial systems. In this chapter, we will focus on one aspect of the machine vision design cycle, namely the algorithm development environment (referred to as NeatVision [1]). It aims to provide novice and experienced machine vision engineers with access to a multi-platform (realised through the use of Java) visual programming development system.

Java is an interpreted programming language, and as such applications written in Java are not executed directly on the host computer; instead, these applications are interpreted by the Java Virtual Machine. As a result, Java programs generally run slower than native compiled programs written in languages such as C or C++. This performance issue is constantly being addressed and a number of improvements have been made since Java was first released. For example, just-in-time compilers significantly improved the performance of Java applications by removing the need to reinterpret already executed code while the introduction of HotSpot technology further enhances application performance by optimising garbage collection and improving memory management. With the recent release of the Java Standard Edition Version 6 Update 22 the performance of Java is approaching that of native programming languages.

The NeatVision environment provides an intuitive interface which is achieved using a drag and drop block diagram approach. Each image-processing operation is represented by a graphical block with inputs and outputs that can be interconnected, edited and deleted as required. NeatVision (Version 2.1) is available free of charge and can be downloaded directly via the Internet www.NeatVision.com.

22.1.1 Standard Installation

The requirements for the standard NeatVision installation are:

- JRE 1.4.X – The J2SE Java Runtime Environment (JRE) allows end-users to run Java applications. (e.g., *j2re-1_4_2_09-windows-i586-p.exe*)

- JAI 1.X – Java Advanced Imaging (JAI) API. (e.g., *jai-1_1_X-lib-windows-i586-jre.exe*)
- NeatVision Standard Edition (*neatvision.jar*). NeatVision is distributed as a ".jar" file. The contents of this file should not be extracted; any attempt to do this will cause NeatVision to cease functioning

Please insure that the JAI is placed in the same path as the JRE. This only becomes an issue if you have multiple versions of Java on your machine (the default is that JAI will place itself in the most recent version of the Java). Assuming the JRE has been installed in `C:\Program Files \ Java\j2re1.4.2_09` and the NeatVision jar file is in `D:\NV`, then the following single line command (The NeatVision argument **must have** a capital "N" and "V") will enable NeatVision to run from `D:\NV`.

```
"C:\Program Files\Java\j2re1.4.2_09\bin\java.exe" -classpath D:\NV\
neatvision.jar NeatVision
```

22.2 NeatVision: An Interactive Development Environment

NeatVision is just one example of a visual-programming development environment for machine vision [2]; other notable examples include commercial programs such as Khoros [3] and WiT [4]. Visual programming involves defining variables, and specifying operations, which are to be performed on these variables, and their derivatives in order to perform a specific task. This is achieved by creating a structured flow of data using branching, looping and conditional processing. Traditionally, computer programs have been written using textual programming languages. These programs can process data in a complex fashion; unfortunately, the data paths and the overall structure of the program cannot be easily identified from the textual description. This can make it very difficult to appreciate the relationship between the source code and the functionality, which it represents. Although the programmer specifies the data flow in a visual program, the order in which the components execute is defined by the availability of data. Conditional processing concepts are supported in the visual domain by using dedicated flow control components. The main disadvantages of existing visual programming environments includes their cost, lack of cross platform support, and the fact that they tend to be focused on image processing rather than image analysis applications (the latter must be considered a key element of any practical machine vision application).

Text-based programming languages such as MATLAB [5] can be a powerful alternative to visual programming. In addition to the disadvantages outlined with respect to the visual programming languages, text-based approaches require the user to have a higher level of programming skills when compared to visual programming environments. Text-based interactive environments are generally more suitable to experienced users; in fact, experienced users can become frustrated by the visual programming environment as complex programs can take longer to develop (Note: we have recently developed a MATLAB compatible *VSG Image Processing & Analysis Toolbox* toolbox [6] that replicates NeatVision's functionality to allow MATLAB users build machine vision solutions). Hence our aim is to produce a suitable environment for those new to machine vision while retaining the flexibility of program design for the more experienced users. Based on our review of exiting text and visual programming based on machine vision development environments, the key criteria necessary are outlined below:

- Multi-platform: The development environment must be able to run on a wide range of computer platforms.
- Focused on machine vision engineering: The environment should contain a wide range of image processing and analysis techniques necessary to implement practical machine vision engineering applications.
- Easy to use: It should allow users to concentrate on the design of machine vision solutions, as opposed to emphasising the programming task.
- Upgradeable: The environment must contain a mechanism to allow users to develop custom vision modules.

A visual program can be created by defining input data using the input components, then implementing the desired algorithm using the processing and flow control components. The data flow is specified by creating interconnections between the components. The program can be completed by adding output components to view the data resulting from the algorithm execution. Details on the design of the NeatVision development environment appear elsewhere [7].

22.3 NeatVision's Graphical User Interface (GUI)

The NeatVision GUI (➊ *Fig. 22.1*) consists primarily of a workspace where the processing blocks reside. The processing blocks represent the functionality available to the user. Support is provided for the positioning of blocks around the workspace, the creation and registration of interconnections between blocks. The lines connecting each block represent the path of the data through the system. Some of the blocks can generate child windows, which can be used for viewing outputs, setting parameters or selecting areas of interest from an image. If each block is thought of as a function, then the application can be thought of as a visual programming language. The inputs to each block correspond to the arguments of a function and the outputs from a block correspond to the return values. The advantage of this approach is that a block can return more than one value without the added complexity of using C-style pointers. In addition, the path of data through a visual program can be dictated using special flow control components. A visual program can range in complexity from three components upwards and is limited only by the availability of free memory.

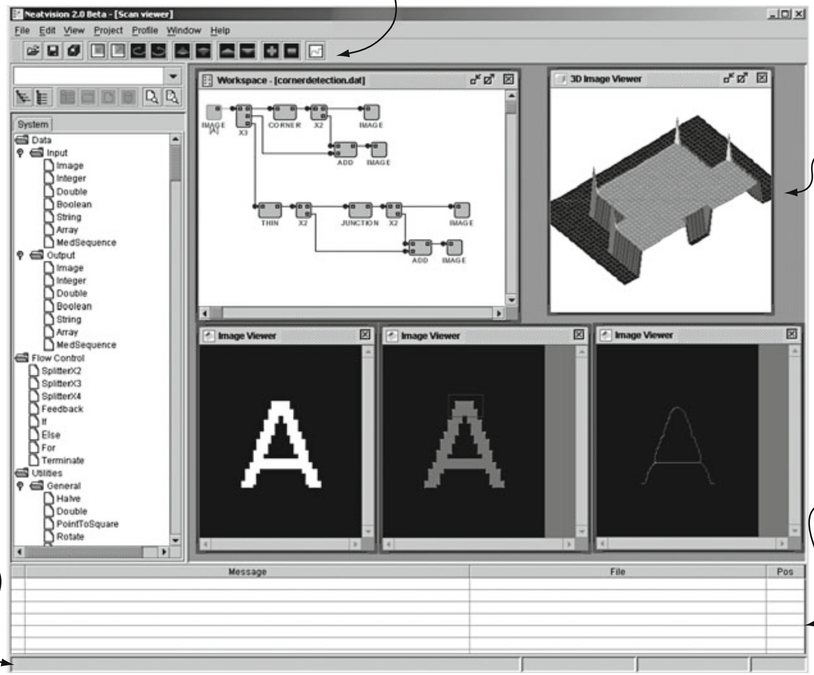
As each block is processed, it is highlighted (in green) to illustrate that it is active. This allows users to see the relevant speeds of parallel data streams within a visual program. This can help identify potential processing bottlenecks within the workspace allowing for a more efficient balanced design. The colour coding of the blocks data connection type and its status also aids in the design process. To aid user operation, each data connection has two colour coded properties, namely the block *data type* and *connection status.* NeatVision currently supports eight data types, i.e., Image (red), Integer/Array data (green), Double precision Floating point data (blue), Boolean data (orange), String data (pink), Fourier data (light blue), Coordinate data (purple) and Undefined data (black). The other connection property relates to its status. There are three main states for a connection, *connected (green), disconnected (red)* and *disconnected but using default value (orange)*.

This approach provides a fast and simple alternative to conventional text-based programming, while still providing much of the power and flexibility. The visual workspace can be compiled and executed as with a conventional programming language. Errors and warnings are

The search utility
due to the vast range of visual components provided by neatvision a search function is provided. Just type the name in here and the relevant component will appear in the workspace.

The controls
the neatvision controls consist of the toolbar and the menubar, these controls adapt to suit the application that is currently selected in the virtual desktop virtual desktop.

The component explorer
The Neatvision component explorer provides access to the currently available components. Components can be dragged directly from the explorer into a visual workspace.



The virtual desktop
any appliaction window such as the visual workspace or the text editor will reside within the virtual desktop of the neatvision application.

The statusbar
The statusbar is used to provide general information for instance compilation success and connection information.

The message window
all java and visual compilation errors and warmings are presented in this window. In order to find out the source of any message just doulble click the messae and the relevant error will be highlighted.

Fig. 22.1
Key features of the NeatVision environment

generated depending on the situation. There is currently support for 15 input graphics file formats and 13 output formats. Some of the main formats are listed below (*R* indicates a read-only file format and *RW* indicates a read/write file format).

- BMP (*RW*) Microsoft Windows Bitmap.
- BYT (*RW*) Raw image data, grey scale only with a maximum pixel depth of 8 bits.
- FPX (*R*) Kodak FlashPix is a multiple-resolution, tiled image file format based on JPEG compression.
- GIF (*R*) Graphics Interchange Format (GIF) is a bitmap file format which utilises Lemple-Zev-Welch (LZW) compression.
- JPEG (JPG) (*RW*) Joint Photographic Experts Group (JPEG) file interchange format is a bitmap file utilising JPEG compression.
- PCX (*RW*) PC Paintbrush (PCX) is a bitmap file using either no compression or Run Length Encoding (RLE).
- PNG (*RW*) PNG is a lossless data compression method for images.
- PBM (*RW*) Portable BitMap is a lowest common denominator monochrome file format.
- PGM (*RW*) Portable Greymap Utilities is a non-compressed bitmap format, hence allowing image data to be left intact.
- PPM (*RW*) Portable PixelMap is a lowest common denominator colour image file format.
- RAS (*RW*) Sun Raster Image (RAS) is a bitmap file format using either no compression or RLE.
- RAW (*RW*) Raw image data is similar to the BYT format described earlier except in this case colour image data is also supported.
- TIFF (TIF) (*RW*) Tagged Image File Format is a bitmapped file format using a wide range of compression techniques.

System parameters can be adjusted and the system may be reset and executed again until the desired response is obtained. At any stage, blocks may be added or removed from the system. NeatVision also contains a built-in web browser to allow easy access to online notes and support tools.

22.4 Design Details

NeatVision is designed to work at two levels. The *user* level allows the design of imaging solutions within the visual programming environment using NeatVisions core set of functions. NeatVision (Version 2.1) contains 300 image manipulation, processing and analysis functions, ranging from pixel manipulation to colour image analysis to data visualisation. To aid novice users, a full introductory tutorial and some sample programs can be found on the NeatVision website. A brief description of the main system (Items in italics are only included in the NeatVision advanced edition.) components [8] is given below; see ➋ Sect. 22.8 for additional details:

- Data types: Image, integer, double, string, Boolean, array, medical image sequences
- Flow control: Path splitting, feedback, if (else), for loop (see ➋ *Fig. 22.2*)
- Utilities: Rotation, pixel manipulation, resize, URL control, additive noise generators, region of interest, masking operations
- Arithmetic operators: Add, subtract, multiply, divide, logical operators

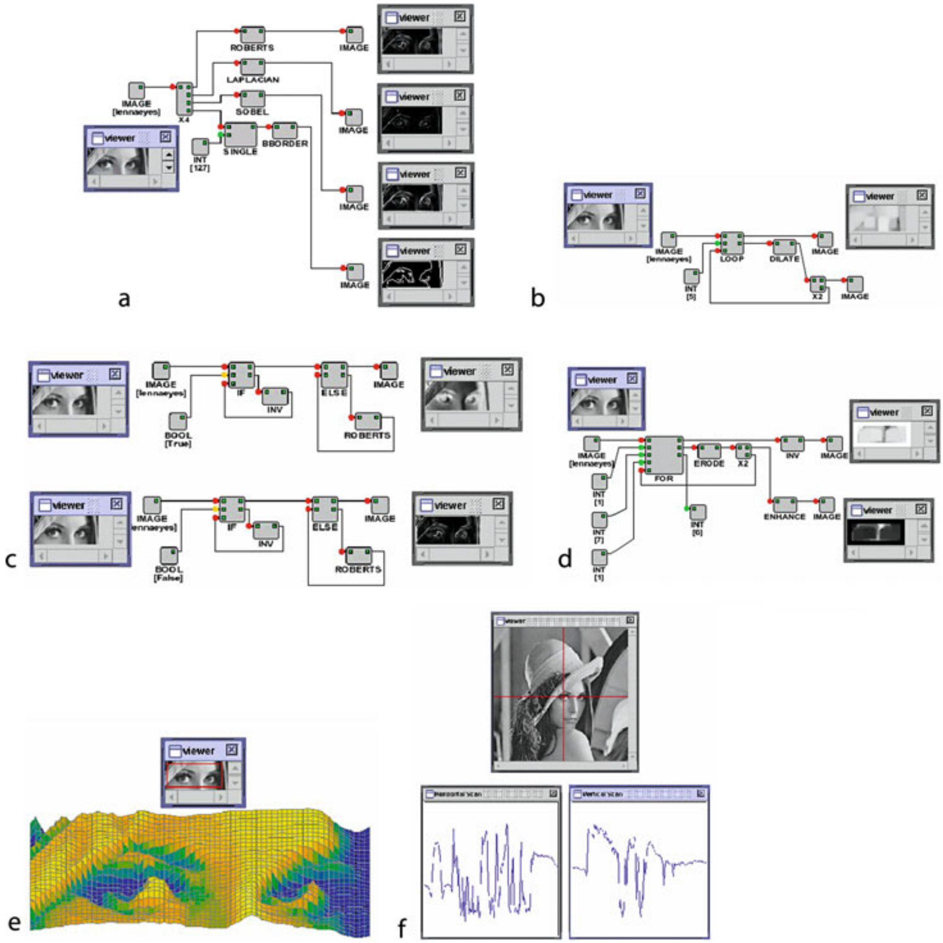


Fig. 22.2
Flow control/graphic utility examples: (a) path splitting, (b) looping feedback, (c) if (else), (d) for loop, (e) 3D viewer, (f) image profiling

- Histogram: General histogram analysis algorithms, local equalisation
- Image Processing: Look-up tables (LUT), threshold, contrast manipulation
- Neighbourhood-based filtering: Lowpass, median, sharpen, DOLPS, convolution, adaptive smoothing (or filtering)
- Edge detection: Roberts, Laplacian, Sobel, zero crossing, Canny
- Edge features: Line/arc fitting, edge labelling and linking
- Analysis: Thinning, binary detection, blob analysis, labelling, shape feature measures, bounding regions, grey-scale corner detectors
- Clustering: K-means (grey scale and colour), unsupervised colour clustering
- Image transforms: Hough (line and circle), Medial Axis, DCT, Cooccurrance, Fourier, distance transforms
- Morphology: Several 2D morphological operators, including erosion, dilation, opening, closing, top-hat, hit-and-miss, watershed

- Colour: Colour space conversion algorithms, RGB, HSI, XYZ, YIQ, Lab
- 3D Volume: 3D operators (thinning, Sobel, threshold, labelling), maximum and average intensity projections, rendering engine (Java and Intel native: wire frame, flat, Gouraud, Phong), *3D to 2D conversion, data scaling, 3D windowing, 3D arithmetic, 3D image processing, 3D labelling, 3D morphological operators, 3D reconstruction, 3D clustering*
- Low Level: Pixel level operators; get pixel value, set pixel value and basic shape generation
- String: String operators, object addition, to upper case and to lower case
- Mathematical: An extensive range of numerical operators and utilities, including constants and random number generation
- JAI Colour: Colour algorithms implemented using JAI (Java Advance Imaging [9]), operators, processing, filters and edge detectors
- OSMIA functions (Wintel native only): NEMA and AIFF image reader, ejection fraction measurement, 2D optical flow (Lucas & Kanasde and Horn & Shcunck courtesy of Barron [10] via the European Union funded OSMIA project [11]), XY normalisation

At the more advanced *developer's* level [12], NeatVision allows experienced application designers to develop and integrate their own functionality through the development and integration of new image-processing/analysis modules.

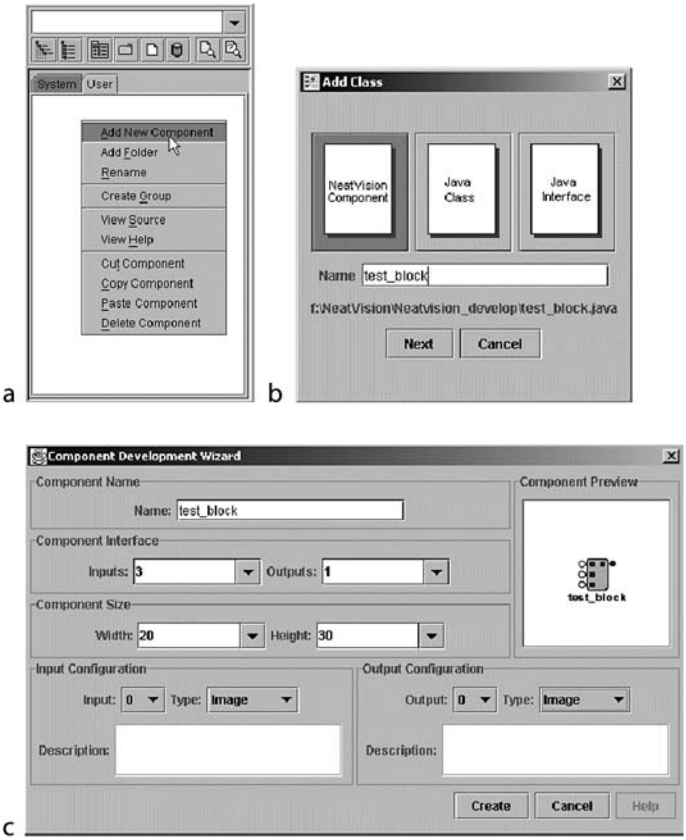
22.5 Developer's Environment

NeatVision was originally designed so that it could be easily extended by building on previously developed algorithms. This feature has been finalised with the release of version 2.1 of the NeatVision visual programming environment. This allows users to:

- Develop new NeatVision components that can ultimately be reused by other NeatVision developers.
- Reuse the core NeatVision components in new user-defined components.
- Submit their component or library of components to wider NeatVision community.

NeatVision development assumes a basic level of familiarity with the Java programming language and the NeatVision developer's plug-in. Additional details on developing for NeatVision [12] and the design concepts behind NeatVision along with detailed explanations of many of the its algorithms are also available [7].

When the developer's interface and the Java Development Kit (JDK) (Installation of the developer plug-in will require you to upgrade the JRE to the JDK. As the JDK compiler is not backward compatible and is frequently modified, we have restricted developer's interface to a specific version of the Java Development Kit – i.e., JDK 1.3.X. To activate the developer's version, you will need the *developer's update file* (*developer.jar*). To install this please close down NeatVision and place *developer.jar* in the classpath. It should be activated the next time you start NeatVision. e.g. `..\java.exe -classpath..\neatvision\neatvision.jar;..\neatvision\developer.jar NeatVision`) (Also be sure to download the latest version of *neatvision.jar*). When the developer's update file is present, a "user" tab appears alongside the "system" tab in the component explorer. The developer can add a new component by right clicking anywhere within the "user" view of the component explorer. After right clicking, a pop-up menu will appear. The "Add New Component" option must be selected from this menu in order to create a new file (➧ *Fig. 22.3*). The user is then queried



Name test_block

f:\NeatVision\Neatvision_develop\test_block.java

Name: test_block

Inputs: 3 Outputs: 1

Width: 20 Height: 30

Input: 0 Type: Image

Description:

Output: 0 Type: Image

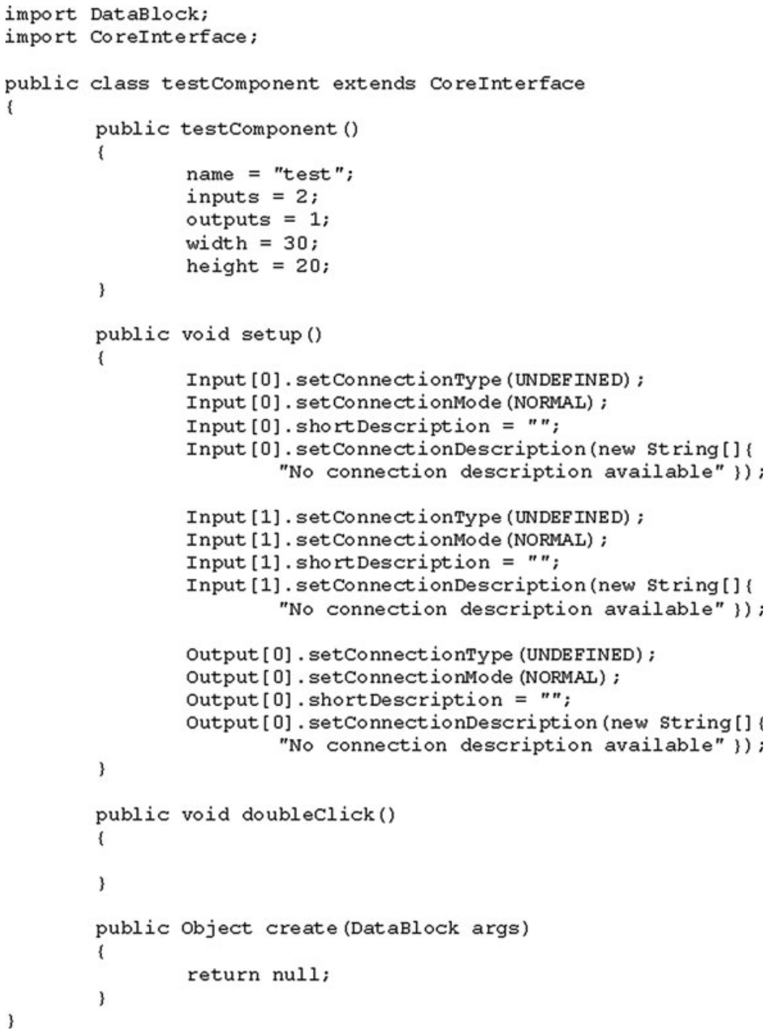
Description:

Fig. 22.3

NeatVision component development wizard. (a) Select *Add New Component* from the pop-up menu of the *user* area of the component explorer. (b) Select component to be added. (c) Define the component skeleton (i.e., the number and type of inputs and outputs for the associated block). This generates the necessary component wrapping code

as to whether they would like to create a NeatVision Component, a Java Class or a Java Interface (Note: a NeatVision Component is just a special type of Java class). A file name for the class or interface must be specified at this point. If a Java Interface or standard Java class is specified, then a text editor window is displayed with the relevant skeleton source code. The developer may edit and compile this code as desired. If a NeatVision component is specified, then a component development wizard is launched.

The wizard allows the developer to specify the visual appearance of the component including width and height in pixels, component name and number of inputs and outputs. The wizard also allows the developer to specify the data type and data description associated with each of the inputs and outputs. Once all of the required information has been entered, the developer need only press the "Create" button to generate to skeleton source code for the desired NeatVision component. The developer can then edit the skeleton source code (➲ *Fig. 22.4*) as required in order to implement the desired component functionality. At any stage, the source code for the component can be compiled. This is achieved by selecting the



```
import DataBlock;
import CoreInterface;

public class testComponent extends CoreInterface
{
        public testComponent()
        {
                name = "test";
                inputs = 2;
                outputs = 1;
                width = 30;
                height = 20;
        }

        public void setup()
        {
                Input[0].setConnectionType(UNDEFINED);
                Input[0].setConnectionMode(NORMAL);
                Input[0].shortDescription = "";
                Input[0].setConnectionDescription(new String[]{
                        "No connection description available" });

                Input[1].setConnectionType(UNDEFINED);
                Input[1].setConnectionMode(NORMAL);
                Input[1].shortDescription = "";
                Input[1].setConnectionDescription(new String[]{
                        "No connection description available" });

                Output[0].setConnectionType(UNDEFINED);
                Output[0].setConnectionMode(NORMAL);
                Output[0].shortDescription = "";
                Output[0].setConnectionDescription(new String[]{
                        "No connection description available" });
        }

        public void doubleClick()
        {

        }

        public Object create(DataBlock args)
        {
                return null;
        }
}
```

Fig. 22.4
The skeleton code for a double input/single output component. Note that the entry point is the `create()` method. This is called whenever the block receives a full complement of input data

relevant compile option from the "Project" menu. Selecting the compile option launches the Java compiler distributed with the JDK, and any errors in the specified file(s) are subsequently listed in the message window at the bottom of the main NeatVision window. For each error a description, file name and line number are provided. The user need only click on an error message to highlight the relevant error in the source code. Once all errors have been corrected, the message "compilation succeeded" is printed in the status bar. Following successful compilation, the block is available for use and can be included in a workspace like any core NeatVision component.

The NeatVision developer's interface extends and complements the visual programming interface by allowing users to develop custom components that can encapsulate the functionality of core NeatVision components, thus extending the already vast NeatVision library of components.

22.5.1 Developing for Reuse

As mentioned previously, the skeleton code for a new NeatVision component is generated using the component development wizard (see ➲ *Fig. 22.4*). In previous versions of NeatVision, the entry point for a component was the `main()` method and the programmer was responsible for interfacing directly with component inputs and outputs to read and write the associated data values. In NeatVision 2.1, the `main()` method is replaced by the `create()` method. This revised approach makes the development of new NeatVision components more straightforward and facilitates component reuse. The programmer is no longer required to interface directly with the component inputs and outputs. Instead, when a component becomes active, NeatVision automatically reads the data values at the inputs to a component and passes the associated data to the `create()` method in the form of a populated `DataBlock` object. Each entry in the `DataBlock` object corresponds to the data at the input with the corresponding index (0, 1, 2, etc. 0 being the topmost input). After the input data has been processed, the results must be stored in a new `DataBlock` object which is returned from the `create()` method upon completion (Note: If only one object is being returned from the `create()` method (i.e. if the block has only one output) then it is not necessary to encapsulate this within a `DataBlock` object. Instead, it can be returned directly and NeatVision will pass the returned object to the single output of the component.) Each entry in the returned `DataBlock` object is then passed to the output with the corresponding index (0, 1, 2, etc. 0 being the topmost output).

22.5.1.1 The `DataBlock` Class

The `DataBlock` class is used to represent the input and output data values associated with a particular NeatVision component. The data associated with a `DataBlock` object is represented as an array of objects. This means that the `DataBlock` class is future proof and will deal with any type of data that may be supported either by the core NeatVision components or any custom components developed by NeatVision users. The specification for the `DataBlock` can be found in the NeatVision developer's guide [12].

22.5.2 How to Reuse

The functionality provided by any of the core NeatVision classes can be called from within custom user-defined classes that are developed using the NeatVision developer's plugin. This is achieved by calling the static `create()` method of the `NeatVision` class.

```
Object NeatVision.create(String class, DataBlock args)
```

The parameters of the `create()` method are a `String` object and `DataBlock` object. The `String` object represents the class name of the desired component, and the `DataBlock` object represents the parameters that will be passed to an off-screen instantiation of the desired component. The `DataBlock` argument must have the same number of entries as the number of inputs connected to the desired component, and each entry must represent the data required by the associated input (0, 1, 2, etc.). The `create()` method then returns a new `DataBlock` object that represents the outputs that were generated after the requested component processed the specified inputs. The `create()` method can also handle up to four arguments that are not encapsulated within a `DataBlock` object, for example:

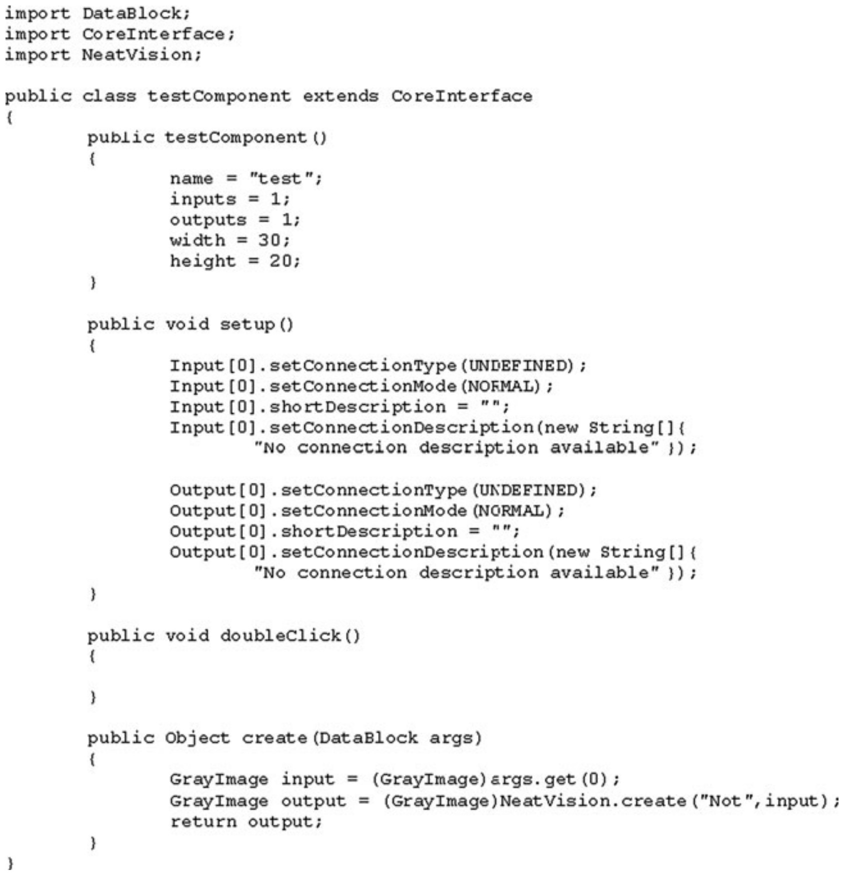
`Object NeatVision.create(String class, Object arg0)`

- Call the create method of the single input component with the specified class name.

`Object NeatVision.create(String class, Object arg0, Object arg1)`

- Call the create method of the dual input component with the specified class name.

All arguments must be represented as objects when using this approach. This means that any primitives must be wrapped before being passed to the `create()` method. Take the integer arguments for the dual threshold operation as an example:



```
import DataBlock;
import CoreInterface;
import NeatVision;

public class testComponent extends CoreInterface
{
        public testComponent()
        {
                name = "test";
                inputs = 1;
                outputs = 1;
                width = 30;
                height = 20;
        }

        public void setup()
        {
                Input[0].setConnectionType(UNDEFINED);
                Input[0].setConnectionMode(NORMAL);
                Input[0].shortDescription = "";
                Input[0].setConnectionDescription(new String[]{
                        "No connection description available" });

                Output[0].setConnectionType(UNDEFINED);
                Output[0].setConnectionMode(NORMAL);
                Output[0].shortDescription = "";
                Output[0].setConnectionDescription(new String[]{
                        "No connection description available" });
        }

        public void doubleClick()
        {

        }

        public Object create(DataBlock args)
        {
                GrayImage input = (GrayImage)args.get(0);
                GrayImage output = (GrayImage)NeatVision.create("Not",input);
                return output;
        }
}
```

Fig. 22.5
A simple example of reuse, calling the `Not` operation from inside a custom user-defined class

```
int hiThresh = 100;
int loThresh = 100;
Integer hiThreshObj = new Integer (hiThresh);
Integer loThreshObj = new Integer (loThresh);
GreyImage output = (GreyImage) NeatVision.create
(``DualThreshold'', input, hiThreshObj, loThreshObj);
```

There are special wrapper classes available for converting all primitive types (`boolean`, `byte`, `short`, `int`, `long`, `double` and `float`) into objects (`Boolean`, `Byte`, `Short`, `Integer`, `Long`, `Double` and `Float`). Objects of these classes can be constructed by simply passing the relevant primitive to the constructor of the relevant class (see int to Integer example above). The static `create()` method of the NeatVision class routes the specified DataBlock object to the `create()` method of the specified class and returns the resulting output DataBlock object.

➤ *Figure 22.5* illustrates a simple example of reuse. This involves calling the `Not` operation from inside a custom user-defined class. ➤ *Figures 22.6* and ➤ *22.7* illustrates the development of the `TestDev` class. This sample program removes boundary regions prior to K-Means clustering. The Canny edge detector is then applied to the original image and keeping only closed structures we find the approximate perimeter of the strong edges. The K-Means and the closed structure overlay images along with the approximate perimeter values are the final block outputs.

22.6 Sample Applications

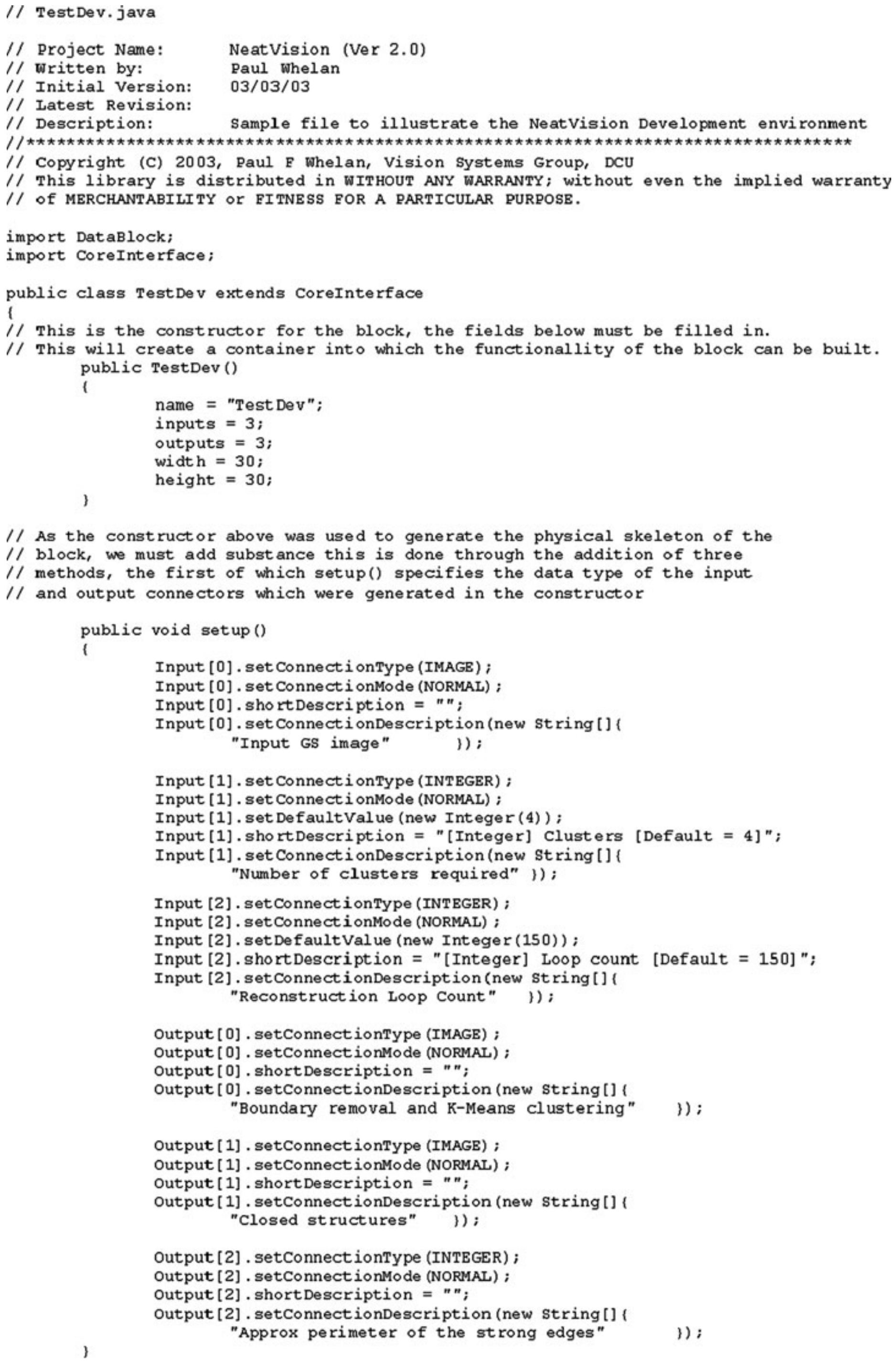
NeatVision provides an image analysis software development environment that can work at several levels. For example, at a relatively low level, individual pixels can be manipulated. Alternatively, NeatVision's built-in functionality can be used to generate solutions to complex machine vision problems. ➤ *Figure 22.8* illustrates how the *reconstruction by dilation* morphological operator can be used to remove or detect objects that are touching the binary image border. ➤ *Figure 22.9* illustrates how NeatVision can be used to isolate defects in the centre panel of a crown bottle top.

22.7 Application Development Case Study

In this section, we will outline the application of NeatVision to a typical machine vision application, namely the characterisation of surface mount components. We aim to develop a robust *NeatVision* program capable of:

- Automatically counting all components fully within the field of view in the image illustrated in ➤ *Fig. 22.10* (i.e. all elements touching the image boundaries must be removed prior to image analysis).
- Isolate and highlight the largest component within the image.
- Find the approximate area of the largest component within the image.

The system must be robust, and with this in mind it should be capable of automatically determining threshold values from the image data.



```
// TestDev.java

// Project Name:        NeatVision (Ver 2.0)
// Written by:          Paul Whelan
// Initial Version:     03/03/03
// Latest Revision:
// Description:         Sample file to illustrate the NeatVision Development environment
//******************************************************************************
// Copyright (C) 2003, Paul F Whelan, Vision Systems Group, DCU
// This library is distributed in WITHOUT ANY WARRANTY; without even the implied warranty
// of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

import DataBlock;
import CoreInterface;

public class TestDev extends CoreInterface
{
// This is the constructor for the block, the fields below must be filled in.
// This will create a container into which the functionallity of the block can be built.
        public TestDev()
        {
                name = "TestDev";
                inputs = 3;
                outputs = 3;
                width = 30;
                height = 30;
        }

// As the constructor above was used to generate the physical skeleton of the
// block, we must add substance this is done through the addition of three
// methods, the first of which setup() specifies the data type of the input
// and output connectors which were generated in the constructor

        public void setup()
        {
                Input[0].setConnectionType(IMAGE);
                Input[0].setConnectionMode(NORMAL);
                Input[0].shortDescription = "";
                Input[0].setConnectionDescription(new String[]{
                        "Input GS image"        });

                Input[1].setConnectionType(INTEGER);
                Input[1].setConnectionMode(NORMAL);
                Input[1].setDefaultValue(new Integer(4));
                Input[1].shortDescription = "[Integer] Clusters [Default = 4]";
                Input[1].setConnectionDescription(new String[]{
                        "Number of clusters required" });

                Input[2].setConnectionType(INTEGER);
                Input[2].setConnectionMode(NORMAL);
                Input[2].setDefaultValue(new Integer(150));
                Input[2].shortDescription = "[Integer] Loop count [Default = 150]";
                Input[2].setConnectionDescription(new String[]{
                        "Reconstruction Loop Count"    });

                Output[0].setConnectionType(IMAGE);
                Output[0].setConnectionMode(NORMAL);
                Output[0].shortDescription = "";
                Output[0].setConnectionDescription(new String[]{
                        "Boundary removal and K-Means clustering"    });

                Output[1].setConnectionType(IMAGE);
                Output[1].setConnectionMode(NORMAL);
                Output[1].shortDescription = "";
                Output[1].setConnectionDescription(new String[]{
                        "Closed structures"    });

                Output[2].setConnectionType(INTEGER);
                Output[2].setConnectionMode(NORMAL);
                Output[2].shortDescription = "";
                Output[2].setConnectionDescription(new String[]{
                        "Approx perimeter of the strong edges"        });
        }
```

Fig. 22.6
(Continued)

```
// The second method is called whenever a double click event occurs over this block,
// double clicks are useful for generating frames or dialogs for viewing or
// altering images or for entering processing parameters, e.g. thresholding
  public void doubleClick() {}

// The next method is called whenever the block has new inputs which need to be
// processed. Basically what you need to do inside this method is read the new
// data in from the sockets, process images using the java image processing libraries
// then  setting the output image and finally but most importantly  signal the new
// state of the block as being WAITING_TO_SEND, i.e. the image has been sent to
// the plug and awaits transmission to the next block.

  public Object create(DataBlock arguments) {
  // setup input variables
    GrayImage argument0 = arguments.getGrayImage(0);
    int argument1 = arguments.getInteger(1);
    int argument2 = arguments.getInteger(2);

  // Setup return variables
    DataBlock return0 = blob_test_dev(argument0,argument1,argument2);
    return(return0);
  }

// The sample program reads in a greyscale image and two integer variables representing
// number of clusters required and the loop count for the reconstruction by dilation
// operation. The block return the k-means clusters after all edge data has been removed
// from the image. It also returns the fitting of closed curves to the strong features
// in the input image. The pixel count of this edge data is also returned.

  private DataBlock blob_test_dev(GrayImage inp1,int clus, int loop_count) {
    int inp1_width = inp1.getWidth();
    int inp1_height = inp1.getHeight();

  // We must wrap all primitive classes e.g. wrap the integer class
    Integer clus_wrap = new Integer(clus);
    GrayImage output_a = new GrayImage(inp1_width,inp1_height);
    GrayImage output_b = new GrayImage(inp1_width,inp1_height);

  // Remove boundary regions using reconstruction by dilation
    output_a = (GrayImage)NeatVision.create("SingleThreshold",inp1, new Integer(0));
    output_a = (GrayImage)NeatVision.create("Mask",output_a, new Integer(3));
    output_a = (GrayImage)NeatVision.create("Not",output_a);
    output_a = (GrayImage)NeatVision.create("Minimum",inp1, output_a);
    for(int lc=0;lc<loop_count;lc++)
     {
     output_a = (GrayImage)NeatVision.create("Dilation",output_a, new Integer(8));
     output_a = (GrayImage)NeatVision.create("Minimum",inp1, output_a);
     }
    output_a = (GrayImage)NeatVision.create("Subtract",inp1,output_a);
    output_a = (GrayImage)NeatVision.create("Median",output_a);

  // Apply K-Means clustering
    output_a = (GrayImage)NeatVision.create("GrayScaleCluster",output_a,clus_wrap);

  // Load the canny magnitude image
    DataBlock temp = (DataBlock)NeatVision.create("Canny", inp1,
                           new Double(1.9),new Integer(20),new Integer(230));
    output_b = (GrayImage)temp.getGrayImage(0);
    output_b = (GrayImage)NeatVision.create("SingleThreshold",output_b, new Integer(127));

  // Keep only closed structures and find approx perimeter of the strong edges
    output_b = (GrayImage)NeatVision.create("EdgeLabel",output_b,new Boolean(true));
    output_b = (GrayImage)NeatVision.create("SingleThreshold",output_b, new Integer(1));
    Integer area_1 = (Integer)NeatVision.create("WhitePixelCounter",output_b);
    output_b = (GrayImage)NeatVision.create("Add",output_b,inp1);

  // Make the data available to other library functions.
    DataBlock returns = new DataBlock(); returns.add(output_a);
    returns.add(output_b); returns.add(area_1); return(returns);
  }
}
```

Fig. 22.6
Development of the `TestDev` class

22.7.1 Outline Solution

The proposed solution, as summarised in the NeatVision program illustrated in *Fig. 22.11*, consists of a number of distinct stages. This solution is used solely to illustrate the power and flexibility of NeatVision and does not represent the optimal solution for such an application.

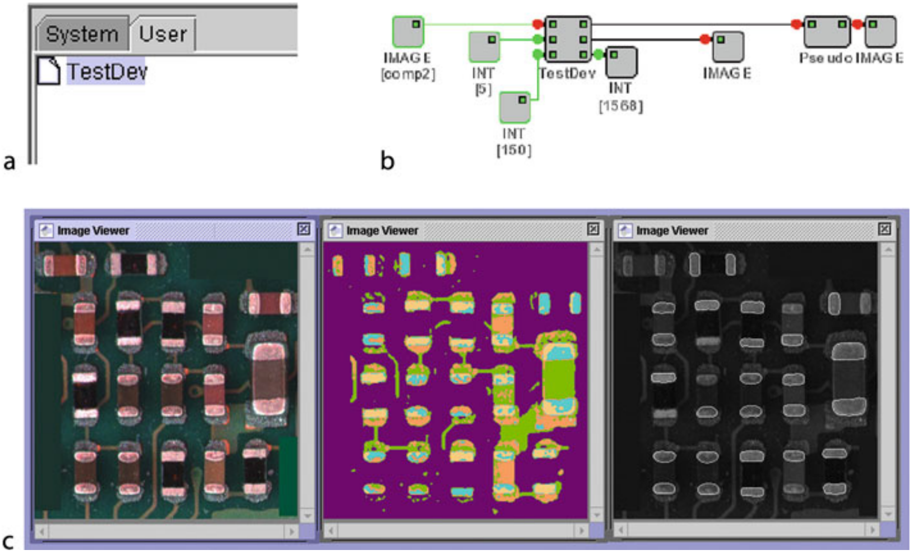


Fig. 22.7
Implementation of the `TestDev` class illustrated in *Fig. 22.6*. (**a**) The associated class file tag in the user area. (**b**) A sample program illustrating the operation of this block. (**c**) Sample images (*left* to *right*): input image, K-Means clustered image and the closed structure overlay image

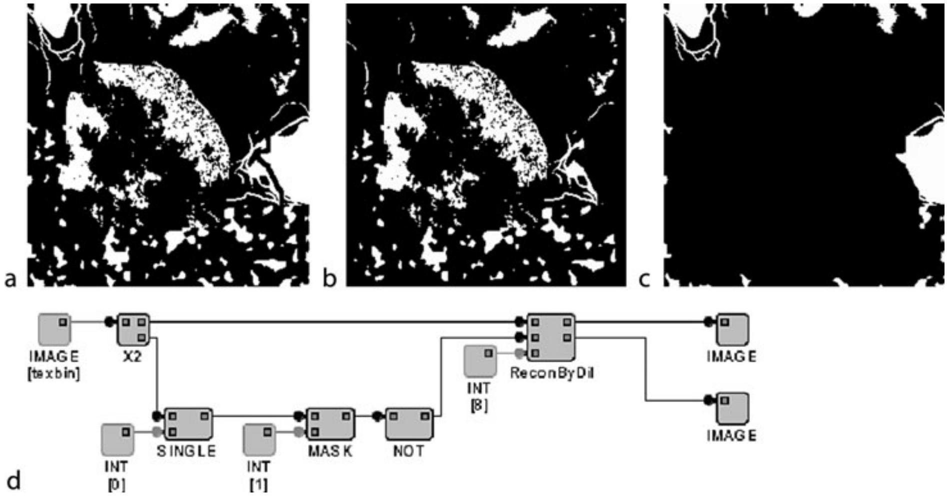
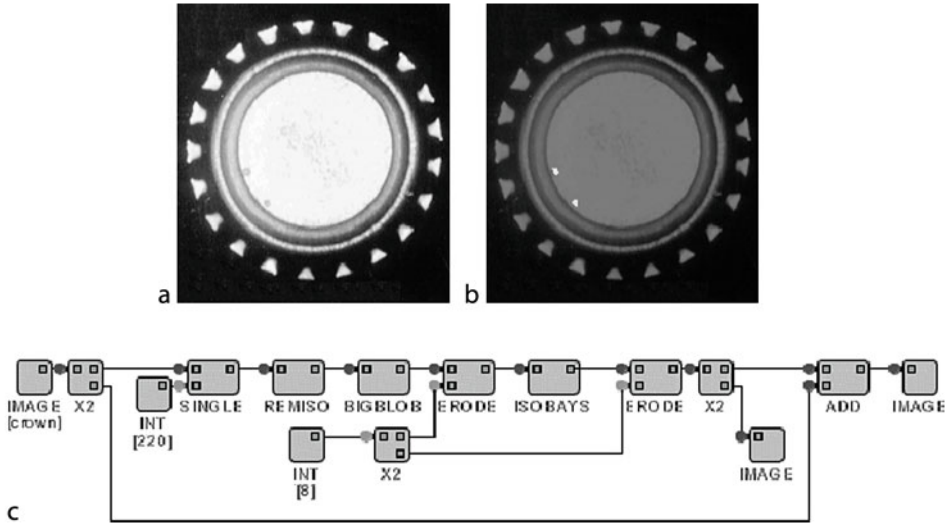
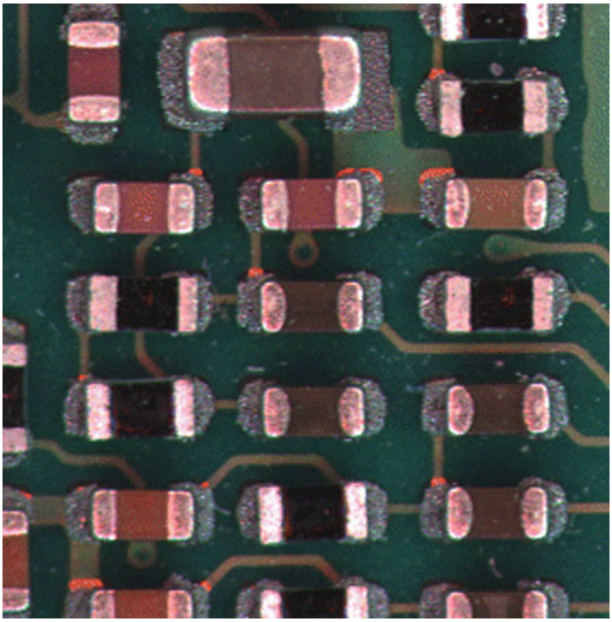


Fig. 22.8
Removal of boundary objects using reconstruction by dilation. (**a**) Original image, (**b**) boundary object removal, (**c**) detected boundary objects, (**d**) associated visual workspace



■ Fig. 22.9
Bottle top inspection. (a) Input image, (b) output image in which the defects are highlighted in *white*, (c) the NeatVision visual program



■ Fig. 22.10
Image segment (pcb_3) containing 15 surface-mounted components fully with in its field of view (Image supplied courtesy of Agilent Technologies (Ireland), acquired using their *sequential colour* technique at 470, 524 and 660 nm (BGR))

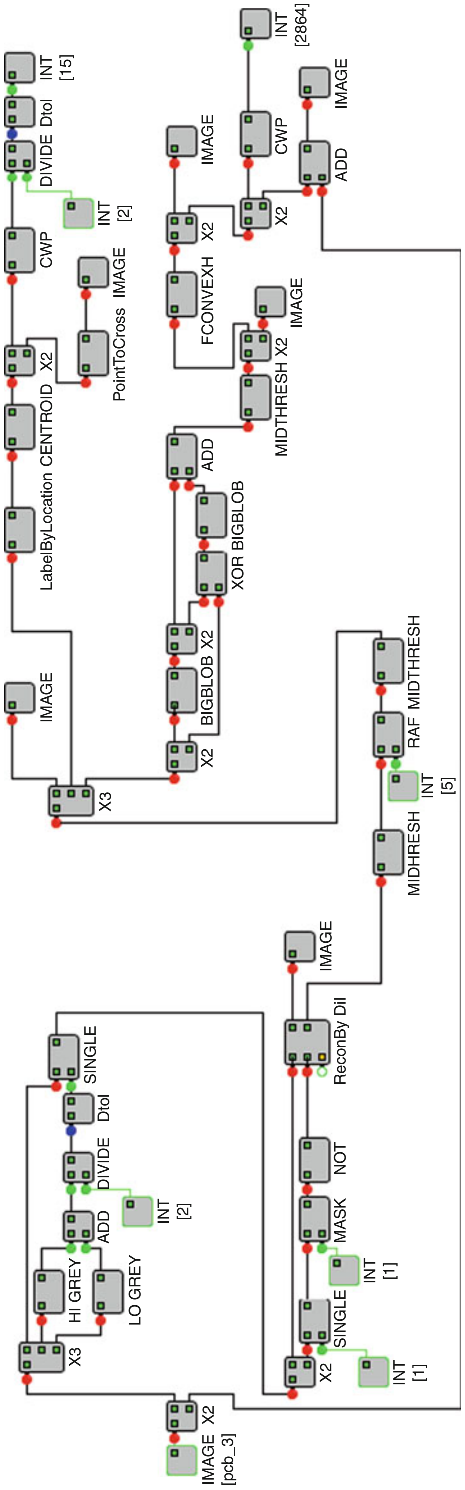


Fig. 22.11
Proposed NeatVision solution

Stage 1: The input colour image (pcb_3.gif) is loaded and converted to grey scale. A binary version of this image is then automatically produced by examining the grey-scale histogram upper and lower values. Due to the high contrast, the automated threshold selection consists of the midpoint between the upper and lower grey scales in the original image (➲ *Fig. 22.12*).

Stage 2: Using the morphological technique *Reconstruction by Dilation*, we isolate any part of the image touching the boundary (➲ *Fig. 22.13*).

Stage 3: Once the incomplete objects touching the image boundary have been removed, we use a 5 × 5 RAF (Rectangular Averaging Filter) to remove any remaining noise (➲ *Fig. 22.14*). Take care to threshold the filtered image at mid-grey as the RAF filter produces a grey-scale image.

Stage 4: Each blob region is now assigned a grey-scale value using labelling by location. These grey patches are then count by finding and marking each grey patch by its centroid. The centroids are then counted and divided by 2 to give the final component count (➲ *Fig. 22.15*).

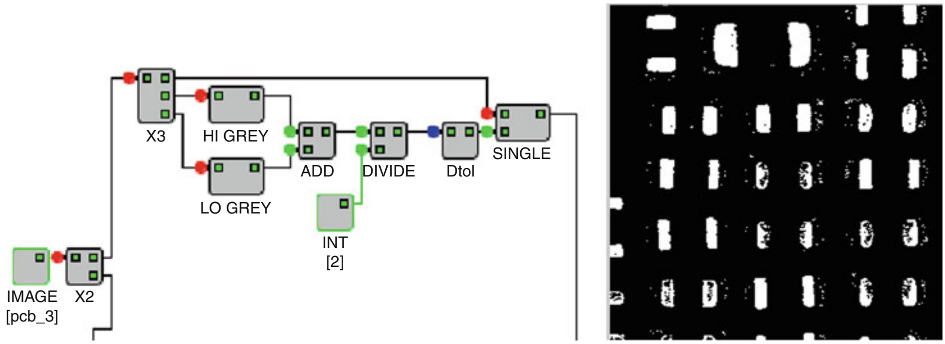


Fig. 22.12
Automated threshold selection stage and the resultant binary image

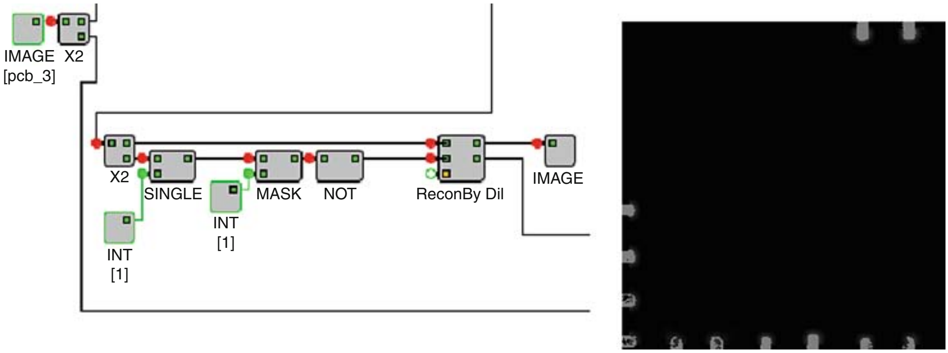


Fig. 22.13
Using reconstruction by dilation to identify the incomplete objects touching the image boundary

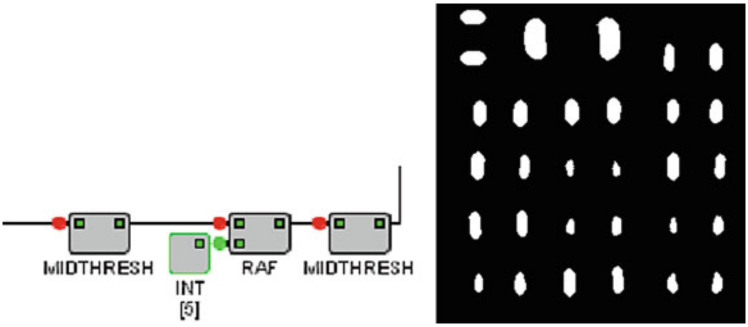


Fig. 22.14

Isolating the components of interest and noise removal

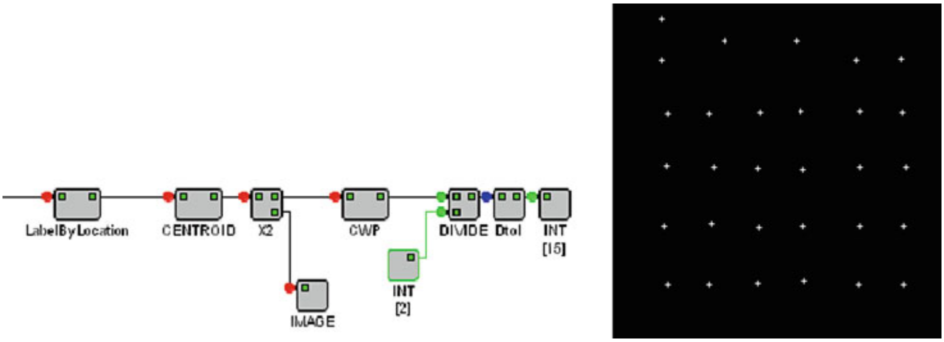


Fig. 22.15

Component counting by isolating and counting the grey patch centroid values (a cross indicates each centroid value)

Stage 5: Using the fact that the largest surface-mounted component is bounded by the two largest blobs in the image, we can identify and extract these blob regions (➲ *Fig. 22.16*).

Stage 6: The final step involves finding the convex hull of the output from the previous stage. This now approximates the largest surface-mounted component. The area of this region is then calculated to determine its size. This convex hull image is then combined with the original image to highlight its location (➲ *Fig. 22.17*).

22.8 Users Summary

The following list summarises some of the main NeatVision methods users may wish to interface too. Many of these are fairly self-explanatory, but if the method you require is not listed or does not have enough information to enable you to use it, drop us an email at

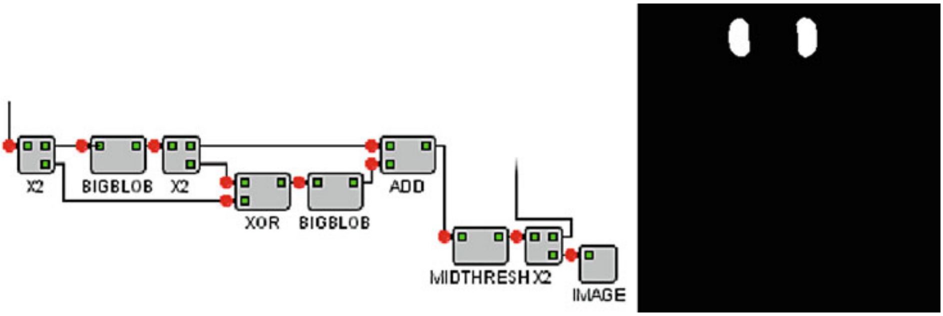


Fig. 22.16
Identifying the bounds of the largest component

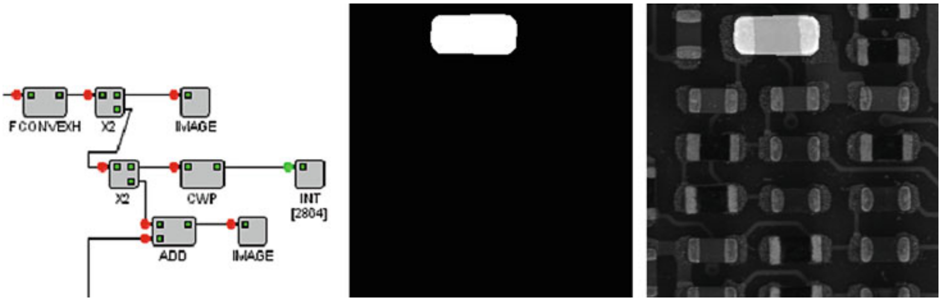


Fig. 22.17
Identifying calculating the area of the largest component. Overlay on original image to highlight its location

tech@neatvision.com with "NeatVision Methods" in the subject line. Additional help can be found in the input/output tags for each block in the NeatVision visual programming interface. Also refer to [2].

Normalisation of grey-scale image operations occurs to keep the output image within grey-scale range 0–255.

22.9 Downloading NeatVision

22.9.1 Overview

NeatVision is a free Java-based image analysis and software development environment, which provides high-level access to a wide range of image-processing algorithms through a well-defined and easy to use graphical interface. NeatVision is in its second major release. New features include: a full developer's guide with method listings and program examples, DICOM and Analyse medical image sequence viewers, URL control, feature fitting, supervised and unsupervised colour clustering, DCT, Improved FFT, 3D volume processing and surface rendering.

Method	Description	Inputs	Outputs
		(Index #: data type [descriptor])	(Index #: data type [descriptor])
DATA	Image, Integer, Double, Boolean, String, Array (of integers) and 3D (DICOM, Analyse, Vol, Sequence)		
FLOW CONTROL	SplitterX2, SplitterX3, SplitterX4, Feedback, If, Else, For and Terminate		
UTILITIES			
HalveImageSize	A grey-scale image whose size is halved	0:GrayImage	0:GrayImage
DoubleImageSize	A grey-scale image whose size is doubled	0:GrayImage	0:GrayImage
PointToSquare	A grey-scale image whose white pixels are represented by white squares	0:GrayImage	0:GrayImage
PointToCross	A grey-scale image whose white pixels are represented by white crosses	0:GrayImage	0:GrayImage
Rotate	A grey-scale image is rotated in a clockwise direction by a user specified amount	0:GrayImage 1: Integer [user specified rotation (degrees)]	0:GrayImage
RotatePlus90	A grey-scale image is rotated in a clockwise direction by 90°	0:GrayImage	0:GrayImage
RotateMinus90	A grey-scale image is rotated in an anticlockwise direction by 90°	0:GrayImage	0:GrayImage
ROI[a]	A grey-scale image from which a rectangular region of interest is extracted by the user via the GUI	0:GrayImage	0:GrayImage
PolyROI[a]	A grey-scale image from which a polygon region of interest is extracted by the user via the GUI	0:GrayImage [User interaction]	0:GrayImage
EnhancePolyROI[b]	A grey-scale image from which a polygon region of interest shall be emphasised. User-defined input region	0:GrayImage [User interaction]	0:GrayImage

Measure_Line	An image from which the Euclidean distance between two user selected points is calculated. Must rerun program to generate new line length	0:GrayImage [User interaction]	0:Double [Euclidean distance]
Scale	A grey-scale image is scaled by user-defined dimensions	0:GrayImage 1: Integer [width of the scaled image] 2: Integer [height of the scaled image]	0:GrayImage
Mask	A grey-scale image whose border is masked by a user specified amount	0:GrayImage 1: Integer [Mask size in pixels, Default = 3]	0:GrayImage
Centroid	Replace the grey-scale shapes (Range 0–255) in the original image by their respective centroids (commonly used after the 8-bit labelling operators)	0:GrayImage	0:GrayImage [Binary]
Centroid_16	Replace the grey-scale shapes (Range 0–65535) in the original image by their respective centroids (commonly used after the Label_16 operators)	0:GrayImage	0:GrayImage [Binary]
BinaryToGreyscale	Convert WHITE (255) pixels in a binary image to a given grey scale	0:GrayImage [Binary] 1:Integer [grey scale (0–255)]	0:GrayImage
GreyScalePixelSum	Generates an integer which is the sum of all pixels contained in the input image	0:GrayImage	0:Integer
FirstWhitePixelLocator	Coordinate point representing the location of the first white pixel in the image input image	0:GrayImage	0:Coordinate
RemoveIsolatedWhitePixels	Any white pixels with less than one white pixel (3 × 3) neighbour are set to black. This can be used to remove noise from a binary image	0:GrayImage	0:GrayImage [Binary]
SaltnPepperGenerator	Add salt and pepper noise to the input image	0:GrayImage 1:Double (0–1.0)	0:GrayImage

(Continued)

Method	Description	Inputs	Outputs
		(Index #: data type [descriptor])	(Index #: data type [descriptor])
AdditiveWhiteNoiseGenerator	Add a user-defined level of white noise to the input image	0:GrayImage 1:Integer (1–1024)	0:GrayImage
GaussianNoiseGenerator	Add a user-defined quantity of Gaussian noise to the input image	0:GrayImage 1:Double (0–255.0)	0:GrayImage
RayleighNoiseGenerator	Add a user-defined quantity of Rayleigh noise to the input image	0:GrayImage 1:Double (1.0–255.0)	0:GrayImage
PoissonNoiseGenerator	Add a user-defined quantity of Poisson noise to the input image	0:GrayImage 1:Double (0–511.0)	0:GrayImage
HTTPSendScript	Send arguments to a URL	0:String [URL] 1:String [Arguments]	0:String [Return values]
HTTPGetImage	Retrieve image from a URL	0:String [URL]	0:GrayImage [Retrieved Image]
ARITHIMETIC			
Add	Image addition	0:GrayImage [A] 1:GrayImage [B]	0:GrayImage [C = A+B]
Subtract	Image subtraction	0:GrayImage [A] 1:GrayImage [B]	0:GrayImage [C = A – B]
Multiply	Image multiply	0:GrayImage [A] 1:GrayImage [B]	0:GrayImage [C = A*B]
Divide	Image division	0:GrayImage [A] 1:GrayImage [B]	0:GrayImage [C = A/B]

And	Boolean AND operation	0:GrayImage [A] 1:GrayImage [B]	0:GrayImage [C = AND(A,B)]
Or	Boolean OR operation	0:GrayImage [A] 1:GrayImage [B]	0:GrayImage [C = OR(A,B)]
Not	Boolean NOT operation	0:GrayImage [A]	0:GrayImage [C = NOT(A)]
Xor	Boolean Exclusive OR operation	0:GrayImage [A] 1:GrayImage [B]	0:GrayImage [C = XOR(A,B)]
Minimum	Minimum of two images	0:GrayImage [A] 1:GrayImage [B]	0:GrayImage [C = Min(A,B)]
Maximum	Maximum of two images	0:GrayImage [A] 1:GrayImage [B]	0:GrayImage [C = Max(A,B)]
HISTOGRAM			
HighestGreyLevelCalculator	Compute the highest grey level from the input image	0:GrayImage	0:Integer [highest grey level]
LowestGreyLevelCalculator	Compute the lowest grey level from the input image	0:GrayImage	0:Integer [lowest grey level]
MeanSquareError	Compare the input images using the mean square error operation	0:GrayImage 1:GrayImage	0:Double [mean square error]
AverageIntensityCalculator	Compute the average intensity of the input image	0:GrayImage	0:Double [average intensity]
EntropyCalculator	Compute the entropy of the input image	0:GrayImage	0:Double [entropy]
VarienceCalculator	Compute the variance of the input image	0:GrayImage	0:Double [varience]
KurtosisCalculator	Compute the kurtosis of the input image	0:GrayImage	0:Double [kurtosis]
StandardDeviationCalculator	Compute the standard deviation of the input image	0:GrayImage	0:Double [standard deviation]
SkewnessCalculator	Compute the skewness deviation of the input image	0:GrayImage	0:Double [skewness]
LocalEqualisation3 × 3	Local histogram equalisation using a 3 × 3 region	0:GrayImage	0:GrayImage
LocalEqualisation5 × 5	Local histogram equalisation using a 5 × 5 region	0:GrayImage	0:GrayImage

■ **(Continued)**

Method	Description	Inputs	Outputs
		(Index #: data type [descriptor])	(Index #: data type [descriptor])
PROCESSING			
Inverse	Inverse the LUT of the input image	0:GrayImage	0:GrayImage
Logarithm	Transform the linear LUT into logarithmic	0:GrayImage	0:GrayImage
Exponential	Transform the linear LUT into exponential	0:GrayImage	0:GrayImage
Power	The linear LUT is raised to a user specified double value	0:GrayImage	0:GrayImage
		1:Integer [power, default = 3.0]	
Square	The linear LUT is raised to power of 2	0:GrayImage	0:GrayImage
SingleThreshold	Single threshold operation	0:GrayImage	0:GrayImage [Binary]
		1:Integer [(1–255): Default = 128]	
MidlevelThreshold	Single threshold operation: threshold level = MIDGREY (127)	0:GrayImage	0:GrayImage [Binary]
DualThreshold	Dual threshold operation. All pixels between the upper and lower thresholds are marked in WHITE (255)	0:GrayImage	0:GrayImage [Binary]
		1:Integer [upper value, default = 128]	
		2:Integer [lower value, default = 1]	
TripleThreshold	This operation produces an LUT in which all pixels below the user specified lower level appear black, all pixels between the user specified lower level and the user specified upper level inclusively appear grey and all pixels above the user specified upper level appear white	0:GrayImage	0:GrayImage
		1:Integer [upper value, default = 128]	
		2:Integer [lower value, default = 1]	

EntropicThreshold	Compute the entropy-based threshold. Relies on maximising the total entropy of both the object and background regions to find the appropriate threshold	0:GrayImage	0:Integer
Threshold3 × 3	Adaptive threshold in a 3 × 3 region	0:GrayImage	0:GrayImage
		1:Integer [constant offset, default = 0]	
Threshold5 × 5	Adaptive threshold in a 5 × 5 region	0:GrayImage	0:GrayImage
		1:Integer [constant offset, default = 0]	
IntensityRangeEnhancer	Stretch the LUT in order to occupy the entire range between BLACK (0) and WHITE (255)	0:GrayImage	0:GrayImage
HistorgramEqualiser	Histogram equalisation	0:GrayImage	0:GrayImage
IntensityRangeStrecher	Stretch the LUT between the lower and upper threshold to occupy the entire range between BLACK (0) and WHITE (255)	0:GrayImage	0:GrayImage
		1:Integer [lower grey level, default = 0]	
		2:Integer [upper grey level, default = 255]	
IntegrateImageRows	Integrate image rows	0:GrayImage	0:GrayImage
IntegrateImageColumns	Integrate Image columns	0:GrayImage	0:GrayImage
LeftToRightSum	Pixel summation along the line	0:GrayImage	0:GrayImage
LeftToRightWashFunction	Left To Right wash function (once a white pixel is found, all pixels to its right are also set to white)	0:GrayImage	0:GrayImage
RightToLeftWashFunction	Right To Left wash function (once a white pixel is found, all pixels to its left are also set to white)	0:GrayImage	0:GrayImage
TopToBottomWashFunction	Top To Bottom wash function (once a white pixel is found, all pixels to its below are also set to white)	0:GrayImage	0:GrayImage

(Continued)

Method	Description	Inputs	Outputs
		(Index #: data type [descriptor])	(Index #: data type [descriptor])
BottomToTopWashFunction	Bottom To Top wash function (once a white pixel is found, all pixels to its above are also set to white)	0:GrayImage	0:GrayImage
FILTER			
Convolution	Convolution. This operation requires coefficients to be specified in the form of a square, odd-sized integer array, "null" represents "don't cares"	0:GrayImage 1:Integer [ ] [Array of mask values. No entry default to null. "Don't Care" = null statement]	0:GrayImage
DOLPS	DOLPS – Difference of low pass 3 × 3 filters. Image A results from applying three iterations of the low pass filter. Image B results from applying six iterations of the low pass filter. DOLPS = A−B	0:GrayImage	0:GrayImage
LowPass	Low pass 3 × 3 filter	0:GrayImage	0:GrayImage
Sharpen	High pass 3 × 3 filter	0:GrayImage	0:GrayImage
Median	Median 3 × 3 filter	0:GrayImage	0:GrayImage
Midpoint	Midpoint 3 × 3 filter	0:GrayImage	0:GrayImage
RectangularAverageFilter	Rectangular Average Filter operation. Size of filter is user defined	0:GrayImage 1:Integer [filter size, default = 5]	0:GrayImage
SmallestIntensityNeighbour	Replace the central pixel of the 3 × 3 mask with the minimum value	0:GrayImage	0:GrayImage

LargestIntensityNeighbour	Replace the central pixel of the 3 × 3 mask with the maximum value	0:GrayImage	0:GrayImage
AdaptiveSmooth	Adaptive smoothing of grey-scale images. In order to apply it to colour images, the input image has to be split into RGB components and adaptive smooth has to be applied to each channel. If the colour image is applied directly the algorithm will smooth the average intensity image. (Slow process)	0:GrayImage 1:Integer [number of iterations: possible values: 1 to 10, default = 5] 2:Double [variance strength: possible values: 0.1 −> 0.9, default = 0.2] 3:Double [Diffusion parameter: possible values: 1.0 −> 20.0, default = 10.0]	0:GrayImage
EDGES			
Roberts	Roberts edge detector	0:GrayImage	0:GrayImage
Sobel	Sobel edge detector	0:GrayImage	0:GrayImage
Laplacian	Laplacian edge detector. User-defined 4-connected or 8-connected neighbourhood	0:GrayImage 1:Integer [possible values: 4 or 8, default = 8]	0:GrayImage
Prewitt	Prewitt edge detector	0:GrayImage	0:GrayImage
FreiChen	FreiChen edge detector	0:GrayImage	0:GrayImage
BinaryBorder	Binary Border edge detector	0:GrayImage [Binary]	0:GrayImage [Binary]
NonMaxima	Edge detection using non-maxima suppression	0:GrayImage	0:GrayImage
IntensityGradientDirection	Compute the 3 × 3 intensity gradient direction. Gradients range from 1 to 8	0:GrayImage	0:GrayImage [pixel values from 1 to 8]

■ **(Continued)**

		Inputs	Outputs
Method	Description	(Index #: data type [descriptor])	(Index #: data type [descriptor])
ZeroCrossingsDetector	Zero crossings edge detector	0:GrayImage	0:GrayImage
Canny	Canny edge detector	0:GrayImage 1:Double [standard deviation or spread parameter, possible values: 0.2 –> 20.0, default = 1.0] 2:Integer [lower threshold, default = 1] 3:Integer [upper threshold, default = 255]	0:GrayImage [edge magnitudes] 1:GrayImage [edge directions]
EdgeLabel	Edge labelling operation. Expects a binary image resulting from the application of the Canny edge detector	0:GrayImage 1:Boolean [Set True if you want closed structures]	0:GrayImage [A binary image whose edge pixels are grouped into polygonal shapes]
LineFitting	Line fitting in the edge structure. Expects a binary image resulting from the application of the Canny edge detector	0:GrayImage 1:Boolean [Set True if you want closed structures]	0:GrayImage [A binary image whose edge pixels are grouped into polygonal shapes]

ArcFitting	Arc fitting in the edge structure. Expects a binary image resulting from the application of the Canny edge detector	0:GrayImage 1:Boolean [Set True if you want closed structures] 2:Boolean [Set True if you want display the circles associated with detected arcs] 3:Boolean [Set True if you want display the lines that are not grouped into arcs segments]	0:GrayImage [A binary image whose edge pixels are grouped into polygonal shapes]
EdgeLinking [13]	Edge linking (scanning window is user defined). Expects a binary image resulting from the application of the Canny edge detector	0:GrayImage 1:Integer [The size of scanning window. (5–11)]	0:GrayImage [Edge linked image]
ANALYSIS			
ThinOnce	Full application of the thinning algorithm. Thin *till completion* resulting in a skeleton image	0:GrayImage [Binary]	0:GrayImage [Binary]
Thin	The input binary image is thinned N times as specified by the user	0:GrayImage [Binary] 1:Integer [N – number of iterations]	0:GrayImage [Binary]
CornerPointDetector	Skeleton corner detection from a binary image based on a 3 × 3 region	0:GrayImage [Binary]	0:GrayImage [Binary]
JunctionDetector	Skeleton junction detection from a binary image based on a 3 × 3 region	0:GrayImage [Binary]	0:GrayImage [Binary]
LimbEndDetector	Skeleton limb end detection from a binary image based on a 3 × 3 region	0:GrayImage [Binary]	0:GrayImage [Binary]
BiggestBlob	Extract the biggest white blob from a binary image	0:GrayImage [Binary]	0:GrayImage [Binary]
SmallestBlob	Extract the smallest white blob from a binary image	0:GrayImage [Binary]	0:GrayImage [Binary]

■ **(Continued)**

Method	Description	Inputs	Outputs
		(Index #: data type [descriptor])	(Index #: data type [descriptor])
BlobFill	Fill the holes in a binary image	0:GrayImage [Binary]	0:GrayImage [Binary]
Labeller	Label by location unconnected shapes in a binary image (Range 0–255)	0:GrayImage [Binary]	0:GrayImage
LabelByArea	Label the unconnected shapes in a binary image in relation to their size (Range 0–255)	0:GrayImage [Binary]	0:GrayImage
MeasureLabelledObjects	Measure the N (user specified) largest objects in a binary image (Range 0–255)	0:GrayImage [Binary] 1:Integer [limit on the number of labelled objects measured, default = 5]	0:String [contains data describing the measured objects: (Grey Scale, Area, Centroid)]
WhiteBlobCount	Count the number of white bobs in a binary image (Range 0–255)	0:GrayImage [Binary]	0:Integer [Range 0–255] 1:GrayImage [A white cross is overlaid on each blob found]
Label_16	Label by location the unconnected shapes in a binary image (Range 0–65535). Note: This is outside the 8-bit display range. Slow process	0:GrayImage [Binary]	0:GrayImage
WhiteBlobCount_16	Count the number of white bobs in a binary image (Range 0–65535). Slow process	0:GrayImage [Binary]	0:Integer [Range 0–65535] 1:GrayImage [A white cross is overlaid on each blob found]
ConvexHull	Compute the convex hull boundary	0:GrayImage [Binary]	0:GrayImage [Binary]
FilledConvexHull	Compute the filled convex hull	0:GrayImage [Binary]	0:GrayImage [Binary]
CrackDetector	Highlight cracks in the input image	0:GrayImage	0:GrayImage
EulerNumberCalculator	Compute the Euler number from a binary image	0:GrayImage [Binary]	0:Integer [Euler number]

WhitePixelCounter	Compute the number of white pixels	0:GrayImage	0:Integer [pixel count]
IsolateHoles	Isolate holes in a binary image	0:GrayImage [Binary]	0:GrayImage [Binary]
IsolateBays	Isolate bays in a binary image	0:GrayImage [Binary]	0:GrayImage [Binary]
ConnectivityDetector	Connectivity detection in a thinned skeleton binary image. Mark points critical for connectivity in a 3 × 3 region	0:GrayImage [Binary]	0:GrayImage
BoundingBox	Minimum area bounding rectangle.	0:GrayImage	0:GrayImage
FilledBoundingBox	Filled minimum area bounding rectangle	0:GrayImage	0:GrayImage
BoundingBoxTop Coordinate	Compute the top left coordinate of the minimum area bounding rectangle	0:GrayImage	0:Coordinate [top left]
BoundingBoxBottom Coordinate	Compute the bottom right coordinate of the minimum area bounding rectangle	0:GrayImage	0:Coordinate [bottom right]
CornerDetector	Grey-Scale (SUSAN) corner detector	0:GrayImage 1:Integer [Brightness threshold] 2:Integer [Geometric threshold]	0:GrayImage [Corner points]
K-MEANS CLUSTERING			
GrayScaleCluster	Cluster a grey-scale image (number of clusters are user defined) using the k-means algorithm	0:GrayImage 1:Integer [Number of clusters]	0:GrayImage [Grey-scale]
ColourCluster	Cluster a colour image (number of clusters are user defined) using the k-means algorithm	0:Image [Colour Image Input] 1:Integer [Number of clusters]	0:GrayImage [Grey-scale]

■ **(Continued)**

Method	Description	Inputs	Outputs
		(Index #: data type [descriptor])	(Index #: data type [descriptor])
Un_ColourCluster	Unsupervised colour clustering using the k-means algorithm	0:Image	0:GrayImage [Grey-scale]
		1:Double [Low threshold (possible values 0.5–1.0), default = 0.6]	1:Image [Colour]
		2:Double [High threshold (possible values 1.0–1.5), default = 1.2]	2:Integer [Number of clusters]
PseudoColour	Pseudo-colour operation	0:Image [grey-scale or colour image]	0:Image [false colour image]
TRANSFORM			
MedialAxisTransform	Medial axis transform operation. Binary image showing the simple skeleton	0:Image [binary]	0:Image [binary]
MedialAxisTransform_GS	Medial axis transform operation. GS image where each point on the skeleton has an intensity which represents its distance to a boundary in the original object	0:Image [binary]	0:GrayImage [grey scale]
FFT	Fast Fourier Transform: FFT	0:GrayImage [Input image dimension must be a power of 2]	0:File [Fourier Data File]
			1:GrayImage [Grey-scale image transformed to its Fourier coefficients]
IFFT	Inverse Fourier Transform	0:File [A Fourier data file which shall be interpreted as an image.]	0:GrayImage [The resulting grey-scale image which represents the interpreted Fourier data]

FFTLowpass	Low pass frequency filter	0:File [Fourier Data File]	0:File [Fourier Data File]
		1:Double [cut-off value (0–1.0)]	1:GrayImage [Grey-scale image transformed to its Fourier coefficients]
FFTHighpass	High pass frequency filter	0:File	0:File [Fourier Data File]
		1:Double [cut-off value (0–1.0)]	1:GrayImage [Grey-scale image transformed to its Fourier coefficients]
FFTAdaptiveLowpass	FFT adaptive lowpass filter	0:File	0:File [Fourier Data File]
		1:Double [limit (0–1.0)]	1:GrayImage [Grey-scale image transformed to its Fourier coefficients]
FFTBandpass	FFT band-pass filter	0:File [Fourier Data File]	0:File [Fourier Data File]
		1:Double [inner limit (0–1.0)]	1:GrayImage [Grey-scale image transformed to its Fourier coefficients]
		2:Double [outer limit (0–1.0)]	
FFTBandstop	FFT band-stop filter	0:File [Fourier Data File]	0:File [Fourier Data File]
		1:Double [inner limit (0–1.0)]	1:GrayImage [Grey-scale image transformed to its Fourier coefficients]
		2:Double [outer limit (0–1.0)]	
FFTMultiply	Multiply two Fourier data files	0:File [Fourier Data File]	0:File [Fourier Data File]
		1:File [Fourier Data File]	1:GrayImage [Grey-scale image transformed to its Fourier coefficients]
FFTDivide	Divide one Fourier data file by another	0:File [Fourier Data File]	0:File [Fourier Data File]
		1:File [Fourier Data File]	1:GrayImage [Grey-scale image transformed to its Fourier coefficients]

■ **(Continued)**

Method	Description	Inputs (Index #: data type [descriptor])	Outputs (Index #: data type [descriptor])
FFTGaussian	FFT Gaussian filter. Input 0 requires an integer value that = 2^n where n is a+ve integer. Note: size = width = height	0:Integer [size of a new Fourier data file which contains Gaussian coefficients] 1:Double [Standard deviation of the Gaussian coefficients (0.1–5.0)]	0:File [Fourier Data File] 1:GrayImage [Grey-scale image transformed to its Fourier coefficients]
FFTSelectivePass	FFT selective frequency filter	0:File [Fourier Data File] 1:Double [The cut-off value of the filter (0–1.0)] 2:Double [The x-offset of the symmetric selective filter (0–1.0)] 3:Double [The y-offset of the symmetric selective filter (0–1.0)]	0:File [Fourier Data File] 1:GrayImage [Grey-scale image transformed to its Fourier coefficients]
FFTSymmetricSelectivePass	FFT selective symmetric frequency filter	0:File [Fourier Data File] 1:Double [The cut-off value of the filter (0–1.0)] 2:Double [The x-offset of the symmetric selective filter (0–1.0)] 3:Double [The y-offset of the symmetric selective filter (0–1.0)]	0:File [Fourier Data File] 1:GrayImage [Grey-scale image transformed to its Fourier coefficients]

DCT2D	Direct Cosine Transform operation	0:GrayImage [Input image dimension must be a power of 2]	0:GrayImage [Real Part] 1:GrayImage [DCT Magnitude]
IDCT2D	Inverse DCT (filtering factor is user defined)	0:GrayImage 1:Double [DCT quality coefficient (0–2.0)]	0:GrayImage [IDCT image]
Hough	Line Hough Transform	0:GrayImage [Binary]	0:GrayImage
InverseHough	Inverse Hough Transform. The integer input specifies how many of the brightest pixels shall be taken into account when performing the Inverse Hough operation	0:GrayImage 1:Integer [Number of bright points to be considered, default = 10]	0:GrayImage
CircHough	Circular Hough Transform	0:GrayImage [binary image to be subjected to the circular Hough transform]	0:GrayImage [Image] 1:GrayImage [Transform space]
CooccurrenceMatrixGenerator	Compute the co-occurrence matrix	0:GrayImage	0:GrayImage
CooccurrenceMatrixEnergyCalculator	Compute the energy of the co-occurrence matrix	0:GrayImage	0:Double
CooccurrenceMatrixEntropyCalculator	Compute the entropy of the co-occurrence matrix	0:GrayImage	0:Double
CooccurrenceMatrixContrastCalculator	Compute the contrast of the co-occurrence matrix	0:GrayImage	0:Double
CooccurrenceMatrixHomogenityCalculator	Compute the homogeneity of the co-occurrence matrix	0:GrayImage	0:Double
DistanceTransform3 × 3	Compute the distance transform in a 3 × 3 window (input binary image)	0:GrayImage [Binary]	0:GrayImage
DistanceTransform5 × 5	Compute the distance transform in a 5 × 5 window (input binary image)	0:GrayImage [Binary]	0:GrayImage
LeftToRightDistanceTransform	Left to right distance transform (input binary image)	0:GrayImage [Binary]	0:GrayImage
RightToLeftDistanceTransform	Right to left distance transform (input binary image)	0:GrayImage [Binary]	0:GrayImage
TopToBottomDistanceTransform	Top to bottom distance transform (input binary image)	0:GrayImage [Binary]	0:GrayImage

■ **(Continued)**

Method	Description	Inputs (Index #: data type [descriptor])	Outputs (Index #: data type [descriptor])
BottomToTopDistanceTransform	Bottom to top distance transform (input binary image)	0:GrayImage [Binary]	0:GrayImage
GrassFireTransform	Grass fire transform (input binary image) [8-connected]	0:Image [Binary]	0:Image [grey-scale]
MORPHOLOGY			
Dilation	Dilation operation (user specify connectivity of the structured element 4 or 8)	0:GrayImage 1:Integer [(4 or 8), default = 8]	0:GrayImage
Erosion	Erosion operation (user specify connectivity of the structured element 4 or 8)	0:GrayImage 1:Integer [(4 or 8), default = 8]	0:GrayImage
Open	Opening operation (user specify connectivity of the structured element 4 or 8)	0:GrayImage 1:Integer [(4 or 8), default = 8]	0:GrayImage
Close	Closing operation (user specify connectivity of the structured element 4 or 8)	0:GrayImage 1:Integer [(4 or 8), default = 8]	0:GrayImage
ErodeNxN	Erosion operation with a user-defined NxN structuring element (X or null = don't cares)	0:GrayImage 1:Integer [Array]	0:GrayImage
DilateNxN	Dilation operation with a user-defined NxN structuring element (X or null = don't cares)	0:GrayImage 1:Integer [Array]	0:GrayImage
MorphologicalValley	Morphological valley operation (user specify connectivity of the structured element 4 or 8) [Default = 8]	0:GrayImage 1:Integer (4 or 8)	0:GrayImage

MorphologicalTophat	Morphological top-hat operation (user specify connectivity of the structured element 4 or 8) [Default = 8]	0:GrayImage 1:Integer (4 or 8)	0:GrayImage
HitAndMiss	Hit-and-miss transformation. Hit and miss array masks must not overlap	0:GrayImage 1:Integer [user-defined hit array, blanks correspond to DON'T CARE)] 2:Integer [user-defined miss array]	0:GrayImage
MorphGradient	Morphological Gradient (user specify connectivity of the structured element 4 or 8) [Default = 8]	0:GrayImage 1:Integer	0:GrayImage
ReconByDil	Reconstruction by dilation	0:GrayImage 1:GrayImage [Seed] 2:Integer [SE size]	0:GrayImage [Reconstructed] 1:GrayImage [Elements removed]
ReconByDil_UI	Reconstruction by dilation via a user selected seed point (8-connected)	0:GrayImage [User interaction]	0:GrayImage [Reconstructed] 1:GrayImage [Elements removed]
DBLT	Double [Hysteresis] Threshold–based reconstruction. Binary Outputs. Seed threshold to reduce noise Mask threshold to maximise signal	0:GrayImage 1:Integer [seed threshold] 2:Integer [mask threshold]	0:GrayImage [Reconstructed] 1:GrayImage [Seed Image] 2:GrayImage [Seed Image]
Watershed	Watershed transform (return the watershed image and the region boundaries image)	0:GrayImage	0:GrayImage [Watershed Image] 1:GrayImage [Binary, Watershed boundaries]
COLOUR			
GreyScaler	Average three colour planes	0:Image [colour]	0:GrayImage

■ **(Continued)**

Method	Description	Inputs	Outputs
		(Index #: data type [descriptor])	(Index #: data type [descriptor])
ColourToRGB	Extract the RGB colour planes	0:Image [colour]	0:GrayImage [R]
			1:GrayImage [G]
			2:GrayImage [B]
RGBToColour	Create an image from individual RGB channels	0:GrayImage [R]	0:Image [colour]
		1:GrayImage [G]	
		2:GrayImage [B]	
ColourToHSI	Extract the HSI colour planes	0:Image [colour]	0:GrayImage [H]
			1:GrayImage [S]
			2:GrayImage [I]
HSIToColour	Create an image from individual HSI planes	0:GrayImage [H]	0:Image [colour]
		1:GrayImage [S]	
		2:GrayImage [I]	
ColourToOpponent	Extract the opponent process colour representation	0:Image [colour]	0:GrayImage [Red_Green]
			1:GrayImage [Blue_Yellow]
			2:GrayImage [White_Black]
ViewOpponent	Normalise (0–255) opponent process colour channels. Used to view the normalised colour (unsaturated) channels	0:GrayImage [Red_Green]	0:GrayImage [Red_Green]
		1:GrayImage [Blue_Yellow]	1:GrayImage [Blue_Yellow]
		2:GrayImage [White_Black]	2:GrayImage [White_Black]
ColourToCMY	Extract the CMY (Cyan, Magenta, Yellow) colour planes	0:Image [colour]	0:GrayImage [C]
			1:GrayImage [M]
			2:GrayImage [Y]

CMYToColour	Create an image from individual CMY (Cyan, Magenta, Yellow) planes	0:GrayImage [C]	0:Image [colour]
		1:GrayImage [M]	
		2:GrayImage [Y]	
ViewCMY	Normalise (0–255) CMY channels. Used to view the normalised colour (unsaturated) channels	0:GrayImage [C]	0:GrayImage [C]
		1:GrayImage [M]	1:GrayImage [M]
		2:GrayImage [Y]	2:GrayImage [Y]
ColourToYUV	Extract the YUV colour planes	0:Image [colour]	0:GrayImage [Y]
			1:GrayImage [U]
			2:GrayImage [V]
YUVToColour	Create an image from individual YUV planes	0:GrayImage [Y]	0:Image [colour]
		1:GrayImage [U]	
		2:GrayImage [V]	
ViewYUV	Normalise (0–255) YUV channels. Used to view the normalised colour (unsaturated) channels	0:GrayImage [Y]	0:GrayImage [Y]
		1:GrayImage [U]	1:GrayImage [U]
		2:GrayImage [V]	2:GrayImage [V]
ColourToYIQ	Extract the YIQ colour planes	0:Image [colour]	0:GrayImage [Y]
			1:GrayImage [I]
			2:GrayImage [Q]
YIQToColour	Create an image from individual YIQ planes	0:GrayImage [Y]	0:Image [colour]
		1:GrayImage [I]	
		2:GrayImage [Q]	
ViewYIQ	Normalise (0–255) YIQ channels. Used to view the normalised colour (unsaturated) channels	0:GrayImage [Y]	0:GrayImage [Y]
		1:GrayImage [I]	1:GrayImage [I]
		2:GrayImage [Q]	2:GrayImage [Q]

■ **(Continued)**

Method	Description	Inputs (Index #: data type [descriptor])	Outputs (Index #: data type [descriptor])
ColourToXYZ	Extract the XYZ colour planes	0:Image [colour]	0:GrayImage [X]
			1:GrayImage [Y]
			2:GrayImage [Z]
XYZToColour	Create an image from individual XYZ planes	0:GrayImage [X]	0:Image [colour]
		1:GrayImage [Y]	
		2:GrayImage [Z]	
ViewXYZ	Normalise (0–255) XYZ channels. Used to view the normalised colour (unsaturated) channels	0:GrayImage [X]	0:GrayImage [X]
		1:GrayImage [Y]	1:GrayImage [Y]
		2:GrayImage [Z]	2:GrayImage [Z]
ColourToLAB	Extract the Lab colour planes	0:Image [colour]	0:GrayImage [L]
			1:GrayImage [a]
			2:GrayImage [b]
LABToColour	Create an image from individual Lab planes	0:GrayImage [L]	0:Image [colour]
		1:GrayImage [a]	
		2:GrayImage [b]	
ViewLAB	Normalise (0–255) Lab channels. Used to view the normalised colour (unsaturated) channels	0:GrayImage [L]	0:GrayImage [L]
		1:GrayImage [a]	1:GrayImage [a]
		2:GrayImage [b]	2:GrayImage [b]
3D VOLUME			
DicomSave	A grey-scale volume image whose pixels shall be saved into DICOM format (*.dcm) (Double-click to activate). Requires the DICOM header file generated by *DicomRead*	0:VolumeImage	
		1:String [Path of the original DICOM header file]	

DicomRead	Extract the grey-scale volume image data from a DICOM image. The header information is also made available to be passed to *DicomSave*		0:VolumeImage 1:String [Path of the original DICOM header file] 2:String [DICOM header]
XYZviewer	Slices in a grey-scale 3D image are viewed from their X, Y and Z directions	0:VolumeImage	
IMGfrom3D	Get a slice from a 3D data set (slice is specified by user). Returns the min max pixel values from within the slice	0:VolumeImage 1:Integer [Range: 1 to the number of slices in the volume, default = 1]	0:GrayImage 1:Integer [minimum pixel value within slice] 2:Integer [maximum value within slice]
Scale3dData	Scale pixel values in a 3D grey-scale image to the range 0 – integer input	0:VolumeImage 1:Integer [Scale range required, (default = 255)]	0:VolumeImage
Thres3D	Threshold the 3D data	0:VolumeImage [Grey-scale] 1:Integer [Threshold value, default = 200]	0:VolumeImage [Binary]
Mask3D	Generate a 3D mask. Zeros a user-defined number of rows, columns and slices	0:VolumeImage [Grey-scale] 1:Integer [size of 3D mask, default = 1]	0:VolumeImage [Grey-scale]
Sobel3D	3D Sobel 3 × 3 × 1 (18-neighbourhood) edge detector	0:VolumeImage [Grey-scale]	0:VolumeImage [Grey-scale]
Blob3D	Extract the 3D blobs from binary 3D image. Each blob is assigned a grey-scale value	0:VolumeImage [Binary]	0:VolumeImage [Binary]

■ **(Continued)**

Method	Description	Inputs	Outputs
		(Index #: data type [descriptor])	(Index #: data type [descriptor])
BigestBlob3D	Extract the N (user-defined) biggest 3D blobs from 3D binary image	0:VolumeImage [Binary]	0:VolumeImage [Binary]
		1:Integer [Number of large blobs required, range 0–255, default = 1]	
Thinning3D	3D thinning operation of a binary 3D data set	0:VolumeImage [Binary]	0:VolumeImage [Binary]
MIP	Maximum intensity projection transform	0:VolumeImage	0:GrayImage
AIP	Average intensity projection transform	0:VolumeImage	0:GrayImage
PushSlice	Push (insert) an image slice into the 3D data set	0:VolumeImage	0:VolumeImage
		1: GrayImage [Image to be inserted]	
		2:Integer [slice number - between 1 and depth (default = 1)]	
		3:Integer [minimum pixel value within slice (default = 1)]	
		4:Integer [maximum pixel value within slice (default = 255)]	
RenderEngine	Surface rendering of a binary image. Image can be displayed as a cloud of points, wire frame, flat shading, Gouraund shading and Phong shading. (Double-click to activate). Allows user to translate, scale and rotate image	0:VolumeImage	0:VolumeImage

LOW LEVEL			
GetPixel	A grey-scale image from which a pixel intensity at a certain coordinate is obtained	0:GrayImage	0: Integer [intensity of the pixel at the specified coordinate]
		1: Coordinate [coordinate of the pixel in question]	
SetPixel	A grey-scale image from which a pixel at a certain coordinate is replaced with one of a user-defined intensity	0:GrayImage	0:GrayImage
		1: Integer [grey-scale intensity of the replacement pixel]	
		2: Coordinate [coordinate of the pixel in question]	
RemovePixel	A grey-scale image from which a pixel at a certain coordinate is removed (removing a pixels sets that pixel to black)	0:GrayImage	0:GrayImage
		1: Coordinate [coordinate of the pixel in question]	
DrawLine	Draw a line in the grey-scale image	0:GrayImage	0:GrayImage
		1: Coordinate [starting coordinate of the line]	
		2: Coordinate [finishing coordinate of the line]	
		3: Integer [grey-scale intensity of the line]	
DrawBox	Draw a hollow box in the grey-scale image	0:GrayImage	0:GrayImage
		1: Coordinate [upper top left]	
		2: Coordinate [lower bottom right]	
		3: Integer [grey-scale intensity]	

■ **(Continued)**

Method	Description	Inputs (Index #: data type [descriptor])	Outputs (Index #: data type [descriptor])
FillBox	Draw a filled box in the grey-scale image	0:GrayImage 1: Coordinate [upper top left] 2: Coordinate [lower bottom right] 3: Integer [fill grey-scale intensity]	0:GrayImage
DrawCircle	Draw a white hollow circle in the grey-scale image	0:GrayImage 1: Coordinate [coordinate of the centre of the circle] 2: Integer [radius]	0:GrayImage
FillCircle	Draw a white filled circle in the grey-scale image	0:GrayImage 1: Coordinate [coordinate of the centre of the circle] 2: Integer [radius]	0:GrayImage
GetImageWidth	Width of the input grey-scale image	0:GrayImage	0: Integer [width of the input grey-scale image]
GetImageHeight	Height of the input grey-scale image	0:GrayImage	0: Integer [height of the input grey-scale image]
GenerateCoordinate	Generate the coordinate value from the (x,y) components	0: Integer [x] 1: Integer [y]	0: Coordinate
GeneratePoints	Generate the (x,y) components of a given coordinate	0: Coordinate	0: Integer [x] 1: Integer [y]

STRING			
StringAdd	Combine two strings (objects)	0: Undefined [first of two strings (objects) which are to be added]	0: String [The resulting string which is made up from the two input strings]
		1: Undefined [second of two strings (objects) which are to be added]	
StringToLowerCase	A string which shall be converted to lower case	0: String	0: String
StringToUpperCase	A string which shall be converted to upper case	0: String	0: String
MATH	Library of standard mathematical operators		
JAIColour	See the Java^{TM} Advanced Imaging website: https://jai.dev.java.net/		
OSMIA – Tina 5 Interface	See http://www.neatvision.com/ for details on interfacing NeatVision with Tina 5.0		

[a] *Left click and hold* to draw the ROI, then release when complete

[b] The user inputs a polygon by *left-clicking* a series of points (marked in red). When the user clicks a point within 4 pixels of the start point or alternatively *right-click* to finalise and close the polygon. Once closed, the polygon will be displayed in green. To begin a new polygon, use *shift-click*.

[c] Some of these functions use data types/variables that are for internal NeatVision use *only*. Access to such data (e.g., pixel access) is can be done directly in Java

NeatVision contains over 300 image manipulation, processing and analysis algorithms. Users can extend the core NeatVision library using the developer's interface, a plug-in which features, automatic source code generation, compilation with full error feedback and dynamic algorithm updates. NeatVision is primarily an image-processing application and offers an extensive range of image analysis and visualisation tools (these include zoom, pseudo colour, intensity scan, histogram and 3D profile mesh). In addition, the ability to read and write a wide range of image file formats is supported.

22.9.2 Downloading

NeatVision can be downloaded and evaluated for 30 days prior to registration. Users can register online and a unique username and password will be automatically issued. To download NeatVision, visit: http://www.neatvision.com/, or the publisher's web site http://extras.springer.com/2012/978-1-84996-169-1.

NeatVision is distributed in two forms. The standard version is distributed as afree full functional product. NeatVision can be downloaded and evaluated for 30 days prior to registration. Users can register online and a unique username and password will be automatically issued.

Registered users can upgrade to the developer's version. This release of NeatVision includes a fully integrated development environment allowing users to to develop their own Java code and image analysis blocks for use within the NeatVision environment, thus extending the extensive range of built-in algorithms. This is achieved using the full power the Java Programming language from Sun Microsystems, the Java AWT Imaging, Java 2D API, Java Advanced Imaging and the basic NeatVision image class approach. This includes a development wizard to allow ease of integration of custom programs or existing JAI programs into the NeatVision environment, providing an open environment for the development, integration and distribution of imaging functions.

22.9.3 Developing Additional NeatVision Components

Registered users can also upgrade to a developer's version, which supports algorithm development based on Java AWT Imaging, Java 2D Imaging and Java Advanced Imaging. The developer's update is compatible with all versions of NeatVision.

22.10 Conclusions

NeatVision was designed to allow novice and experienced users to focus on the machine vision design task rather than concerns about the subtlety of a given programming language. It allows machine vision engineers to implement their ideas in a dynamic and straightforward manner. NeatVision standard and developers versions are freely available via the Internet and are capable of running on a wide range of computer platforms (e.g. Windows, Solaris, Linux).

Acknowledgements

While NeatVision is primarily a collaboration of researchers within the Vision Systems Group in Dublin City University (Ireland), we would also like to acknowledge all those who have supported the NeatVision project through the submission of algorithms for inclusion or for their constructive and useful feedback. Development of the MATLAB compatible *VSG Image Processing & Analysis Toolbox* which offers MATLAB users NeatVision functionality was funded in part by HEA PRTLI IV *National Biophotonics and Imaging Platform Ireland* (NBIPI) [14].

References

1. NeatVision: image analysis and software development environment. http://www.neatvision.com. Accessed July 2010
2. Sage D, Unser M (2003 Nov) Teaching Image Processing Programming in Java. IEEE Signal Processing Magazine 43–52
3. Khoros: Khoral Research, Inc. http://www.khoral.com. Accessed July 2010
4. WiT: logical vision, Available at http://www.logicalvision.com. Accessed July 2010
5. MathWorks: Matlab. http://www.mathworks.com. Accessed July 2010
6. VSG image processing and analysis toolbox (VSG IPA TOOLBOX beta). http://www.cipa.dcu.ie/code.html. Accessed July 2010
7. Whelan PF, Molloy D (2000) Machine vision algorithms in Java: techniques and implementation. Springer, London
8. NeatVision: users guide. http://neatvision.eeng.dcu.ie/user.html. Accessed July 2010
9. JAI: The Java Advanced Imaging (API). http://java.sun.com/products/java-media/jai 2005. Accessed 25 Oct 2005
10. Barron JL, Fleet DJ, Beauchemin S (1994) Performance of optical flow techniques. Int J Comput Vision 12(1):43–77
11. OSMIA – open source medical image analysis, EU fifth framework programme (IST: accompanying measures). http://www.eeng.dcu.ie/~whelanp/osmia/. Accessed July 2010
12. NeatVision: developers guide. http://neatvision.eeng.dcu.ie/developer.html. Accessed July 2010
13. Ghita O, Whelan PF (2002) A computationally efficient method for edge thinning and linking using endpoints. J Electron Imaging 11(4):479–485
14. National biophotonics and imaging platform Ireland (NBIPI).http://www.nbipireland.ie/. Accessed July 2010

Further Reading

Robert JT, Sadleir PF, Whelan Padraic MacMathuna, Fenlon HM (2004) Informatics in radiology (infoRAD): portable toolkit for providing straightforward access to medical image data. Radiographics 24(4):1193–1202

Whelan PF, Sadleir RJT (2004) A visual programming environment for machine vision engineers. Sensor Rev 24(3):265–270

Whelan PF, Sadleir RJT (2004) NeatVision – visual programming for computer aided diagnostic applications. Radiographics 24(6):1779–1789

23 Intelligent Image Processing Using Prolog

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_23,

Abstract: Some potential applications of Machine Vision, particularly those involving highly variable objects (➲ Chap. 2), require greater reasoning power than standard image processing languages can provide. This chapter is based on the premise that an intelligent vision system can be made, by the simple expedient of embedding image manipulation and measurement routines, such as those provided by QT (➲ Chap. 21), within the Artificial Intelligence language Prolog. The great strength of Prolog is its ability to perform symbolic reasoning. It is a declarative language. That is, it operates by searching automatically for a solution to a problem, given a description of (part of) the world. The user specifies the desired characteristics that a solution must have. Prolog's search engine then hunts for the specific conditions that must exist to obtain a solution. If the user chooses, or programs Prolog appropriately, it will search for multiple solutions. Unlike a conventional imperative language, Prolog does not have any instructions. Instead, it accepts, a series of statements that are then tested individually for their veracity. It does not possess any of the usual flow-control tools, such as IF ... THEN, FOR, WHILE, CASE, etc. Instead, it relies on back-tracking and recursion. The former allows temporary assumptions that have been made while searching for a solution to be revised before continuing the search. A software interface interconnecting QT with a modern implementation of Prolog (SWI-Prolog) was designed. The result, called PQT, allows image processing operations to be invoked procedurally within a Prolog program, or executed interactively, as in QT. Image measurement values can be used to define temporary values for (i.e. instantiate) Prolog variables. Using PQT, it is possible to define general spatial relationships (left, above, concentric, parallel, etc.), geometric features (T, X, U, V), shape characteristics (round, curved, elongated), colour and texture (yellow, smooth, spotty). These would be much more difficult to define and use in a standard programming language. PQT can also be used effectively to control external peripheral devices but not in real time. PQT inherits Prolog's limited ability to analyse and extract meaning from natural language sentences, such as spoken commands. While PQT was designed to perform vision-based tasks, such as inspecting food products, it is also capable of the high-level reasoning needed to incorporate them into a well-balanced meal.

This chapter is in three parts. In the first, we explain how certain tasks involving abstract symbolic reasoning about images, can be accomplished, using a Prolog program with embedded image processing commands. In the second, we discuss the processing of natural language using Prolog. This allows us to build a vision system that is able to understand spoken commands expressed in English. In addition, such a machine can perform such tasks as operating an array of lights, or controlling a simple robot. In the third part, we describe how Prolog and QT, the image processing software package discussed in ➲ Chap. 21, can be interfaced to one another. This combination implements a convenient, easy-to-use language that has the ability to process and analyse images and then reason logically about them.

23.1 Part 1: Basic Features of Prolog

23.1.1 Adding Intelligence

Many potential applications of machine vision require greater reasoning power than the image processing languages described earlier can provide. This chapter is founded on the premise that an intelligent vision system can be made, by the simple expedient of embedding image manipulation and measurement routines within the AI language Prolog. The top-level control language for such a system will be called *Prolog+*. One of the most recent implementations of Prolog+ was constructed by interfacing QT (➤ Chap. 21) to SWI-Prolog [1] and will therefore be called *PQT*. We shall see later that Prolog+ is able to solve some of the problems that do not yield to standard image processing techniques (➤ Chap. 14) operating alone. (QT should not be confused with Qt which is a cross-platform application and user-interface framework, produced by Nokia. URL http://qt.nokia.com/products/.)

Prolog was devised specifically for developing AI programs and is used extensively for tasks involving Natural Language processing [2], machine learning [3] and for building rule-based and expert systems [4]. The range of applications is impressive in its breadth and includes areas as diverse as financial risk assessment, bridge design, planning health care, designing computer systems, route planning, symbolic mathematical manipulation [3, 5].

Prolog is an unusual computer language and bears no real resemblance to any of the "conventional" languages of Computer Science. In the core language, there are no FOR loops, IF . . . THEN statements, CASE statements, or any of the usual flow-control constructions. There is no variable assignment process; Prolog does not even allow instructions. The reader is, therefore, warned against trying to reconcile the distinctive features of Prolog with his/her prior programming experience. *Prolog is different!* In the following section, we describe this fascinating language for readers with no prior knowledge; we shall try to explain *why* Prolog is important. However, to obtain a more comprehensive understanding, the uninitiated reader is urged to refer to the standard textbooks. [4, 6, 7] Later, we shall attempt to justify the choice of Prolog, by describing and listing PQT programs that solve some interesting vision problems. For the moment, let it suffice to say that many major benefits have been obtained through the use of Prolog that could not have been obtained nearly so easily using other languages.

From his own experience, which now stretches over more than 20 years, the author can report that no serious shortcomings of Prolog+ have come to light. PQT combines SWI-Prolog with the proven set of image processing utilities embodied in QT. In Prolog+, it is possible to define abstract relationships/attributes such as *left_of*, *above*, *bigger*, *between*, *beside*, *near*, *yellow*, *curved*, *spotty*, etc. Representing these concepts in more popular languages, such as Java, C, C++, etc. is much more difficult.

Experienced Prolog programmers will appreciate that Prolog+ is a super-set of standard Prolog, with numerous additional built-in predicates that perform image processing functions. These behave much like the standard *write/1* predicate; by trying to satisfy a goal, the desired effect (e.g., negating, thresholding or filtering an image) is actually achieved. Predicates that calculate numeric values (e.g., Euler number, blob count, etc.) instantiate variables in the normal way. Readers who are familiar with Prolog may wish to omit ➤ Sect. 23.2 and skip directly to ➤ Sect. 23.3.

23.1.2 Introducing Prolog

It took the author 2 years to understand *why* Prolog is important and just 2 weeks to become reasonably proficient at using it! The reason why Prolog is worth further study is simply that it allows a novel and very natural mode of programming to be used. It permits a person to state the nature of an acceptable solution to a given search-based task, rather than *how* to find one. To understand this, consider the task of finding a marriage partner for a given hypothetical man. (*Not the author!*) It is relatively easy to specify the basic "requirements" to a dating agency, in terms such as those listed below. (The following list and Prolog program are intended to illustrate a point about Prolog; they do not constitute any statement about the absolute desirability of any type of religion, or personal/racial characteristics.)

Sex	Female
Age	[45,55]
Height	[150,180]
Weight	[45,75]
Language	English
Desirable characteristics	List might cover personality, faith, interests, hobbies, life-style, etc.

Clearly, this list cannot guarantee success in romance but it is sufficiently detailed to allow us to illustrate the general principles of what is known as *Declarative Programming*. Conventional programming languages are, by contrast, termed *imperative*, since they consist of a sequence of instructions. Prolog "programs" consist of a set of logical tests, rather than commands.

Writing a Prolog program to find a wife/husband is straightforward and has actually been used by at least one commercial dating agency. Here is a program to find a suitable wife, using the very small number of criteria specified above:

```
suitable_wife(X):-
        person(X),                 % Find a person called X
        sex(X,female),             % Is person X female?
        age(X,A),                  % Find age, A, of person X
        A >= 45,                   % 45 or more years old?
        A =< 55,                   % 55 or fewer years old?
        height(X, H),              % Find height, H, of person X
        H >= 150,                  % Is X at least 150 cm tall?
        H =< 180,                  % Is X at most 180 cm tall?
        weight(X,W),               % Find the weight of person X
        W >= 45,                   % Weight >= 45 kg?
        W =< 75,                   % Weight =< 75 kg?
        speaks(X,English).         % Does X speak English?
        must_be(X,[christian,kind, truthful,generous,loving,loyal]).    % Obvious meaning
```

Given such a program and a set of stored data about a collection of people, Prolog will search the database, to find a suitable match. There is no need to tell Prolog how to perform the search. The reader is urged to consider rewriting the program using Basic, C, Fortran, Java, or Pascal. An imperative-language program will take much longer to write, will consist of many more lines of code and be less transparent, because it imposes an unnatural mode of thought on the problem. Declarative (Prolog) programming is very much more natural in its style and allows a person to think directly about the type of solution required, rather than how to find it. Prolog differs from most other computer languages in several very important ways:

(a) Firstly, a Prolog "program" does *not* consist of a sequence of instructions, as routines written in conventional languages do. (The correct term is "application", since a program is, strictly speaking, a sequence of instructions. However, we shall continue to use the term "program", since this is more familiar to most readers.) Instead, it provides a medium for describing (part of) the world. As we have already stated, Prolog is referred to as a *declarative* language, while most other computer languages, military orders, knitting patterns, automobile repair manuals, musical scores and culinary recipes are all examples of *imperative* languages. This is a vital difference, which distinguishes Prolog and a small group of related languages from the better known conventional languages of Computer Science.

(b) The "flow of control" in a Prolog program does not follow the normal convention of running from top to bottom. In a Prolog program, the flow is very likely to be in the reverse direction, through a mechanism called *back-tracking*.

(c) Through the use of back-tracking, it is possible to make and subsequently revise temporary assignments of values to variables. This process is called *instantiation* and is akin to re-evaluating assumptions made earlier. (The word "*instantiation*" is derived from the same linguistic root as "instance". Prolog tries to find an *instance* of some variable(s) which cause the given predicate to be true.) Instantiation is performed, in order to try and prove some postulate, theorem or statement, which may or may not be true. As far as Prolog is concerned, theorem proving (also called *goal satisfaction*) is the equivalent process to running or executing an imperative language program.

(d) It is possible to make very general statements in Prolog in a way that is not possible in most other languages. We shall see more of this feature later, but for the moment, let us illustrate the point with a simple example. In Prolog it is possible to define a relationship, called *right* in terms of another relationship, called *left*.

In English:	"A is to the *right* of B if B is to the *left* of A."
In Prolog:	*right(A,B) :- left(B,A).*

(Read "**:-**" as "*can be proved to be true if*" or more succinctly as "*if*"). Notice that neither A nor B have yet been defined. In other words, we do not need to know what A and B are in order to define the relationship *right*. For example, A and B might be features of an image such as blob centres, or corners. Alternatively, A and B may be objects on a table, people (*Hitler* and *Lenin*) or policies (*National Socialism* and *Marxism*). The point to note is that Prolog+ allows the relationships *left* and *right* to be applied to any such objects, with equal ease. (Is *hitler* to the right of *lenin*? In the political sense, "yes", while the answer is "no", when we consider at the layout of words on this page.)

(e) Prolog makes very extensive use of *recursion*. While, Java, C and many other imperative languages also allow recursion, in Prolog it forms an essential control mechanism.

Prolog is not normally recommended for writing programs requiring a large amount of numerical manipulation. Nor is Prolog appropriate for real-time control, or other computational processes requiring frequent processing of interrupts.

23.1.2.1 Sample Program

The following program deals with the ancestry and ages of members of two fictitious families. (Not the author's!) Remember that ":-" means "*is true if*", while comma (",") means *AND*. Single-line comments are prefixed with %, while multi-line comments begin with /* and are terminated thus */.

```
/*
The following facts specify in which years certain people were born. Interpretation:
            born(roger,1943)
means that
            "roger was born in 1943".
                                                                        */
born(roger,1943).
born(susan,1942).
born(pamela,1969).
born(graham,1972).
born(thomas,1953).
born(angela,1954).
born(elizabeth,1985).
born(john,1986).
born(marion,1912).
born(patricia,1911).
born(gertrude,1870).
born(david,1868).
/*
These facts describe the parent-child relationships that exist in the families. Interpretation:
parent(X,Y) means that
              "X is a parent of Y".
                                                                        */
parent(roger,pamela).
parent(roger,graham).
parent(patricia,roger).
parent(anne,patricia).
parent(david,patricia).
parent(marion,susan).
parent(susan,graham).
parent(susan,pamela).
parent(thomas,john).
parent(angela,john).
parent(thomas,elizabeth).
parent(angela,elizabeth).
/*
Defining a relationship called "child". Read this as follows:
              "A is a child of B if
                          B is a parent of A."
                                                                        */
```

```
child(A,B) :- parent(B,A).
/*
Defining a relationship called "older". Read this as follows:
            "A is older than B if
                      the age of A is X AND
                      the age of B is Y AND
                      X > Y".
                                                                        */
older(A, B) :-
            age(A,X),                   % Age of A is X
            age(B,Y),                   % Age of B is Y
            X > Y.                      % Is X greater than Y?

/*
Defining a relationship "age". Read this as follows:
            "A has age B if
                      A was born in year X AND
                      it is now year Y AND
                      X =< Y AND
                      B is equal to Y – X."
                                                                        */
age(A,B) :-
            born(A,X),
            date(Y,_,_),
            X =< Y,
            B is Y - X.
/*
The definition of "ancestor" has two clauses. Prolog always tries to satisfy the top one first.
If this fails, it then tries to satisfy the second clause.
Interpretation: ancestor(A,B) means that
            "A is an ancestor of B."
The first clause should be interpreted as follows:
            "A is an ancestor of B if A is a parent of B".
                                                                        */
ancestor(A,B) :- parent(A,B).
/*
The second clause should be interpreted as follows:
            "A is an ancestor of B if
                      A is a parent of Z AND
                      Z is an ancestor of B."
Notice the use of recursion here.
                                                                        */
ancestor(A,B) :-
          parent(A,Z),
          ancestor(Z,B).
/*
Definition of "print_descendents".
```

```
This uses backtracking to find all possible solutions.
The first clause always fails but in doing so it prints the descendants and their dates of birth.
____________________________________________________________*/
print_descendents(A) :-
          nl,                              % New line
          write('The known descendants of' ),
                                           % Print a message
          write(A),                        % Print value of A
          write(' are:' ),                 % Print a message
          ancestor(A,Z),                   % A is ancestor of Z
          born(Z,Y),                       % Z was born in year Y
          nl,                              % New line
          tab(10),                         % 10 white spaces
          write(Z),                        % Print value of Z
          write(', born ' ),               % Print a message
          write(Y),                        % Print value of Y
          fail.                            % Force back-tracking
% Second clause always succeeds and prints a new line
print_descendents(_) :- nl.
```

23.1.2.2 Queries

A Prolog program consists of a set of *facts* and *rules* for interpreting them. The user then presents a *query*, which Prolog tries to prove to be true. The following goal was satisfied, because the program contains facts that explicitly state it to be true.

Query: *born(susan, 1942)*
Result: *YES*

In the following query, X is a variable. The query tests the proposition that susan was born some time (X). The goal succeeds, even if we refuse to accept that X = 1942.

Query: *born(susan, X)*
Result: *X = 1942*
YES

It is possible to use a variable for the first argument, to find who was born in 1942.

Query: *born(X, 1942)*
Result: *X = susan*
YES

It is also possible to use two or even more variables to specify very general queries:

Query: *born(X, Y)*

(*X* and *Y* are both variables, since their names have capital initial letters.) There are several possible solutions to this query.

Result: *X = roger*
Y = 1943

X = susan
Y = 1942

X = pamela
Y = 1969

X = graham
Y = 1972

X = thomas
Y = 1953

X = angela
Y = 1954

X = elizabeth
Y = 1985

X = john
Y = 1986

X = marion
Y = 1912

X = patricia
Y = 1911

X = gertude
Y = 1870

X = david
Y = 1868

NO MORE SOLUTIONS
(Notice the alternative solutions generated by this general query.)

The following query is satisfied by using the simple rule for *age/2* given above.

Query: *age(marion, Z)*
Result: *Z = 77*
YES

However, the following query uses a higher-level rule; *older/2* is defined in terms of *age/2* which is itself defined using *born/1*. (It is customary to specify the *arity* (number of arguments) of a Prolog predicate in the following way: *predicate_name/arity.)*

Query: *older(marion, susan)*
Result: *YES*

It is not always possible to satsify a given goal.

Query: *older(susan, marion)*
Result: *NO*
(*NO* should properly be expressed as *NOT PROVEN.*)

Another query which can only be solved by using a rule (*child*), which relies on another explicit relationship (parent):

Query: *child(susan, Z)*
Result: *Z = marion*
YES

The following is an example of a query that is satisfied by using a recursive rule (i.e., *ancestor*, clause 2) This is tantamount to asking who is a descendent of susan.

Query: *ancestor(susan, Z)*
Result: *Z = graham*
Z = pamela
NO MORE SOLUTIONS
(Notice the alternative solutions.)

The following query finds all of the ancestors of a given object (We know that *graham* is likely to be a person, but Prolog does not, as we have not explicitly told it so.)

Query: *ancestor(Z, graham)*
Result: *Z = roger*
Z = susan
Z = patricia
Z = anne
Z = david
Z = marion
NO MORE SOLUTIONS

Finally, here are two queries involving a high-level predicate:

Query: *print_descendents(marion)*
Result: *The known descendants of marion are:*
susan, born 1942
graham, born 1972
pamela, born 1969
YES

Query: *print_descendents(anne)*
Result: *The known descendants of anne are:*
patricia, born 1911

```
            roger, born 1943
            pamela, born 1969
            graham, born 1972
        YES
```

23.1.2.3 Back-tracking

Consider the following program, which prints all of the grandparent-grandchild pairs that can be inferred from a *parent/2* database like that given above.

```
1.  go :-                % Starting the definition of the first clause
2.      parent(A,B),     % parent/2 is defined above. Find one parent-child pair.
3.      parent(B,C),     % Find child of B (i.e., grand-child of A)
4.      write([A,C]),    % Writes one of the solutions i.e., a [grandparent,grandchild] pair.
5.      nl,              % New line
6.      fail.            % First clause ends. It fails after all solutions have been found

7.  go :-                % Second clause. Is tried when clause 1 fails
8.      write('The end'), % Writes a simple message
9.      nl.              % New line. This clause and the goal "go" always succeed here
```

The goal *go* always succeeds. Clause 1 is evaluated first. In line 2, a parent-child pair (A,B) is found. This value of the variable B is used in line 3, as Prolog seeks a child (C) of B. (C is, of course a grandchild of A.) In line 4, the pair [A,C] is printed, while line 5 simply prints a new line. Line 6 always fails. In Prolog, goal failure is not catastrophic; it simply forces Prolog to begin back-tracking, as it continues its search for other possible solutions. Line 5 is then re-evaluated, but the built-in predicate *nl/0* is not resatisified. Back-tracking therefore continues to line 4. The built-in predicate *write/1* is not resatisified either, so back-tracking continues to line 3. If B has more that one child, the variable C is re-instantiated in line 3. In this case, lines 4, 5 and 6 are evaluated next. This cycle is repeated until all of the children of B have been found. Once this happens (i.e., line 3 has exhausted all solutions for a given value of B), back-tracking returns to line 2. Here, a new pair [A,B] is found and the whole process described so far is repeated. (A and/or B may change as a result of this re-instantiation.) Eventually, line 2 runs out of possible solutions. When this happens, clause 2 fails and Prolog begins evaluating clause 2, which in this case always succeeds.

When writing Prolog programs, it is not normally necessary to pay much attention to the possibility of back-tracking. However, an experienced Prolog programmer is often able to structure the program so that it searching is performed efficiently. One way to do this is to "freeze" all instantiations, once one partial solution has been found. This is accomplished using a device known as the *cut*, which is represented in Prolog by the operator "!". ➲ *Figure 23.1* explains the flow of control during backtracking and cut (1) using the so-called Box Model. This is a representation of Prolog execution, in which we think in terms of *procedures* rather than *predicates*. Each sub-goal is represented by a box with four ports:

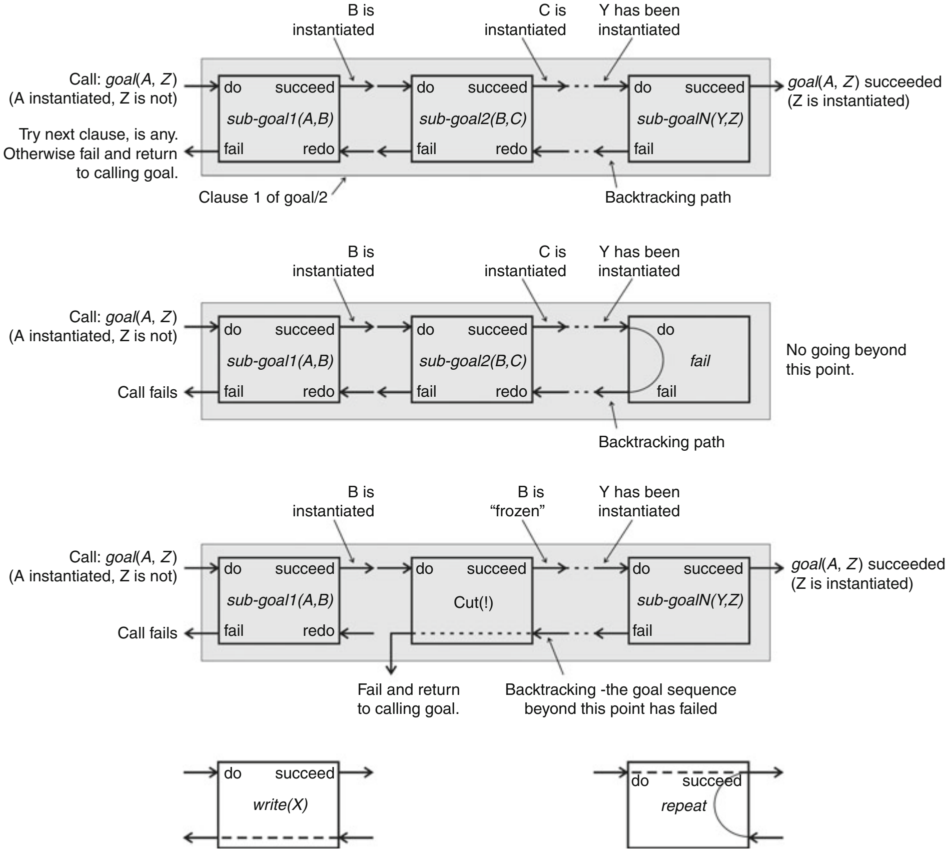

B is instantiated
C is instantiated
Y has been instantiated
Call: goal(A, Z) (A instantiated, Z is not)
do
succeed
sub-goal1(A,B)
fail
redo
sub-goal2(B,C)
sub-goalN(Y,Z)
goal(A, Z) succeeded (Z is instantiated)
Try next clause, is any. Otherwise fail and return to calling goal.
Clause 1 of goal/2
Backtracking path
Call fails
No going beyond this point.
B is "frozen"
Cut(!)
Fail and return to calling goal.
Backtracking -the goal sequence beyond this point has failed
write(X)
repeat

Fig. 23.1

Box model of flow in a Prolog program. (a) Sequence of sub-goals not containing *fail* or cut (!). (b) Sequence containing *fail* but not cut (!). (c) Sequence containing cut (!) but not *fail*. (d) Model for printing by the built-in predicate *write/1*. The goal is not resatisfied on backtracking. Control continues to flow backwards. (e) Model for the built-in predicate repeat/0. The goal is satisfied on first meeting *repeat* and is always resatisfied on backtracking

Do Input port. This receives data from the previous sub-goal and then tries to satisfy the sub-goal.

Succeed Output port. When the sub-goal succeeds, previously and newly instantiated variables are sent to the next sub-goal.

Fail Output port. No new instantiations have been made. Return to the previous sub-goal. That is back-track one step.

Redo Input port. Receives control when the following subgoal fails.

The box representing the Built-in Predicate *fail* does not have *Succeed* or *Redo* ports. The box representing cut (!) directs the output from its *Fail* port to the calling goal/query, when it receives an input from the following sub-goal, without performing any other action. Any other clauses are ignored. When cut (!) receives a *Do* input, it freezes all instantiations in the clause.

23.1.2.4 Recursion

Prolog provides just two methods of controlling program "flow": back-tracking and recursion. We have already encountered recursion, in the definition of *ancestor/2*. Prolog handles recursion very efficiently and is one of the main reasons why it is so powerful. In the following example, *left_of/2* builds on the predicate *left/2*, whose meaning is obvious and which will be defined later.

```
% A is left_of B if A is to (the immediate) left of B
left_of(A,B) :- left(A,B).                     % Terminating clause

% A is left_of B if A is to (the immediate) left of C and C is left_of B
left_of(A,B) :- left(A,C), left_of(C,B).       % Recursive clause
```

Structurally, this is similar to *ancestor/2*, although it should be understood that there are many other ways to employ recursion. One of the dangers of using recursion is the possibility of creating programs that never halt. However, an experienced programmer avoids this trap by following one simple rule: place a terminating clause above the recursive clause.

Why is recursion so important? The reason is that many operations leading to "intelligent" behaviour are easily naturally expressed in terms of recursion, rather than iteration. The process of packing 2D shapes is a good example; the procedure is simple:

(a) Define the space available for placing an object.
(b) Choose an object to fit in place. This involves optimising its position and possibly its orientation and/or posture ("heads up" or "tails up").
(c) Put the object into its chosen place and calculate the space that is unoccupied.
(d) Apply steps (a) to (d) recursively.

As we shall see later, packing is one of the tasks that Prolog+ is able to perform but which presents major problems for QT, or imperative languages such as C, Java, etc.

23.1.3 Prolog+

The reader is reminded that Prolog+ is an extension of standard Prolog, in which a rich repertoire of image processing functions is made available as a set of built-in predicates. For convenience, we shall employ QT's image processing mnemonics in our description of Prolog+. This results in the language PQT that we shall discuss hereafter in this chapter. In the past, other implementations of Prolog+ have been devised [8]. PQT is simply the latest of these. *Figure 23.2* shows the basic structure of a generic Prolog+ system.

The way that the image processing predicates operate follows the standard pattern established for printing in Prolog (c.f. *nl, write, tab*). That is, they always succeed but are never resatisfied on back-tracking. For example, as a "side effect" of trying to satisfy the goal *neg*, Prolog+ calculates the negative of QT's current image. Similarly, the goal *thr(125,193)* always succeeds and in doing so thresholds the current image. Again, this goal is never resatisfied on back-tracking. When X is initially uninstantiated, the goal *cwp(X)* succeeds and instantiates X to the number of white pixels in the current image. However, the goal *cwp (15294)* will only succeed if there are exactly 15294 white pixels in the current image; it will fail otherwise see (*Fig. 23.3*).

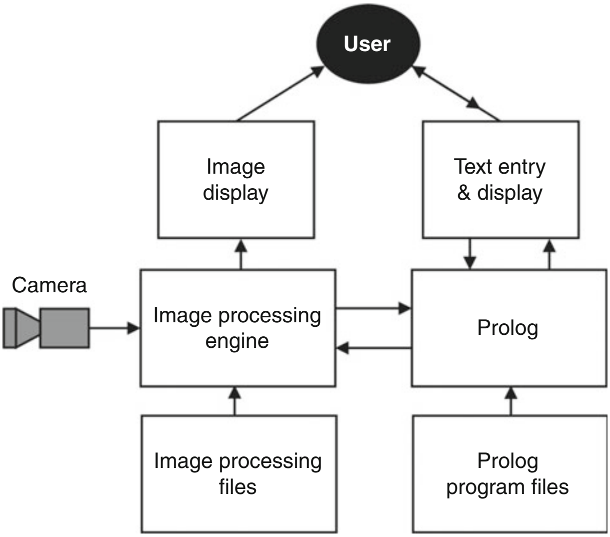


Fig. 23.2
Basic structure of a Prolog+ system. This may be implemented using one, or two, computers, or one computer, hosting Prolog, and a hardware image processing module

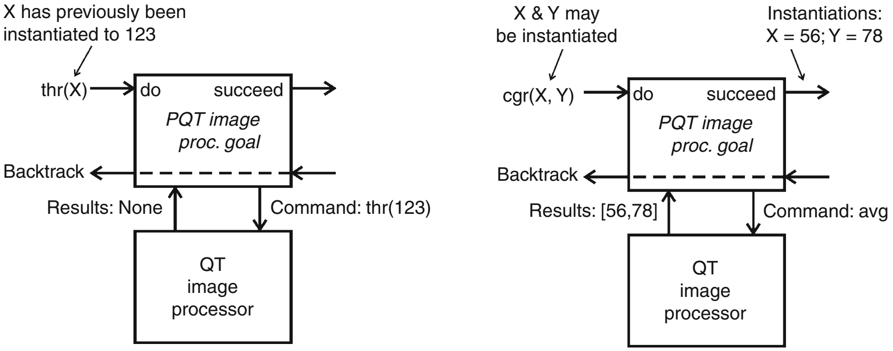


Fig. 23.3
Box model for PQT image processing predicates. (a) Image processing operations (e.g., *thr/1*) that do not return results. The goal is not resatisfied on backtracking. Control continues to flow backwards. (b) Model for PQT image processing operations (e.g., *cgr/2*) that return results. The goal is not resatisfied on backtracking. Control continues to flow backwards. If the results calculated by QT do not match the instantiations of X and Y, the goal will fail and backtrack to the previous goal

The distinction between assignment in an imperative language and instantiation in Prolog is important and must be understood before the merits of Prolog+ can be appreciated.) While PQT is trying to prove the compound goal

avg(Z),thr(Z)

Z is instantiated to the average intensity within the current image (calculated by *avg*). This value is then used to define the threshold parameter required by *thr*. With these points in mind, we are ready to examine our first PQT program.

```
grab_and_threshold :-
        grb,                    % Digitise an image
        caf,                    % Blur (Low-pass filter)
        avg(Z),                 % Calculate average intensity
        thr(Z).                 % Threshold at average intensity
```

Since each sub-goal in this simple program succeeds, the effect is the same as if we had written a sequence of commands using a conventional (i.e., imperative) computer language. Here is the QT equivalent M-function

```
        function grab_and_threshold
        grb                     % Digitise an image
        caf                     % Blur (Low-pass filter)
        Z = avg(Z);             % Calculate average intensity
        thr(Z)                  % Threshold at average intensity
```

The Prolog+ goal *grab_and_threshold* always succeeds. However, the following PQT program is a little more complicated.

```
big_changes(A) :-
        repeat,                 % Always succeeds on backtracking
        grb,                    % Digitise an image from the camera
        raf,                    % Perform raf (low-pass filter)
        sca(3),                 % Retain 3 bits of each intensity value
        rei,                    % Read image stored in previous cycle
        swi,                    % Switch current & alternate images
        wri,                    % Save the image for the next cycle
        sub,                    % Subtract the two images
        abv,                    % Absolute value of intensity
        thr(10),                % Threshold at intensity level 10.
        cwp(A),                 % A is number of white pixels
        A > 100.                % Are image differences significant?
```

The built-in predicate *repeat* succeeds, as does each of the image processing operators, [*grb*, *raf*, . . ., *cwp(A)*]. If the test *A* > *100* then fails, the program back-tracks to *repeat*, since none of the image processing predicates is resatisfied on backtracking. (The predicate *repeat/0* is always resatisfied on backtracking.) Another image is then captured from the camera and the whole image processing sequence is repeated. The loop terminates when *A* exceeds 100. When this happens, the goal *big_changes(A)* succeeds and *A* is instantiated to the number of white pixels in the difference image. The goal *big_changes(A)* performs the processing sequence [*grb*,. . ., *cwp(A)*] an indefinitely large number of times and only succeeds when two consecutive images are found that are significantly different from one another. This program could be used as the basis for a crude intruder alarm that detects when a large object (a person) enters a restricted area. By adjusting the number in the final sub-goal (i.e., *A* > *100*), it is possible to tolerate small changes (e.g., a cat wandering in front of the camera), while still being able to detect large variations due to a person being in view. By now, the reader should be able to write a QT program to perform the same function.

Although image processing commands, such as *grb, thr, cwp* etc. are always satisfied, errors will be signalled if arguments are incorrectly specified. Since *thr(X)* requires one numeric argument, an error condition will arise if X is uninstantiated. On the other hand, the following compound goal is satisfied

```
X is 197, thr(X),
```

as is

```
gli(A,B),                  % Lower & upper intensity limits
C is (A+B)/2,              % Average A and B
thr(C).                    % Threshold at average value
```

The compound goal

```
X is 186, Y is 25, thr(X,Y)
```

fails, since *thr* fails when its second argument is less than the first one. (The image processor signals an error, which causes the failure of the Prolog goal, *thr(186,25).*) The compound goal

```
X is 1587, thr(X)
```

fails, because the parameter (X) is outside the range of acceptable values, i.e., [0,255].

Notice that *thr* has already been used with different numbers of arguments. We can use *thr* with 0, 1 or 2 arguments. Other image processing predicates will be treated in the same way. For example, we have already used *wri/1* (write image to disc) and *rei/1* (read image from disc) without arguments. We can also use arguments instantiated to a string of alpha-numeric characters. Such arguments may be generated according to the usual Prolog conventions. For example, the Prolog symbol generator, *gensym/2*, may be used to create a series of file-names, *image_file1, image_file2,...* as the following illustration shows:

```
process_image_sequence :-
        grb,                            % Digitise an image
        process_image,                  % Process the image
        gensym(image_file,X),           % Generate new symbol name
        wri(X),                         % Write image
        process_image_sequence.         % Repeat processing
```

Notice here that we have "condensed" almost all of the image processing into the subsidiary predicate, *process_image/0.* Using this approach, a simpler, revised version of *big_changes/1* may be defined using the subsidiary predicate, *process/0.*

```
big_changes(A) :-
        process      % Listed below
        cwp(A),      % Instantiate A to number of white pixels
        A > 100.     % Are differences between images large enough? If not, back-track.
```

where *process/0* is defined thus:

```
process :-
        grb,            % Grab an image from the camera
        lpf, lpf, lpf,  % Low pass filter applied three times
```

```
          sca(3),        % Reduce number of bits representing intensity to 3
          rei,           % Read image from disc
          swi,           % Interchange the Current and Alternate images
          wri,           % Save image on disc
          sub,           % Subtract current and alternate images
          abv,           % "Absolute value" (fold intensity scale around mid grey)
          thr(1).        % Threshold at level 1
```

The use of subsidiary predicates, such as *process_image* and *process*, allows the programmer to think at a higher conceptual level and to defer deciding exactly what image processing is to be performed until later.

23.1.4 Sample Programs

Now that we have illustrated the basic principles of Prolog+, we are able to consider more advanced programs to illustrate its main features. It is important to realise that we must always use Prolog+ to describe the *image* generated by the camera, not the object/scene being inspected. The importance of this point cannot be over-emphasised. Additional points of general interest will be discussed as they arise.

23.1.4.1 Recognising Bakewell Tarts

Figure 23.4 which shows a small cake, called a Bakewell tart. We shall explain how Prolog+ might be used to inspect it from an overhead view.

The top level of the program consists of four separate tests, incorporated into the predicate *bakewell_tart/0:*

```
     bakewell_tart :-
          segment_image,          % Convert image as in Fig. 23.4
          outer_edge,             % Check the outer edge
          cherry,                 % Check the cherry
          icing.                  % Check the icing
```

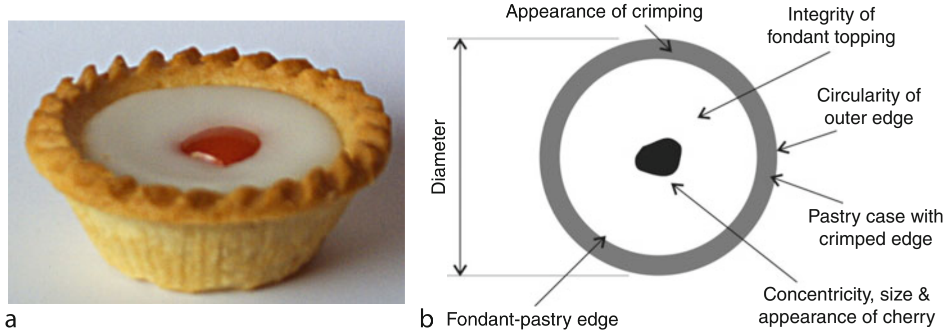

a
b
Appearance of crimping
Integrity of fondant topping
Circularity of outer edge
Diameter
Pastry case with crimped edge
Concentricity, size & appearance of cherry
Fondant-pastry edge

Fig. 23.4
Bakewell tart. (a) Photograph (Credit: B. R Wheatley, *copyright free*) and (b) critical features

Programs written in PQT are almost invariably written from the top level downwards. In this instance, *bakewell_tart/0* was the first predicate to be defined. Notice that there are four obvious stages in verifying that the tart is a good one:

(a) Simplify the image (possibly with 2–4 intensity levels), using *segment_image/0.*
(b) Check the integrity of the outer edge, using *outer_edge/0.*
(c) Check the presence, size and placing of the cherry, using *cherry/0.*
(d) Check the icing, using *icing/0.*

Clearly, *bakewell_tart/0* is only satisfied if all four of the subsidiary tests succeed. Although the secondary predicates are not defined yet, this is not necessary for us to understand the broad concepts involved in recognising a Bakewell tart. Three of these are defined below. (*segment_image/0* is not given here, because it is problem specific and would distract us from the main point.)

```
outer_edge :-
        thr(1),                 % Select outer edge
        circular.               % Test for circularity. Defined below

cherry :-
        thr(1),                 % Select outer edge
        cgr(X1,Y1),             % Centroid of outer edge
        swi,                    % Switch images
        thr(200),               % Select cherry
        swi,                    % Switch images
        cgr(X2,Y2),             % Centroid of the cherry
        distance([X1,Y1,],[X2,Y2],D),
                                % D is distance [X1,Y1] to [X2,Y2],
        D < 20.                 % Are cherry & outer edge concentric?

icing :-
        thr(128),               % Select icing
        rei(mask),              % Read annular mask image from disc
        exr,                    % Differences between these images
        cwp(A),                 % Calculate area of white region
        A > 50.                 % Allow few small defects in icing

circular :-
        cwp(A),                     % Calculate area
        perimeter(P),               % Calculate perimeter.
        S is A/(P* P),              % Shape factor
        S < 0.08.                   % S >= 1/(4* π) (min. for circle)
```

Notice the highly modular approach to Prolog+ programming and the fact that it is possible to define what is an “acceptable” Bakewell tart, in a simple and natural way. Apart from the predicate *segment_image/0*, whose definition depends upon the lighting and camera, the program *bakewell_tart/0* is complete and provides a simple yet effective means of inspecting Bakewell tarts.

23.1.4.2 Recognising Printed Letters

This application is included to illustrate Prolog+, rather than explain how a practical optical character recognition system might work. The top two layers of a Prolog+ program for recognising printed letters are given below:

```
% Top level predicate for recognising printed
% letters, which may be either upper or lower case
% and in any one of three fonts.
letter(X) :- upper_case(X).                 % May be upper case ...
letter(X) :- lower_case(X).                 % ... or lower case

upper_case(X) :-
        font(Y),                            % Find what font we are using
        member(Y,[times,courier,helvetica]),
                                            % Is font Y known to us?
        recognise_upper_case(X,Y).          % X is upper case in font Y

lower_case(X) :-
        font(Y),                            % Find what font we are using
        member(Y,[times,courier,helvetica]),
                                            % Is font Y known to us? to us
        recognise_lower_case(X,Y).          % X is lower case in font Y
```

The complex task of recognising an upper- or lower-case letter in any of the three known fonts has been reduced to a total of 156 (=3*2*26) much simpler sub-problems. (A simple declarative definition of the *sans serif* upper-case letter A is presented later.) Now, let us consider what changes have to be made if a new font (e.g., *Palatino*) is to be introduced. Two changes have to be made:

(i) the second line in the body of *upper_case/1* and *lower_case/1* is changed to
member(Y,[times,courier,helvetica,palatino])
(ii) two new clauses are added for each letter X, one for
recognise_upper_case(X,palatino)
and another for
recognise_lower_case(X,palatino).

If we wanted to add recognition rules for the numeric characters, then 10 new clauses would be added, as in (ii). In other words, extending the scope of a Prolog+ program is conceptually simple, if rather tedious to accomplish. Here, as promised, is a naive but quite effective declarative definition of the *sans serif* upper-case letter A:

```
recognise_upper_case(a,sans_serif) :-
        apex(A),                    % There is an apex called A.
        tee(B),                     % There is a T-joint called B
        tee(C),                     % There is a T-joint called C
        line_end(D),                % There is a line_end called D
```

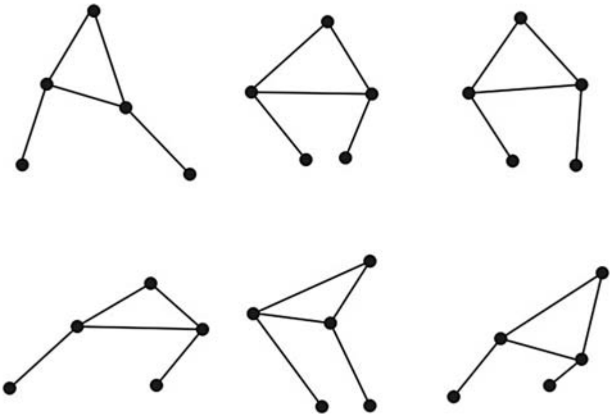


Fig. 23.5
Objects that are inappropriately recognised as the sans serif upper-case letter A by the goal *recognise_upper_case(a,san_serif)*

```
line_end(E),                    % There is a line_end called E
above(A,B),                     % A is above B.
above(A,C),
about_same_vertical(B,C),
about_same_vertical(D,E),
above(B,D),
above(C,E),
connected(A,B),
connected(A,C),
connected(B,D),
connected(C,E),
connected(B,C),
left(B,C),
left(D,E).
```

The reader should be able to understand the above program without detailed knowledge about how the predicates *apex/1, tee/1, line_end/1, above/1, about_same_vertical/2, above/2, connected/2, left/2* and *right/2* are defined. *Figure 23.5* shows some of the objects which are recognised by this program. Obviously, *recognise_upper_case/2* can be refined by adding further conditions, to eliminate some of the more bizarre objects that are recognised by the present definition and which are shown in *Fig. 23.5*.

23.1.4.3 Identifying Table Cutlery

The following Prolog+ program identifies items of table cutlery that are viewed in silhouette. (It is assumed, for the sake of brevity, that the input image can be segmented using simple thresholding.) The top-level predicate is *camera_sees(Z)* and is a general purpose utility that can find any object, given an appropriate definition for the subsidiary predicate *object_is/1*. In its present somewhat limited form, the program recognises only forks and knives; additional clauses for *object_is/1* are needed to identify other utensils, such as spoons, plates, mats, etc. When the query *camera_sees(Z)* is specified, Z is instantiated to the type of object seen by the camera. In practice, there may be many objects visible to the camera and the program will progressively analyse each one in turn. Any objects visible to the camera that are not recognised are indicated by instantiating Z to the value *unknown_type*.

```
% Top level predicate for recognising individual items of table cutlery
camera_sees(Z) :-
        grb,                    % Digitise the camera image
        segment_image,          % Example: [enc, thr(128)]
        ndo,                    % Shade resulting binary image; each
                                % blob has different intensity
        wri(temp),              % Save image in disc file
        repeat,                 % Begin loop - analyse blobs in turn
        next_blob,              % Select 1 blob from image in "temp"
        object_is(Z),           % Blob is object of type Z
        finished.               % Succeeds when no more blobs
% Select one blob from the image stored in disc file “temp”.
% Remove this blob from the stored image, so that it will not be considered next time.
next_blob :-
        rei(temp),              % Read image from disc file “temp”
        gli(_,A),               % Identify next blob - i.e., brightest
        hil(A,A,0),             % Remove it from stored image
        wri(temp),              % Save remaining blobs
        swi,                    % Revert to previous stored image
        thr(A,A).               % Select one blob
% Recognises individual non-overlapping objects in a binary image. See➔ Fig. 23.6
object_is(fork) :-
        mma(X,Y),               % Lengths of major & minor axes. mma is not defined here
        X >= 150,               % Length must be >= 150 pixels
        X =< 450,               % Length must be =< 450 pixels
        Y >= 25,                % Width must be >= 25 pixels
        X =< 100,               % Width must be =< 100 pixels
        Z is X/Y,               % Calculate aspect ratio - whatever
                                % orientation
        Z =< 10,                % Aspect ratio must =< 10
        Z >= 4,                 % Aspect ratio must be >= 4
        count_limb_ends(N),
                                % N is number of limb ends
        N >= 3,                 % Skeleton of fork has >= 3 limb ends
        N =< 5.                 % Skeleton of fork has =< 5 limb ends
% Add extra clauses to recognise each possible type of object. See➔ Fig. 23.4
object_is(knife) :-
        mma(X,Y),               % Lengths of major & minor axes
        X >= 150,               % Length must be >= 150 pixels
        X =< 450,               % Length must be =< 450 pixels
```

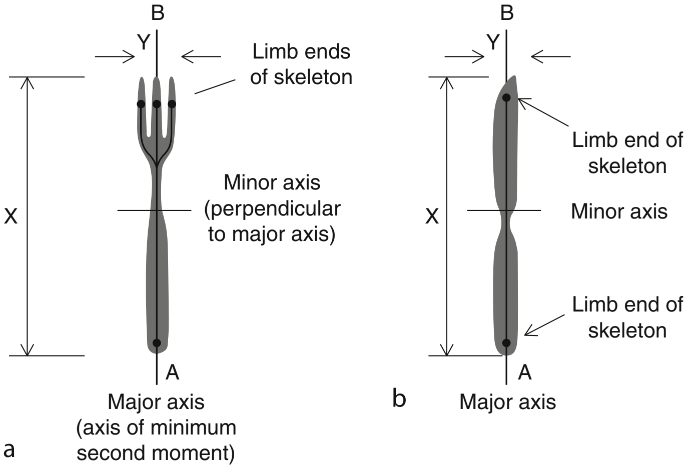


Fig. 23.6

Explaining the operation of two object recognition programs (a) *object_is(fork)*. AB is the axis of minimum second moment. Criteria to be satisfied, before this object can be accepted as a fork: 150 =< X =< 450; 25 =< Y =< 100; 4 =< X/Y =< 10; skeleton here has 3–5 limb ends. (b) *object_is(knife)*. AB is the axis of minimum second moment. Criteria to be satisfied, before this object can be accepted as a fork: 150 =< X =< 450; 25 =< Y =< 100; 6 =< X/Y =< 12; skeleton must have exactly two limb ends

```
        Y >= 25,            % Width must be >= 25 pixels
        X =< 100,           % Width must be =< 100 pixels
        Z is Y/X,           % Calculate aspect ratio - whatever
                            % orientation
        Z =< 12,            % Aspect ratio must be =< 12
        Z >= 6,             % Aspect ratio must be >= 6
count_limb_ends(2).
                            % Skeleton of knife has 2 limb ends
object_is(unknown_type).
                            % Catch-all clause. Not recognised
% Search is finished. Image is black everywhere.
finished :-
        rei(temp),          % Read image from disc
        thr(1),             % Threshold stored image
        cwp(0).             % Succeeds if no. of white points = 0
% Count the limb ends on a skeleton ("match-stick") figure.
count_limb_ends(N) :-
        mdl,                % Generate skeleton of the blob
        cnw,                % Count white neighbours, 3*3 window
        min,                % Ignore back-ground points
        thr(2,2),           % Select limb ends
        eul(N).             % Instantiate N to no. of limb ends
```

23.1.4.4 Analysing All Visible Objects

The list of all objects that are visible to the camera can be found using the predicate *list_all_objects/1* defined below. This will be useful in the next section.

```
list_all_objects(A) :-
        grb,                    % Same pre-processing . . .
        segment_image,          % . . . as is used in . . .
        ndo,                    % . . . the predicate . . .
        wri(temp),              % . . . "camera_sees"
        find_object_list        % Generate list of objects seen
        ([],A).
% Terminating recursion. Succeeds when there are no more blobs left to analyse.
find_object_list(A,A) :- finished.
% Analyse all blobs in the image
find_object_list(A,B) :-
        next_blob,              % Select blob from image in file "temp"
        object_is(C),           % Blob is object of type C
        !,                      % Cut (!) makes recursion more efficient as it freezes
                                variable C
        find_object_list        % Recursion - analyse all blobs
        ([C|A],B).
```

This makes use of the fact that *object_is/1* is able to recognise an object in the presence of other objects, provided that they do not touch or overlap. We will now make good use of *list_all_objects/1* in performing a more challenging task, namely that of recognising a well-laid table place setting.

23.1.4.5 Recognising a Table Place Setting

The following Prolog+ program can recognise a table place setting with the cutlery laid out as shown in ➋ *Fig. 23.7a*. For this program to work properly, we must first define additional clauses for the predicate *object_is*, so that appropriate types of object, such as *small_knife/1, tea_spoon/1, dinner_fork/1, plate/1, mat/1*, etc., can be recognised. Notice that *table_place_setting/0* is defined in standard Prolog, without any further use of the image processing built-in predicates.

```
table_place_setting :-
        list_all_objects(A),
        equal_sets(A, [mat, plate, dinner_knife, small_knife,
        dinner_fork, small_fork, soup_spoon, desert_spoon]),
        left(dinner_fork, mat),                 % Defined below
        left(mat, dinner_knife),
        left(dinner_knife, small_knife),
        left(small_knife, soup_spoon),
        below(mat, small_fork),                 % Defined below
        below(small_fork, desert_spoon).
```

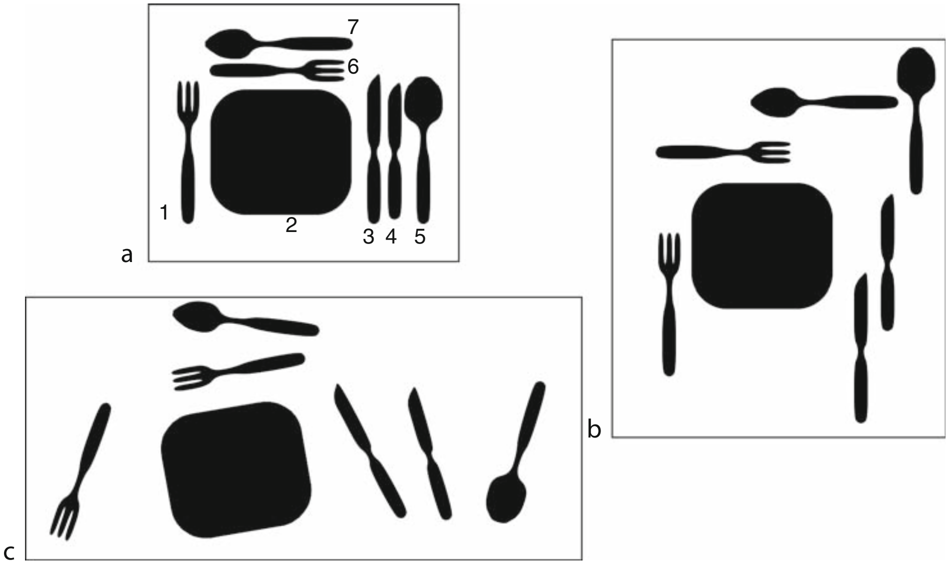


Fig. 23.7
Table place settings: (a) Ideal arrangement. Key: 1, dinner_fork; 2, mat; 3, dinner_knife; 4, small_knife; 5, soup_spoon; 6, small_fork; 7, desert_spoon. (b) Scene that is misrecognised by *table_place_setting/0* with the original definitions of *left/2* and *below/2* but which is correctly rejected by the revised version. (c) A scene that is incorrectly recognised by the revised version of the program

For completeness, we now define the *predicates left/2, below/2* and *equal_sets/2.* These and many other useful "general purpose" predicates like them form part of a PQT Library, which augments the basic language.

```
left(A,B) :-
        location(A,Xa,_),          % Horizontal position of A
        location(B,Xb,_),          % Horizontal position of B
        !,                         % Inhibit backtracking
        Xa < Xb.                   % Compare horizontal positions
below(A,B) :-
        location(A,_,Ya),          % Vertical position of A is Ya
        location(B,_,Yb),          % Vertical position of B is Yb
        !,                         % Inhibit backtracking
        Ya < Yb.                   % Compare vertical positions
equal_sets([],[]).                 % Terminate recursion
% Checking two non-empty lists are equal
equal_sets([A|B],C) :-
        member(A,C),               % A is a member of C
        cull(A,C,D),               % Delete A from C. Result D
        !,                         % Improve efficiency
        equal_sets(B,D).           % Are sets B & D equal?
cull(_,[],[]).                     % Empty list, Do nothing
```

```
% Delete A from list if A is at its head
cull(A,[A|B],C) :-
        !,                              % Don't back-track - improves efficiency
        cull(A,B,C).                    % Repeat until A & B are empty
% A is not head of "input" list [B|C] so work on its tail
cull(A,[B|C],[B|D]) :-
        !,                              % Improves efficiency
        cull(A,C,D).                    % Repeat until A & B are empty
```

Using these simple definitions, a range of unusual configurations of cutlery and china objects is accepted (see Fig. 23.7b). To improve matters, we should refine our definitions of *left/2* and *below/2*. When redefining *left(A,B)*, we simply add extra conditions, for example that *A* and *B* must be at about the same vertical position:

```
left(A,B) :-
        location(A,Xa,Ya),
        location(B,Xb,Yb),
        !,
        Xa < Xb,                            % Compare horizontal positions
        about_same(Ya,Yb, 25).              % Tolerance level = 25
about_same(A,B,C) :- A =< B + C.
about_same(A,B,C) :- A >= B − C.
```

The predicate *below/2* is redefined in a similar way:

```
below(A,B) :-
        location(A,Xa,Ya),
        location(B,Xb,Yb),
        !,
        Ya < Yb.                              % Compare vertical positions
        about_same(Xa,Xb, 25).                % Tolerance level = 25
```

In a practical application, it would probably be better for the tolerance parameter required by *about_same/3* (third argument) to be related to the size of the objects to be recognised. This would make a program such as *table_place_setting/0* more robust, by making it size-independent. This modification to our program improves matters, but it still recognises certain cutlery arrangements as being valid place settings, even though we would probably want to exclude them in practice (Fig. 23.7c). Clearly, we can go on adding further conditions to our program, in order to reduce the number of cutlery arrangements accepted by it.

In Part 2, we discuss the task of recognising a well-laid place setting, using a natural language (English) description of the ideal layout.

23.1.5 Justifying the Design

We have repeatedly emphasised that Prolog is able to handle abstract concepts, expressed in symbolic form. In this section, we shall try to justify this claim and explain why our latest implementation of Prolog+ was based on QT, rather than the unadorned MATLAB Image

Processing Tool-box. In doing so, we shall explain how to recognise items of fruit in PQT and then pack a lunch box that provides a well-balanced diet. This is achieved using the same language that we might use for other very different tasks such as inspecting piece parts, identifying flaws in a woven fabric, operating a robot or controlling a lighting array.

23.1.5.1 Additional Spatial Relationships (➲ *Fig. 23.8*)

The predicate *table_place_setting/0* relies upon *abstract* concepts about spatial relationships between pairs of objects in an image. The relationships in question (i.e., *left/2* and *below/2*) were defined earlier. Another predicate based on the spatial relationship between two objects is *encloses/2* and is clearly related to its reciprocal relationship, *inside/2*:

```
% A encloses B if B is inside A
encloses(A,B) :- inside(B,A).
inside(A,B) :-
        wri(image),            % Save the original image
        isolate(A),            % Isolate object A
        wri(A),                % Save image until later
        rei(image),            % Recover the original image
        isolate(B),            % Isolate object B
        rei(A),                % Recover saved image during "wri"
        sub,                   % Subtract images
        thr(0,0),              % Find all black pixels
        cwp(0).                % There are zero black pixels
```

Notice that *inside(A,B)* creates two binary images, containing objects *A* and *B*. These images are then compared. If all white pixels in one image (i.e., the one containing object *B*) are also white in the other (containing object A), then we can conclude that *A* is inside *B* and the goal *inside(A,B)* succeeds.

There are, of course, many other abstract relationships of this kind. Here are the definitions of a few of them:

```
% Are objects A and B concentric?
concentric(A,B) :-
        location(A,Xa,Ya),            % Definition may be based on centroid
        location(B,Xb,Yb),
        near([Xa,Ya],[Xb,Yb],10).     % Distance [Xa,Ya] to [Xb,Yb] =< 10?

% Are objects A and B in about the same vertical position?
about_same_vertical(A,B) :-
        location(A,Xa,Ya),            % Definition may be based on centroid
        location(B,Xb,Yb),
        about_same(Ya,Yb,10).         % Is |Ya−Yb|=< 10?

% Test whether object A is in the upper part of
% object B (i.e., the bottom-most point in A is
```

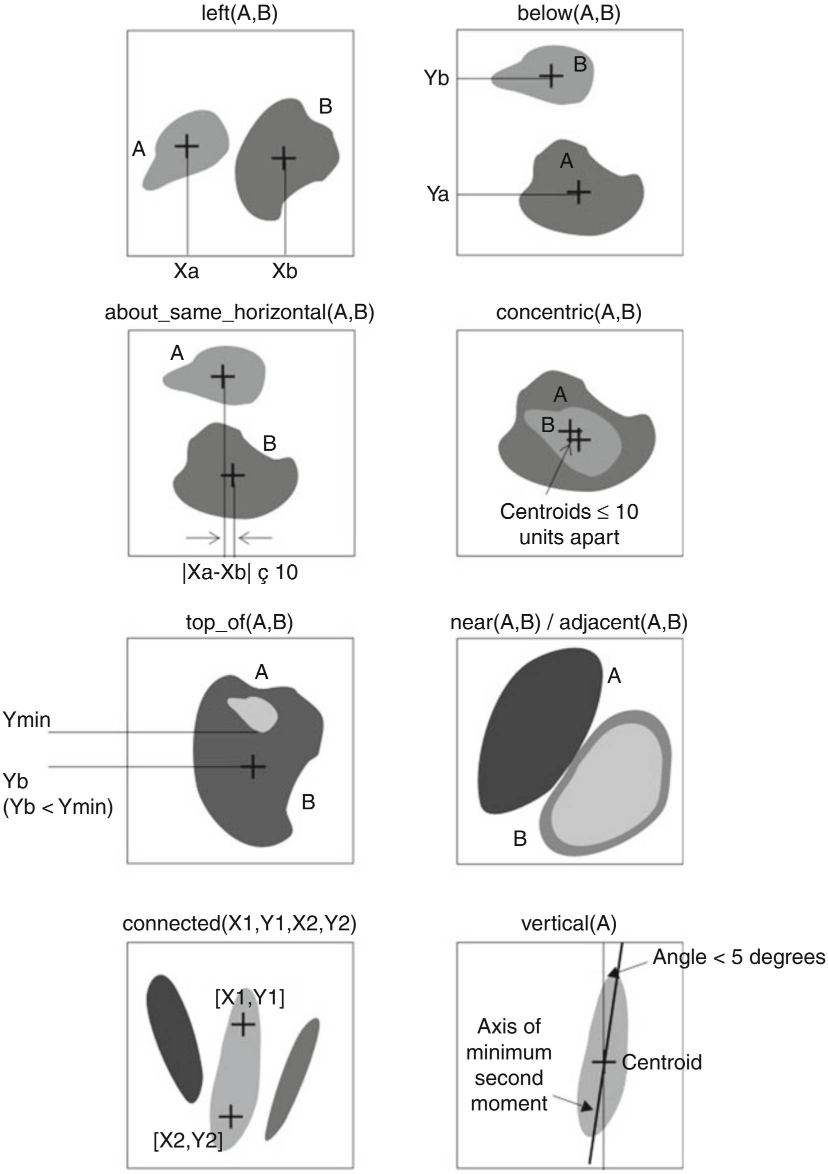


Fig. 23.8
Spatial relationships in Prolog+

```
% above B's centroid). A must be entirely contained inside B.
top_of(A,B) :-
        isolate(A),                 % Isolate object A
        dim(_,Ya,_,_),              % Bottom of object A
        isolate(B),                 % Isolate object B
        cgr(B,_,Yb),                % Centroid of B
        Ya =< Yb,                       % Is bottom point in A above the centre of B?
        inside(A,B).

% Are the points [X1,Y1] and [X2,Y2] connected by a continuous set of white pixels?
connected(X1,Y1,X2,Y2) :-
        ndo,                            % Shade blobs.
        pgt(X1,Y1,Z),                   % Z is intensity at [X1,Y1]
        pgt(X2,Y2,Z),                   % Z is intensity at [X2,Y2]
        Z = 255.                        % Both pixels are white

% Are regions A and B adjacent?
adjacent(A,B) :-
        isolate(A),                     % Isolate region A
        exw,                            % Expand region by 1 pixel
        wri,                            % Save image for use later
        isolate(B),                     % Isolate region A
        rei,                            % Read image A expanded
        mni,                            % Logical AND of the 2 images
        cwp(N),                         % Count white points
        N > 0.                          % Are there some?
```

The reader may like to consider how a predicate can be defined that can test whether an object in a binary image is nearly vertical (➔ *Fig. 23.8h*). (The pivotal QT function is *lmi.*) This predicate could be used to prevent *table_place_setting/0* from detecting scenes like that shown in ➔ *Fig. 23.7c*.

23.1.5.2 High-Level Concepts

Suppose that we want to devise a Machine Vision system to identifying highly variable objects that conform to a set of rules, expressed either formally, or in a natural language, such as English, Dutch or German. We might, for example, want to recognise objects in an image that are

yellow
sausage-like in shape
curved, with a bend of 5–30°
15–50 mm wide
70–200 mm long.

There are no prizes for guessing what class of objects we are trying to recognise! We can express the recognition rule in Prolog+, in the following way:

```
object(banana) :-
        isolate(yellow),        % Keep “yellow” objects
        big,                    % Ignore small objects
        sausage,                % Shape recognition rule based on “length” and “width”
        curved,                 % Another test for shape
```

```
width(W),              % Measure the width
W >= 15, W =< 50,      % Arithmetic tests
length(L),             % Measure the length
L >= 70, L =< 200.     % Arithmetic tests
```

Look around the room where you are reading this. Is there anything in it that could possibly match this specification? It is most unlikely that there is, unless of course, you happen to have a banana nearby! However, a few other things do look like bananas (e.g., plastic replica bananas, sweet-corn (maize) cobs, rolls of yellow paper, etc.) In addition, some perfectly good bananas may be too large, too small, or be so deformed that this rule fails. Indeed rule-based recognition criteria will almost always make some mistakes. However, if we have been diligent in seeking the most appropriate discriminating criteria, the error probability should be quite low. This approach frequently results in heuristic, rather than algorithmic, rule-based recognition procedures (see ➜ Chap. 17). There are no formal, universally acceptable rules for recognising a "sausage", or a "curved" object. Human beings do not always agree about what is/is not "yellow". (The author and his wife often disagree about what is blue/green! See ➜ Chaps. 4 and ➜ 16) Despite this, our test for "banana-ness" is quite precise; very few objects exist that pass such a test. The reason is that we have juxtaposed several highly specific factors, relating to colour, shape and size. We might also add descriptions of texture, by including a test for "spottiness", based upon the density of small dark features within a yellow blob. The reader may like to contemplate how such an idea can be expressed in PQT.

The recognition rule for "banana-ness" relies on elementary concepts that can each be specified in QT. Clearly, "raw" MATLAB, NeatVision (➜ Chap. 22) and similar systems might be used instead. Each of these is, on its own, able to perform all of the necessary image processing, arithmetic and logical operations needed to implement the object(banana) recognition rule. We shall explain later why Prolog+ has been implemented using QT. For the moment we shall concentrate on justifying the use of Prolog. To do this, consider the following extensions of our program.

First of all, bananas can be green, yellow and green, yellow, or yellow with brown spots. Our program can easily be modified to recognise under-ripe, ready to eat and over-ripe bananas:

```
fruit(banana) :- object(under_ripe_banana).
fruit(banana) :- object(ripe_banana).
fruit(banana) :- object(over_ripe_banana).
```

Each of these three clauses then requires the description of a banana in a different state of ripeness. While this adds to the total quantity of program code that we must create, each of the new subsidiary predicates (i.e., *object(under_ripe_banana), object(ripe_banana)* and *object (over_ripe_banana))* is easy to write. We have divided the problem into simpler rules that each deals with a separate state of affairs.

Moreover, we can recognise the more general predicate *fruit*, by combining several rules like the one already listed:

```
fruit(apple) :- object(apple).
fruit(banana) :- object(banana).
fruit(grapefruit) :- object(grapefruit).
fruit(kiwi) :- object(kiwi).
fruit(orange) :- object(orange).
fruit(pear) :- object(pear).
```

Of course, each class, such as *apple,* can be sub-divided, which may will probably make the program easier to write:

```
object(apple) :- object(cox_orange_pippin_apple).
object(apple) :- object(golden_delicious_apple).
object(apple) :- object(macintosh_apple).
object(apple) :- object(pink_lady_apple).
```

In addition, we can plan a lunch box thus:

```
lunch_box(A,B,C,D,E) :-
        savoury(A),          % Savoury item (e.g., ham sandwich))
        desert(B),           % Choose a desert (e.g., cake)
        fruit(C),            % Choose one item of fruit
        fruit(D),            % Choose another item of fruit
        not(C = D),          % Items of fruit must be different
        drink(E).            % Choose a drink
```

It is a straightforward matter to add extra criteria for checking dietary balance; we simply add a database of important nutritional factors for each food type and add simple tests to make sure that the lunch-box contains a healthy meal.

We are able to plan the contents of the lunch box in the very same language that we use to perform a wide variety of image processing operators. We have moved quickly and effortlessly from image-based recognition criteria of specific types of fruit (e.g., ripe bananas), through the more general concept of *banana*, then to the bigger class *fruit*. After a short step further, we are able to discuss the planning of a healthy meal. We can do all this in PQT, which is also well suited to tasks such as controlling a robot and multi-function lighting rig.

PQT is an extension to Prolog, giving it the ability to sense the world visually. This does not limit Prolog's powers in any way. So, all of the arguments that support the use of Prolog in applications requiring abstract reasoning are all still valid.

Notice that the Prolog inference engine is not used to handle images directly; QT does that. This is an important point as Prolog does not handle large volumes of numeric data comfortably.

23.1.5.3 Why Base Prolog+ on QT?

Thus far, we have explained why we have used Prolog but have not yet justified the reliance on QT. There are several points in answer to this question.

1. QT is built on the solid ground of experience and is the latest in a series of interactive image processing systems whose development began over 30 years ago [9]. During this time, the command repertoire has been developed specifically for analysing industrial inspection and visual control applications. Some of the commands available in MATLAB, for example, do not have any obvious use in industrial Machine Vision systems.

2. QT's functional command structure fits in well with Prolog's syntax. Although the two are not directly compatible it is a straightforward matter to translate from one form into the other. On the other hand, MATLAB's basic command structure is not so easily matched to Prolog.
3. QT has a standardised image format and a simple operating paradigm, based on the current-alternate image model. While it may seem to be restrictive, QT's heavy reliance on monochrome images reflects current usage in many industrial Machine Vision systems and has never been observed to cause even minor difficulties. Recall that QT can also perform a wide variety of operations on colour images, despite this constraint. (If the 8-bit model ever proved to be unduly restrictive, it is reassuring to note that a 16-bit version of QT could be produced with reasonable ease.) On the other hand, "raw" MATLAB allows the use of certain image formats, most noticeably sparse and indexed arrays, that are unlikely to be used in dedicated, high-speed systems. In short, MATLAB's great versatility makes it more difficult to use, compared to QT.
4. PQT must be capable of being operated interactively, as well as in programmed mode. It is important therefore that the same commands be used in both modes of operation.

QT is simple to operate because it does not require the user to think about the identity of the image being processed nor where the result will go. Consider a simple image processing function, such as negation (*neg*). In QT, the input image is implicitly taken to be the *current* image (see Chap. 21). The result is placed in the *current* image and the original (i.e., the contents of the current image just before *neg* was performed) is stored in the *alternate* image. Almost all QT functions behave in this way. Dyadic operators (e.g., *adi*, *sub*, *mxi*) require two inputs. Again, these are the current and alternate images. In neither case, is there any need to specify the source and destination images. Here is the equivalent operation to neg expressed in "raw" MATLAB format:

```
global current
global alternate
alternate = current;
current = imcomplement(current);
subplot(1,2,1)
imshow(current)
subplot(1,2,2)
imshow(alternate)
```

(For the sake of fairness, we should shorten the image names to minimal size.) It requires a lot of typing to perform even a simple task, such as negate (*neg* in QT) and is certainly not conducive to maintaining effective user interaction. In addition, the user has to keep a mental note of all the images that have been created. QT allows additional images to be created but normally discards those that will not be needed again.

23.1.5.4 Using Prolog+

Using PQT for interactive image processing is almost as easy as using QT. In response to each system prompt, the user types a simple query. The processed image is displayed almost immediately. The syntax is, of course, slightly different for PQT and QT; the latter sometimes requires a few more characters to be typed. As a result, the computational speed and effectiveness of the user interaction are hardly diminished. It is just as easy to write a new PQT predicate, as it is to define a MATLAB script to execute a sequence of QT functions.

In normal practice, when using PQT, it is best to group together any long sequences of image processing commands before beginning to program in conventional "Prolog mode". There is no point in carrying a burden of perhaps 20, or more, separate image processing operations when writing programs that involve backtracking and recursion. As far as possible, "wrap up" standard image processing sequences first. Thereafter, try, as far as possible, to write Prolog code from the top down, bearing in mind what image features can be detected, located and measured, easily. In many cases, morphology can be used to good effect to do this (see ➲ Chap. 19). It is a straightforward matter then to express any relationships that exist between them, image features by declarative programming.

We have already seen that the spatial relationships between the apex, T-joints and limb-ends of upper-case letter A are simple to define in terms of *left_of/2*, *above/2*, etc. Higher level spatial relationships have been defined (➲ *Fig. 23.8*). In addition, region properties can be expressed in PQT predicates, including *spotty/1*, (*spotty(X)* tests whether region/feature *X* is spotty*)*, *smooth/1*, *striped/1*, *darker/2 (darker(X,Y)* tests whether region *X* is darker than region *Y)*, *brighter/2*, etc. Do not be afraid to use very simple definitions of relationships and qualities like these. In general terms, do not be tempted to be very precise when programming in PQT. Such a blatant lack of precision presents some people with a fundamental difficulty. However, the maxim when programming in PQT is to accet that it is concerned with "sloppy programming". (The word *fuzzy* would be better but this invokes ideas of Fuzzy Sets, which is not intended.) This approach is justified in ➲ Chaps. 2 and ➲ 17.

23.2 Part 2: Using Prolog to Understand Natural Language

23.2.1 Prolog and Natural Language

It has long been the ambition of computer scientists to use natural language for programming. While the structure of a language, such as English, Welsh or German, is very complex, it is possible to represent a *small subset* using Prolog. For example, the sentences that might be needed to describe the movements of pieces on a chess board use both a restricted vocabulary and have a well-defined structure that can be represented using Prolog. Of course, a human being can always find ways to confuse a machine, by perversely using complex phraselogy and abstruse references. For the purposes of the present discussion, it is expected that human beings will behave reasonably when holding conversations with a machine.

23.2.1.1 Definite Clause Grammars

One popular approach to linguistic analysis relies upon *Definite Clause Grammars* (DCGs), which form an integral part of Prolog. DCGs closely resemble the familiar Backus-Naur Form (BNF), which is frequently used to define computer language syntax. A definite clause grammar consists of a set of *production rules* of the form

```
head –> body.                    % To recognise “head” recognise “body”.
body –> part1, ..., partn.       % Concatenation of part1, ..., partn.
```

Production rules may also contain the connective operator ‘;’ (sometimes ‘|’ is used instead), which allows elements to be ORed together. For example,

```
head –> test1 ; test2 ; test3 ; . . .. . . ; testn.
```

states that in order to recognise *head*, it is sufficient to recognise *any one* (or more) of the subsidiary tests: *test1*, *test2, test3,* . . .. . ., *testn.* An alternative is to use several separate rules. Thus:

```
head –> test1.
head –> test2.
head –> test3.
etc
```

has the same meaning as:

```
head –> test1 ; test2 ; test3.
```

or

```
head –> test1.
head –> test2 ; test3.
etc
```

The operator ‘,’ (comma) is used to signify that two or more tests must *all* be satisfied. Thus:

```
head –> test1,test2,test3.
```

states that, in order to satisfy *head*, all three of the subsidiary tests (*test1, test2, test3*) must be satisfied. Syntactically, production rules closely resemble Prolog rules. However, it should be understood that they are treated in a different way.

The body of a DCG is composed of terminal symbols and conditions, separated by commas or semi-colons. Here is a very simple DCG which describes traditional British personal names

```
name –> forename, surname.                          % Forenames & surname
forename –> forename3 ; forename 4.                 % Multiple forenames
forename3 –> forename1 ; forename3, forename1.      % Male names
forename4 –> forename2 ; forename4, forename2.      % Female names
% Items in square brackets [...] are terminal symbols, which Prolog does not try to analyse
forename1 –> [philip] ; [edward] ; [andrew] ; [jonathan] ; [godfrey].
forename2 –> [elizabeth] ; [elaine], [susanna] ; [louisa] ; [victoria] ; [catherine], [grace].
surname –> [smith] ; [higgins] ; [cook] ; [martin] ; [jones].          % Terminal symbols
```

Here are a few of the possible names that satisfy these grammar rules.

```
[philip,andrew,cook]
[elizabeth, susanna,martin]
```

[jonathan,philip,higgins]
[godfrey, jonathan, godfrey, philip, godfrey, edward, godfrey, andrew, godfrey, jones]

The first three are quite acceptable. However, the last one is silly and is permitted since we have chosen to use an over-simplified set of grammar rules.

23.2.1.2 Example: Moving Chess Pieces

The grammar rules listed below define a language for controlling a robot that moves pieces around a chess board. (They do not yet explain how "meaning" is extracted from the commands to operate a real robot.)

```
move –> order, man, [to], position.
move –> order, man, [from], position, [to], position.
order –> movement ; [please], movement.
movement –> [move] ; [shift] ; [translate] ; [transfer] ; [take].
man –> article, color, piece.
article –> [] ; [my] ; [your] ; [a] ; [the].
color –> [] ; [white] ; [black].
piece –> [pawn] ; [rook] ; [castle] ; [knight] ; [bishop] ; [queen] ; [king].
position –> [column], number, [row], number ; [row], number, [column], number.
position –> number, number.
number –> [1] ; [2] ; [3] ; [4] ; [5] ; [6] ; [7] ; [8].
number –> [one] ; [two] ; [three] ; [four] ; [five] ; [six] ; [seven] ; [eight].
```

The following sentences are accepted by this grammar:

[please, move, my, white, pawn, to, column, 5, row, 6]
[shift, a, black, pawn, to, row, 4,column, 6]
[please, move, the, white, queen, from, column, 3, row, 6, to, column, 5, row, 6]
[transfer, my, bishop, from, 5, 6, to, 7, 3]
[take, the, black, rook, to, 7, 3]
[please, transfer, my, knight, from, row, 3, column, 2, to, 6, 8]

23.2.1.3 Parsing

DCG rules are recognised by a parser, which is embodied in the Prolog predicate *phrase/3*. However, for our present purposes, the simpler form

parse(A,B) :- phrase(A,B,[]).

is preferred. The goal

parse(A,B)

tests to see whether the list B, which may contain both terminal symbols and variables, conforms to the grammar rules defined for A. Here is a simple example, showing how it is used in practice:

parse(move, [please, move, the, white, queen, from, column, 3, row, 6, to, column, 5, row, 6]).

Prolog will simply satisfy this goal. On the other hand, the following goal is not satisfied *parse(move, [please, do, not, move, the, white, queen, from, column, 3, row, 6, to, column, 5, row, 6]).*

An important feature of the parser is that it allows us to analyse sentences that are incompletely specified. (Notice that the variable Z is initially uninstantiated.) Consider the compound goal

parse(move, [take, the, Z, to, row, 3, column, 2]), write(Z), nl, fail.

This prints seven possible instantiations for Z:

pawn, rook, castle, knight, bishop, queen, king

This suggests a possible method for extracting "meaning" from a sentence:

(a) Use the parser to check that the sentence conforms to the given set of grammar rules.
(b) Use the unification process to search for specific symbols in certain places within the given sentence.

23.2.1.4 Another Example: Controlling a Set of Lamps

The following rules define a grammar for controlling a set of lamps. Comments indicate possible symbols that would be accepted by the parser. At this stage there is no attempt, to extract meaning or switch the lamps on/off.

```
% Simplify the problem by breaking it into simpler sub-problems.
```

Examples of each object type are given in the comments (i.e. following the % symbol)

```
light –> light1 ; light2 ; light3 ; light4 ; light5 ; light6 ; light7 ; light8.
light1 –>
        light_verb,                     % set
        type1,                          % every
        light_dev1,                     % lamp
        light_par1.                     % [to,level,5]

light2 –>
        light_verb,                     % set
        [all],                          % all
        light_dev2,                     % lamps
        light_par1.                     % half_on

light3 –>
        light_verb,                     % put
        light_dev1,                     % lamp
        numbers,                        % 5
        light_par2.                     % on

light4 –>
        light_verb,                     % [], empty string
        light_dev1,                     % lamp
        [number],                       % number, no option
        numbers,                        % 5
        light_par2.                     % half_on
```

```
light5 –>
        light_verb,                        % switch
        light_dev3,                        % front_light
        light_par2.                        % off

light6 –>
        light_verb,                        % switch
        light_dev1,                        % lamp
        numbers,                           % 3
        [to],                              % to, no option
        numbers.                           % 7

light7 –>
        light_verb,                        % put
        light_dev1,                        % lamp
        numbers,                           % 1
        [to],                              % to, no option
        lamp_intensity,   % [brightness, levl]
        numbers.                           % 8

light8 –>
        light_verb,                        % switch
        light_dev1,                        % lamp
        [number],                          % number, no option
        numbers,                           % 3
        [to],                              % to
        lamp_intensity,                    % level
        numbers.                           % 5

light_verb –> [] ; [set] ; [put] ; [switch].

type1 –> [each] ; [every].

light_dev1 –> [light] ; [lamp].
light_dev2 –> [lights] ; [lamps].
light_dev3 –> [laser] ; [projector] ; [back_light] ; [front_light].

light_par1 –> [on] ; [off] ; [half_on] ; [to], [level], numbers.
light_par2 –> [on] ; [off] ; [half_on].

lamp_intensity –> [] ; [level] ; [brightness] ; [brightness, level] ; [intensity] ; [intensity,
                  level].
```

➜ *Table 23.1* Lists a sample of the sentences that are accepted by this set of grammar rules.

Table 23.1
Some of the sentences that are accepted by the DCG rules defined in the text

Clause	Accepted string
light1	[switch, each, lamp, off]
	[switch, every, light, on]
	[set, each, lamp, off]
	[put, each, lamp, half_on]
	[every, lamp, off]
	[each, lamp, half_on]
light2	[put, all, lamps, off]
	[set, all, lights, on
	[all, lamps, half_on]
light3	[put,lamp, 6, off]
	[switch, light, 4, half_on]
light4	[put, lamp, number, 6, off]
	[light, number, 8, on]
light5	[switch,laser,off]
	[put, back_light, on]
	[projector, off]
light6	[set, lamp, 6, to, 7]
	[light, 5, to, 9]
light7	[put, lamp,6, to, level, 7]
	[light, 5, to, brightness,4]
light8	[put, lamp, number, 6, to, intensity, level, 7]
	[set, light, number, 6, to, level, 4]

23.2.1.5 Extracting Meaning

First Method: Discarding Non-relevant Terminal Symbols
Consider the goal sequence

```
X = [put, lamp, number, 6, to, intensity, level, 7],
parse(light8,X),                    % We are parsing according to "light8" rule not "light"
discard_non_numbers(X,[Y,Z]),  % Y is the lamp identifier. Z is the lamp brightness.
switch_lamp(X,Y)                   % Switch lamp X to state Y
```

This instantiates Y to 6 and Z to 7, so lamp 6 is switched to brightness level 7. Notice that the parser is being used simply to check that it is reasonable to apply *discard_non_numbers*/2, which is unintelligent and therefore could not be used safely on its own. Of course, a similar approach is adopted for *light1, light2, ..., light8.*

Second Method: Pattern Matching

Consider the predicate

```
meaning(light7,X,[A,B]) :- parse(light7,X), X =[_,_,A,_,_,B].
```

This tries to match X to a list containing six elements, of which only the third and sixth are retained. Matching is only allowed if X first satisfies DCG rule *light7*. Again, the parser is being used to check that it is reasonable to apply the pattern matching goal, (*X =[_,_,A,_,_,B]*).

However, the rule for *light_verb* permits both *empty* and *non-empty* strings, which causes some slight complications. (The list X can vary in length.) To accommodate all of the cases where rule *light7* holds, we require *four* separate conditions:-

```
meaning(P,X,Y) :-
        parse(light7,P),
        (P = [_,_,X,_,_,Y] ; P = [_,X,_,_,Y]),        % Semi-colon (;) denotes OR
        number(X).                                     % Check that X is a number

meaning(P,X,Y) :-
        parse(light7,P),
        (P = [_,_,X,_,_,_,Y] ; P = [_,X,_,_,_,Y]).     % Semi-colon (;) denotes OR
```

In this way, Prolog can "understand" the lighting control language. (➔ *Table 23.2*) The goal *meaning(A,B,C)* tries to parse the given sentence A. If the parsing succeeds, then A is analysed, so that the lamp(s) (B) and brightness level (C) can be found.

```
% Understanding sentences which are accepted by "light1"
meaning(P,all,Z) :-
        parse(light1,P),
        (P = [_,_,_,Z] ; P = [_,_,Z]).
% Understanding sentences which are accepted by "light2"
meaning(P,all,Z) :-
        parse(light2,P),
        (P = [_,_,_,Z] ; P = [_,_,Z]).
% Understanding sentences which are accepted by "light3"
meaning(P,X,Y) :-
        parse(light3,P),
        (P = [_,_,X,Y] ; P = [_,X,Y]).
```

Table 23.2
Sample queries for *meaning(A,B,C)*

A	B	C
[put,all,lamps,off]	all	off
[switch,each,lamp,half_on]	all	half_on
[set,laser,on]	laser	on
[lamp,3,to,level,2]	3	2
[switch,lamp,number,6,to,intensity,level, 5]	6	5
[switch,lamp,number,6,off]	6	off

```
% Understanding sentences which are accepted by "light4"
meaning(P,X,Y) :-
        parse(light4,P),
        (P = [_,_,_,X,Y] ; P = [_,_,X,Y]).
% Understanding sentences which are accepted by "light5"
meaning(P,X,Y) :-
        parse(light5,P),
        (P = [_,X,Y] ; P = [X,Y]).
% Understanding sentences which are accepted by "light6"
meaning(P,X,Y) :-
        parse(light6,P),
        (P = [_,_,X,_,Y] ; P = [_,X,_,Y]).
% Understanding sentences which are accepted by "light7"
meaning(P,X,Y) :-
        parse(light7,P),
        (P = [_,_,X,_,_,Y] ; P = [_,X,_,_,Y]),
        number(X).          % Resolve ambiguity with next clause
meaning(P,X,Y) :-
        parse(light7,P),
        (P = [_,_,X,_,_,_,Y] ; P = [_,X,_,_,_,Y]).
% Understanding sentences which are accepted by "light8"
meaning(P,X,Y) :-
        parse(light8,P),
        (P = [_,_,_,X,_,_,Y] ; P = [_,_,X,_,_,Y]),
        number(X).          % Resolve ambiguity with next clause
meaning(P,X,Y) :-
        parse(light8,P),
        (P = [_,_,_,X,_,_,_,Y] ; P = [_,_,X,_,_,_,Y]).
% Catch-all rule for use when the sentence is not recognised
meaning(_,unknown,unknown) :-
        message_to_user(['Sentence was not recognised']).
```

Third Method: Including Variables and Prolog Goals Within DCG Rules

Consider the following revised DCG rule:

```
light6 –>
        light_verb,                 % Unchanged
        light_dev1,                 % Unchanged
        Y,                          % Variable - lamp identifier
        {number(Y)},                % Prolog goal: check that Y is a number
        [to],                       % Unchanged
        Z,                          % Variable lamp brightness
        {number(Z)},                % Prolog goal: check that Z is a number
        {control_lights6(,[Y,Z])}.  % Prolog goal must be satisfied before parsing
                                      is accepted
```

This grammar rule includes three Prolog sub-goals, which must all be satisfied before the parser accepts the input list as valid. An example will illustrate this:

parse(light6,[switch, lamp, 3, to 7]).

While the parser is trying to prove that this is true, the following tests are carried out:

1. *switch* is verified as being a *light_verb.*
2. *lamp* is verified as being a *light_dev1.*
3. *Y* is instantiated to 3.
4. *Y* is checked to be a number (embedded Prolog goal).
5. *to* is found in the input list.
6. Z is instantiated to 7.
7. Z is checked to be a number (embedded Prolog goal).
8. *control_lights6(,[Y,Z])* switches lamp Y to brightness level Z (Prolog code).

Choosing Which Method to Use

A parser is able to check whether a vaild sentence has been presented. This can be envisaged as acting like a filter, eliminating sentences that are non-sensicle to the system. A machine can respond by indicating that it does not understand, leaving it to the user to think again about what he is saying. Shifting the responsibility for achieiving an effective dialogue is acceptable within limits; people learn quicker than machines and will tolerate some restrictions, if the overall benefits are large enough. This means that naive methods for extracting meaning, such as Method 1, are safer than we might imagine. Simply deleting words that have no obvious relevance to the discourse would be dangerous without such a filter. Individual grammar rules are usually very selective. That is, safe! DCGs can be made more general in a variety of ways. Each time we add a new rule, or generalise a rule, perhaps by including variables, the system becomes a bit less secure, simply because we allow more sentences to pass through the filter. However, most DCGs are safe because they are still highly selective compared to natural unconstrained human speech. As a result, Method 1 is well worth exploring first, because it is simple to program. Method 2 is conceptually straightforward but rather cumbersome, while the third method is concenptually the most elegant. Once mastered, Method 3 is probably the most attractive because actions, such as switching lights ON/OFF, are directly embedded in the DCG rules.

23.2.1.6 Remarks

The use of DCGs for understanding a limited sub-set of natural language has considerable potential that has, as yet, not been exploited to its full potential for vision systems. As stated earlier, it is assumed that people using such a system as the one described will act reasonably and will not deliberately look for ways to confuse the program. With this in mind, it is feasible now to build a system of moderate cost that accepts input from a speech recognition device and can perform tasks such as controlling

- Lights, within a fixed array
- Simple manipulator, including
    - (X,Y,θ)-table
    - Pick-and-place arm

  - Conveyor stop/start
  - Accept/reject mechanism
  - Filter wheel
- Pan, tilt, zoom and focus of a camera
- High-level image processing operations
- Navigation of help files, including the catalogue of lighting-viewing methods [10]
- Administration and analysis of the Machine Vision Questionnaire (MVQ)
- System initialisation, configuration and testing
- Recall of performance statistics

When designing such a speech-controlled natural language system, it is important to remember that homophones can cause confusion. For example: *rows* and *rose* sound the same and therefore must be treated as the same entity when defining a set of grammar rules. While the attractions of hands-free speech-controlled operation of industrial vision systems are obvious, there are great difficulties when trying to use DCGs to describe complex and highly variable images. Matters are made somewhat easier in the context of Automated Visual Inspection, where we know quite precisely what kind of objects and faults to expect. Nevertheless, with our present capabilities, it is impractical to employ low-level programming of vision systems (e.g., specifying individual QT commands).

23.3 Part 3: Implementation of Prolog+

23.3.1 Implementation of PQT

Using Prolog to control an image processing sub-system was conceived by David H. Mott in the mid-1980s [11]. (Also see [8, 12].) He linked Prolog and an image processing package, similar to QT, running on the same computer. Since then, several other approaches have been followed (➲ *Table 23.3*). PQT is one of the most recent implementations of Prolog+ and was constructed by interconnecting SWI-Prolog [1] and the QT image processing software described in ➲ Chap. 21. Communication between them is achieved using “glue” programs written in Java [13]. This allows Prolog and QT to run on the same or different computers, perhaps thousands of kilometres apart. While a Prolog program is able to control several QT “slave” modules, we shall concentrate on implementing Prolog+ using just one image processing engine. This does not limit the number of the cameras that can be used, nor their location. Another interface between MATLAB and SWI-Prolog is described in [14], although this was not designed specifically for image processing.

23.3.1.1 Structure of a PQT System

➲ *Figure 23.9* shows the overall structure of a system implementing PQT and which employs the following software components

(a) QT, implemented using MATLAB’s *Image Processing Tool-box* [15].
(b) *SWI-Prolog* [1]. This is available free of charge from the University of Amsterdam.
(c) Java interfacing (“glue”) software between QT and SWI-Prolog (boxes labelled J1, . . ., J4 in ➲ *Fig. 23.9*).

Table 23.3
Methods for implementing Prolog+

Prolog	Image processing	Interface	Remarks
Computer/ single-card processor	External hardware	Serial line (e.g., RS232)	Easy to build. Despite the low data rate across the Prolog-IP interface, this is surprisingly effective.
	Separate computer		
	Smart camera		Low cost. Simple to implement. IP command repertoire is fixed by the camera and may be somewhat limited compared to QT. Adequate for some of the less demanding shop-floor applications.
	External hardware (e.g., pipe-line or array processor), or plug-in card	Parallel bus	Fast enough for some shop-floor applications.
Same computer		Tight integration; direct connection between IP and Prolog programs	Can be difficult to maintain, as Prolog, the image processor and the operating system all evolve separately.
		Loose integration via operating system	Easy to build and maintain, as a standard interface is used.
Computer	Separate computer	Ethernet/Internet	Easy to build and maintain as a standard interface is used. Possible to implement multi-camera & multi-processor systems. Unpredictable delays are possible on public networks.

It is not necessary or appropriate to describe the minutiae of the Prolog-QT interface. Let it suffice to say that it involves several data-type conversions, involving MATLAB arrays and strings, Prolog strings (different delimiters), terms, atoms, lists and numbers.

The following general points should be noted:

(i) Prolog and QT can be run on the same, or different, computers.

(ii) Images from any web-enabled camera, or image file with a known URL, can be acquired, processed and analysed by the PQT system.

(iii) A range of electro-mechanical devices can be operated, via a suitable device-control hardware unit. One supervisor module that the author has used is web friendly and responds to HTTP commands. It can operate up to 16 slave boards containing power switches for such devices as stepper motors, DC motors, LEDs, solenoids, etc. (Also see Appendix J.) Since the supervisor module is connected to the network, these devices could be located anywhere in the world. Effective communication can be achieved using standard wired, or wireless, Internet paths and protocols.

(iv) Once the initialisation process has been completed, both SWI-Prolog and QT run continuously; there are no delays associated with re-starting either of them. This also means that we do not have to take any special measures to carry data from one query/command

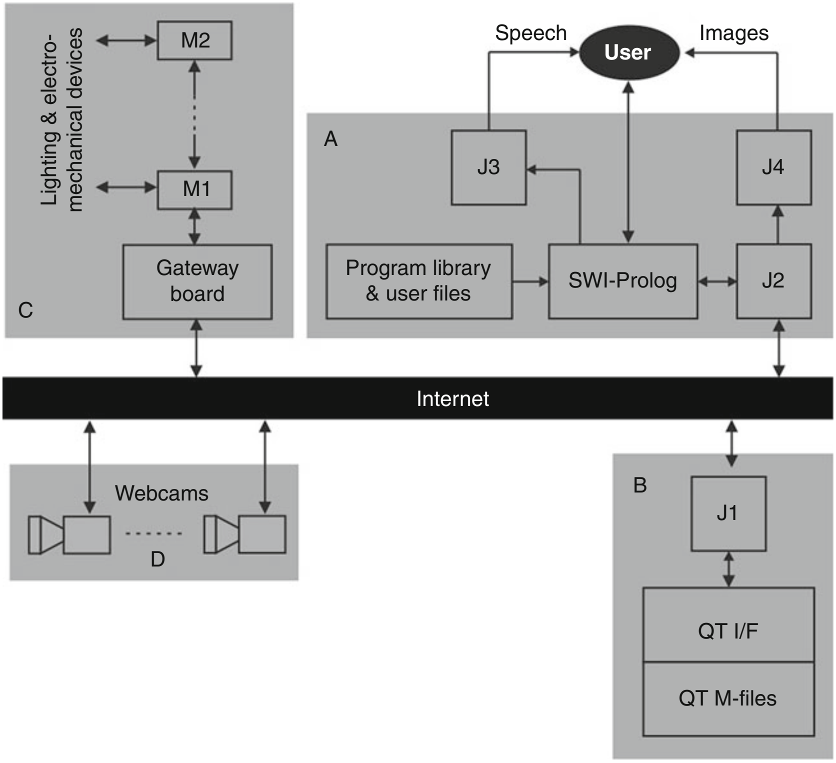


Fig. 23.9
Implementing PQT using Java "glue" software. Box C contains hardware control devices. J1 – J4 are small Java programs

over to the next. (Keeping both QT and Prolog "alive", particularly during error conditions, was one of the main factors that influenced the design of the "glue" software.)

(v) The user interface has been enhanced by using a speech synthesiser controlled from Prolog. Since Prolog can generate natural language sentences using DCGs, it is possible to construct quite sophisticated user feed-back sequences. An earlier implementation of Prolog+ also used speech recognition hardware to provide hands-free user interaction [8]. This showed quite very clearly that speech control is potentially very useful - and is great fun!

(vi) An infix Prolog operator (∼) has been programmed to perform repeated operations. This allows iterated processes, such as image filtering and morphology, to be performed very easily. (This operator is listed later.)

(vii) New functions can be written for PQT in either of two ways:
- Adding new Prolog code
- Adding MATLAB M-functions

23.3.1.2 Processing Prolog+ Goals

We shall now explain how Prolog goals incorporating image processing operations are processed

1. Prolog generates a command (C1) for QT. (The process of generating C1 from the original Prolog specified interactively by the user, or from a goal in a QT program, will be explained later.) C1 is expressed in the form of a Prolog string (S1) and sent via Java interface programs (J2 and J1 in ➲ *Fig. 23.7*) to MATLAB.
2. The command string S1 is received by a MATLAB interface M-function (*QT I/F*), which is responsible for all communication between QT and Prolog. Assuming that string S1 is correctly formatted, the appropriate command is executed by MATLAB and the results is passed back to *QT I/F.*
3. *QT I/F* standardises the data format for values that are to be returned to Prolog. First, a (MATLAB) array of size 7 × 1 elements is created, irrespective of how many values the QT M-function executed by S1 actually computes. Then, the array is converted to a string, S2. ➲ *Table 23.3* shows some examples which illustrate this. (This 7-element array allows for six values or fewer to be returned from QT, plus a variable indicating any error conditions that have occurred. Error 0 (zero) indicates no error. No current QT operator returns more than six values, although there is no fundamental reason why more values could not be calculated.)
4. String S2 is then despatched, via Java programs J1 and J2 to Prolog.
5. Program J2 reformats S2, to form a Prolog string (S3).
6. Prolog converts string S3 into a Prolog list, L1.
7. Prolog then analyses L1, to extract the values calculated by QT and ignoring the undefined ("padding") variables. Several layers of Prolog code are required to do this. For the convenience of the user, Prolog-style operators have been implemented and are explained below.

23.3.1.3 Prolog Programs Implementing Prolog+

Step 6 in the sequence described above results in a 7-element Prolog list. The head of this list indicates how many of the numbers in its tail are valid (i.e., how many outputs the QT command S1 actually generated.) As we indicated earlier, analysing this list is made easier by using Prolog-style operators.

The <- Operator represents the first step towards implementing Prolog+, although it does not provide a complete solution.

Syntax

Prefix operator	<- *command*
Infix operator	*[N,V1,V2,V3,V4,V5,V6]* <- *command*

Function Performs the QT function *command*, if it exists

Returned values N indicates how many of *V1,V2,V3,V4,V5,V6* are valid
Vi (i = 1,...,N) are the values calculated by the QT function *command*
If N is negative an error condition has been found. MATLAB does not quit.

Example 1

Goal	*[N,V1,V2,V3,V4,V5,V6]* <- *cgr*
Results	N = *2;* V1 and V2 are instantiated to the (X,Y)-coordinates of the centroid

Example 2

Goal *[N,V1,V2,V3,V4,V5,V6] <- eul(4)*

Results N = 1*;* V1is instantiated to the 4-neighbour Euler number

Example 3

Goal *<- neg*

Results [N,V1,V2,V3,V4,V5,V6] = [0,0,0,0,0,0,0]

Example 4

Goal *<- thr(123,234)*

Results [N,V1,V2,V3,V4,V5,V6] = [0,0,0,0,0,0,0]

Example 5

Goal [N|_] <- qwerty

Results N = −1 (Error: QT command *qwerty* does not exist)

The # Operator is easier to use than <- and takes us closer to Prolog+ see (*Table 23.4*). # is a prefix operator and conveniently masks some quite sophisticated processing. To understand how # works, consider the Prolog goal

# abc(1,2,3,X,Y,Z).

First, the variables are removed, leaving the Prolog term:

abc(1,2,3)

This is then converted into the following Prolog goal, which is then resolved in the manner described above:

[N,V1,V2,V3,V4,V5,V6] <- abc(1,2,3).

Table 23.4

Commands and data format used by MATLAB. The right-hand column shows the output *string* generated by the QT I/F program. This is then converted to a Prolog list that looks the same when printed. (Valid results is this list are shown in bold-face type; undefined padding variables are indicated by italics.)

Command (C1)	Operation	No. of output values calculated	String S2
neg	Negate	0	[0,0,0,0,0,0,0]
thr(123,234)	Threshold	0	[0,0,0,0,0,0,0]
avg	Average intensity	1	[1,**123**,0,0,0,0,0]
cgr	Centroid	2	[2,**111,222**,*0,0,0,0*]
mar	Min. area rectangle	4	[4,**111,222,333,444**,*0,0*]
eul(8)	Euler number	1	[1,**4**,0,0,0,0,0]
nonsense	Not a QT command	Error!	[−1,0,0,0,0,0,0]

Let us assume that the QT command *abc(1,2,3)* has been completed satisfactorily and has returned three values, instantiating the variables in this list as follows:

N = 3
V1 = 444
V2 = 555
V3 = 666

The "padding" variables V4, V5 and V6 remain uninstantiated. The final action of the # operator is to match these variables with those specified in the original goal (C1):

X = V1
Y = V2
Z = V3

Hence, the goal succeeds with the following instantiations:

```
# abc(1,2,3,444,555,666).
```

There are three other cases to consider:

1. The command *abc(1,2,3)* returns fewer than three values. In this case, some of the variables specified in the original goal remain uninstantiated. For example, if N = 1 (i.e., V2–V6 have undefined values), only X is instantiated. The goal then succeeds as
   ```
   # abc(1,2,3,123,_,_).
   ```
   Notice that the last two arguments remain uninstantiated.
2. The command *abc(1,2,3)* returns more than three values. In this case, only the results V1, V2 and V3 are retained; V4, V5 and V6 are discarded. The goal succeeds as before
   ```
   # abc(1,2,3,444,555,666).
   ```
3. If the command *abc(1,2,3)* generates a QT error, the Prolog goal C1 fails.

Completing the implementation of Prolog+ While the # operator takes us closer to implementing Prolog+, there is a still a small gap remaining. To avoid the need to precede each QT command by the # symbol, it is expedient to define a library of simple rules like those following, for each QT image processing function:

```
neg :- # neg.               % Predicate neg/0 can now be used in lieu of #neg
thr(A,B) :- # thr(A,B).     % Predicate thr/2 can now be used in lieu of thr(A,B)
thr(A) :- # thr(A,255).     % Predicate thr/1 can now be used in lieu of #thr(A)
thr :- # thr(128,255).      % Predicate thr/0. Default values can be redefined in Prolog
cgr(X,Y) :- # cgr(X,Y).     % Predicate cgr/2 can now be used in lieu of # cgr(X,Y)
cgr :- # cgr(A,Y), write(X), nl, write(Y),nl.
etc.
```

Writing rules like this for all QT commands is straightforward but rather tedious to do manually. To automate this task, the author devised a small Prolog program which builds a library of rules based on the total number of arguments (i.e., inputs plus outputs), needed for each QT function. (This "input" data was obtained from QT's *hlp*, HELP, command.) Here is a small sample of the database generated automatically by the program:

```
.....
dor(V1,V2,V3,V4) :- # dor(V1,V2,V3,V4).
```

Table 23.5
Command formats in QT, with the <- and # operators and in PQT

QT	<- Operator (L is a 7-element list)	# Operator	Library rules	PQT (X, Y and Z are unistantiated)
neg	<- neg	# neg	neg :- # neg.	neg.

thr(12,34)
thr(123)
thr
	<- thr(12,34)
	<- thr(123)
	<- thr
		# thr(12,34)
		# thr(123)
		# thr
			thr(A,B) :- # thr(A,B).
			thr(A) :- # thr(A).
			thr :- # thr(128,255).
				thr(12,34).
				thr(123).
				thr.

x =avg avg	L <- avg	# avg(X)	avg(X) :- # avg(X). avg :- # avg.	avg(X). avg.
[a,b,c] = avg Fails - too many results expected	L <- avg	# avg(A,B,C)		avg(X,Y,Z).

x = eul(8)
x = eul
	L <- eul(4)	# eul(A,X)	eul(A,X) :- # eul(A,X)
			eul(A) :- # eul(8,X)
				eul(8,X).
				eul(X).

[x,y] = cgr	L <- cgr	# cgr(X,Y)	cgr(X,Y) :- # cgr(X,Y).	cgr(X,Y).
x = cgr		# cgr(X)		cgr(X).

Table 23.6
Comparing the syntax of programs written in QT, PQT and using the # operator. A single PQT goal *[cbl(22)]* counts the blobs and checks that there are 22 of them. It performs the same function as *[cbl(N), N is 22]*. Similarly for *# cbl(22)*

QT	PQT	# Operator	Function
function [a,n] = analyse(b)	analyse(A,B,N) :-	analyse(A,B,N) :-	
enc	enc,	# enc,	Enhance contrast
thr	thr,	# thr,	Threshold at mid grey
big	big,	# big,	Isolate largest blob
a = cwp;	cwp(A),	# cwp(A),	Count white pixels
if a >250	A > 250,	A > 250,	Test value of an integer
b = round(a/2);	B is A//2,	B is A//2,	Integer division
swi	swi,	# swi,	Interchange images
kgr(b)	kgr(B),	# kgr(B),	Discard small blobs
n = cbl;	cbl(22),	# cbl(22),	Count blobs
if n == 22,	% Blank line here	% Blank line here	
display(‘Accept’)	write(‘Accept ’),	write(‘Accept ’),	Display message
else	nl,	nl,	
display(‘Reject’)	!.	!.	Prolog: cut (!)
end			
end	analyse_image(N) :-	analyse_image(N) :-	Prolog: Display message
	write(‘Reject’), nl.	write(‘Reject’), nl. nl.	

```
dor(V1,V2,V3) :- # dor(V1,V2,V3).
dor(V1,V2) :- # dor(V1,V2).
dor(V1) :- # dor(V1).
dor :- # dor.
dri(V1,V2) :- # dri(V1,V2).
dri(V1) :- # dri(V1).
dri :- # dri.
drp :- # drp.
.....
```

The set of rules generated in this manner now forms part of the PQT library and is loaded automatically when SWI-Prolog is started.

➔ *Tables 23.5* compares Prolog+ notation with that of the <- and # operators, while ➔ *Table 23.6* compares a QT program, its Prolog+ equivalent and another using the # operator. However, remember that most Prolog+ programs cannot be translated directly into QT. Hence, comparisons like this should not be taken too far.

References

1. What is SWI-Prolog? University of Amsterdam, Netherlands. URL http://www.swi-prolog.org/. Accessed 31 Oct 2007
2. Gazdar G, Mellish C (1989) Natural language processing in prolog. Addison-Wesley, Wokingham
3. Coelho H, Cotta J (1988) Prolog by example. Springer, Berlin. ISBN 0 471 92141 6
4. Sterling L, Shapiro E (1994) The art of prolog, second edition: advanced programming techniques (logic programming). MIT Press, Cambridge. ISBN ISBN0-262-19338-8
5. Spenser C (ed) (1995) Prolog for industry Proceedings of the LPA Prolog Day RSA, Logic Programming Associates, London, ISBN 1 899754 00 8
6. Bratko I (2000) Prolog programming for artificial intelligence. Addison-Wesley, Wokingham
7. Clocksin WF, Mellish CS (2003) Programing in Prolog: using the ISO standard. Springer, Berlin
8. Batchelor BG (1991) Intelligent Image Processing in Prolog. Springer, Berlin. ISBN 978-3-540-19647-1. URL http://www.springer.com/978-3-540-19647-1
9. Batchelor BG (Apr 1979) Interactive image analysis as a prototyping tool for industrial inspection. Proc IEE Comput Digit Tech 2(2):61–69, Part E
10. Lighting-viewing methods, this book, Chapters 8 and 40
11. Mott DH (Oct 1985) Prolog-based image processing using Viking XA. In: Zimmerman N (ed) Proceedings of the international conference on robot vision & sensory controls. IFS, Amsterdam, pp 335–350. ISBN 0-903608-96-0
12. Bell B, Pau LF (1992) Context knowledge and search control issues in object-oriented prolog-based image understanding. Pattern Recognit Lett 13(4):279–290
13. Caton SJ (2010) Networked vision systems. Ph.D. thesis, School of Computer Science, Cardiff University, Cardiff, Wales, UK
14. Abdallah S, Rhodes C. Prolog-Matlab interface. Queen Mary College University of London. http://www.elec.qmul.ac.uk/digitalmusic/downloads/index.html#plml. Accessed 17 Feb 2011
15. MATLAB (2007) Image processing toolbox. Mathworks, Natick. URL http://www.mathworks.com/. Accessed 31 Oct 2007
16. Bell B, Pau LF (Sept 1990) Contour tracking and corner detection in a logic programming environment. PAMI 12(9):913–917

24 Pattern Recognition

Malayappan Shridhar · Paul Watta
The University of Michigan-Dearborn, Dearborn, MI, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_24,

Abstract: Human beings are hard-wired to discover and categorize patterns as a way of understanding the world. Humans can easily read poorly handwritten text, parse and understand human speech, recognize faces in a crowd, classify people as old or young, etc. However, it is difficult for a machine to solve these kinds of perceptual problems. In this chapter, we explore some of the concepts and techniques of machine-based pattern recognition, including statistical-based approaches and neural network-based approaches. We illustrate the concepts with case studies taken from image and video processing.

24.1 Introduction

24.1.1 What Is Pattern Recognition?

Human beings are hard-wired to discover and categorize patterns as a way of understanding the world. We find patterns everywhere: in the structures of nature [2], language and music [7, 8], human behavior [9] and human history [3]. Human beings are prone to seeing patterns even when none exist! [12].

In terms of processing perceptual data, human beings are amazingly adept. They can easily read poorly handwritten text, parse and understand human speech, recognize faces in a crowd, classify people as old or young, etc. However, it is difficult for a machine to solve these kinds of perceptual problems. One difficulty stems from having to process the enormous amount of data inherent in these perceptual problems. With the advent of modern computers, significant advances have been made for machine understanding and processing of patterns.

Automated pattern recognition has been successfully applied in a variety of fields, including engineering, science, medicine, and business. Applications include character and string recognition, speech understanding, image understanding, mobile robots, and security systems. Successful commercial applications include address recognition in mail, isolated word recognition, recognition of bank checks, automated processing of tax forms, finger print matching, detection of abnormalities in manufactured parts, face recognition, signature verification, speaker verification, and detection of forgery in financial documents.

Pattern recognition involves mapping input perceptual data into some other (often simpler) form so that meaningful action can be taken. A schematic diagram of a pattern recognition system is shown in ➲ *Fig. 24.1*. There are four stages: preprocessing, feature extraction, recognition, and decision. In the preprocessing stage, the raw data is put in a form suitable for further processing. For example, if the raw data consists of MPEG video, a preprocessing step might be to uncompress the video. In an image processing application, typical preprocessing involves cropping the area of interest, size normalization, and performing color correction, such as histogram equalization.

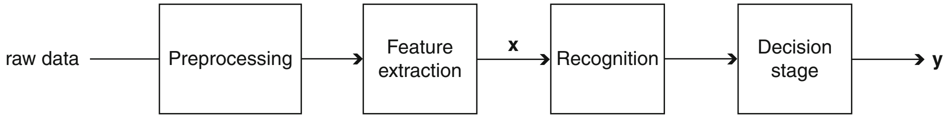


Fig. 24.1
Main computational stages of a pattern recognition system

The second stage involves extracting useful features from the input perceptual data. For our purposes, a feature vector **x** is simply a set of numbers which accurately represents the input perceptual data. A main goal here is to come up with an efficient representation; that is, reduce the dimensionality of the input as much as possible, but not to the point where it adversely affects recognition and classification performance. As an example, speech is typically processed at about 10 kHz. Thus a 1 min segment of speech will have 600,000 samples. However, through an in-depth understanding of the process of speech generation, it is possible to extract ten features for each 250 ms segment of sampled speech, giving a total of only 2,400 features. Similarly, a word such as “*hundred*” written on a piece of paper has a size of about 2″ × 0.25″. When scanned at 200 dpi and 8 bits/pixel, the resulting image is 400 × 50 pixels. These 20,000 pixels can be reduced to features that are essentially eight 64-dimensional vectors.

An illustration of feature extraction for a character recognition problem is shown in *Fig. 24.2*. Here, the feature of interest is the contour of the character. A quantitative measure of the contour can be obtained by examining the behavior of local pixel changes in four directions of interest: 0°, 45°, 90°, 135°, as shown in *Fig. 24.2a*. Starting with the binary image shown in *Fig. 24.2b*, we first extract the edges, as shown in *Fig. 24.2c*. A feature vector can be constructed consisting of the histograms of the chain codes of the contour elements. The rectangular frame enclosing character contours is divided into 4 × 4 rectangular regions. In each region, a local histogram of the chain codes is calculated. The feature vector is composed of these local histograms. Thus the feature vector has 64 components when all the 16 regions are included. *Figure 24.2* illustrates the procedure. Thus if we consider written digit recognition, we will have ten such patterns for each digit from 0 to 9.

The third stage in the pattern recognition system is the actual recognition or classification algorithm employed. Here, the task is to partition the feature space (that is, the set of all possible feature vectors) into a set of disjoint regions, as shown in *Fig. 24.3*. It is hoped that feature vectors from the same class are as close together as possible, while feature vectors from different classes are as distinct as possible. *Figure 24.3a* shows an example where a simple linear separation is possible between the two classes. In *Fig. 24.3b*, perfect separation is not possible; however, it is desired that the separating surface be positioned to minimize the resulting misclassification. The patterns in *Fig. 24.3c* require something more sophisticated than a linear separating surface; for example, a quadratic surface. In a multi-class problem shown in *Fig. 24.3d*, sometimes it is possible to achieve a piecewise linear separation between classes.

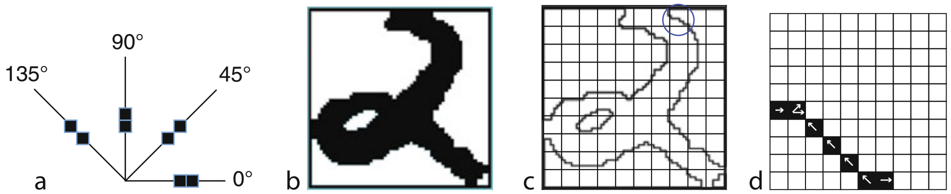


Fig. 24.2
Illustration of feature vector extraction for a hand-written charcter. (a) Pixel alignment for the four directions of interest: 0°, 45°, 90°, 135°. (b) Input image. (c) Extracted edges. (d) Focusing on the pixel circled in (c), we compute the direction of each pixel

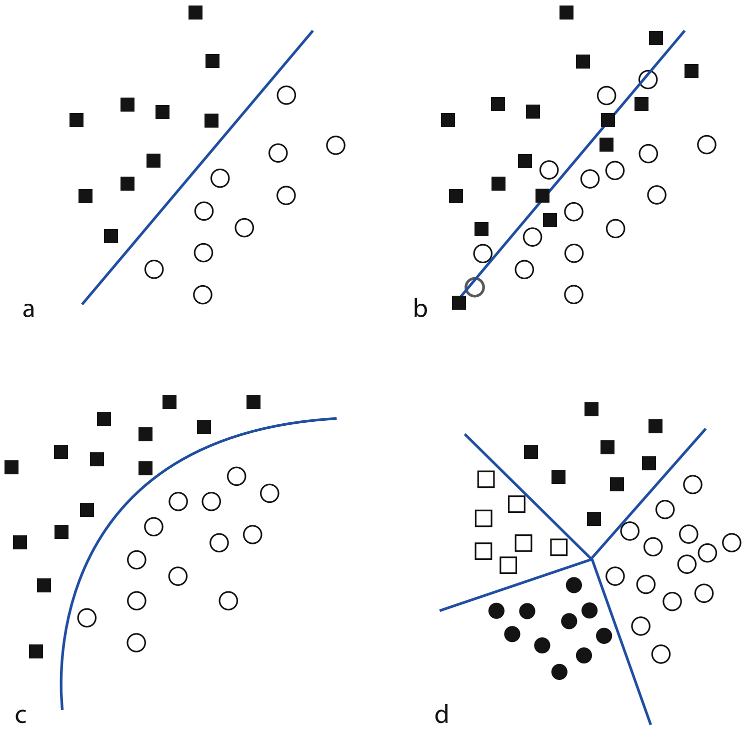


Fig. 24.3
Separation of patterns in feature space. (a) Good linear separation. (b) Linear separation, but statistical variation. (c) Quadratic separation. (d) Multiclass problem with piecewise linear separating regions

The primary objective in any pattern recognition application is the assignment of a specific class (from a finite list of known classes) to data acquired from a specific application. It is also highly desirable to provide a *confidence* value to this assignment. Thus in digit recognition, each input pattern has to be assigned one of ten classes with associated confidence value. In some cases, the test character may not be a digit. In this case, one expects the associated confidence to be very low and the input to be rejected.

In the decision stage, one examines the confidence values of all classes and makes a determination as to what action to take. For example, if the confidence is high enough, the pattern might be recognized and assigned to the best matching class. A different objective arises when it is required to determine whether a given pattern belongs to a specific class or not. Such applications are termed *verification tasks*.

For any given pattern recognition application, the system designer has to address the following questions:

- What are the stored features and how are they derived?
- What is the process of comparison?
- How does one determine the confidence value associated with a specific class?
- What does a confidence value really mean? In most pattern recognition applications, one utilizes a distance metric that generates a numeric value whose magnitude is

a measure of how similar the test pattern features are compared to the reference pattern features. It is often a challenge to convert these numbers to a confidence value in the range 0–100. Scale normalization is frequently required when combining classifiers. As an example one could look at results generated by a statistical classifier using one set of features and combining this with results generated by a neural network using a different set of features. Such issues are often dealt with in an ad hoc manner depending on specific applications.

In this chapter, these questions will be further discussed and several strategies presented to deal with a large variety of applications.

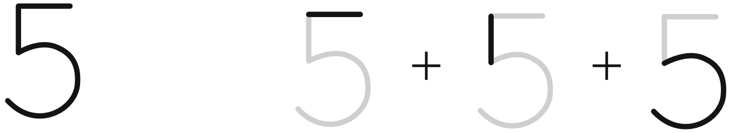
24.1.2 Pattern Recognition Strategies

Pattern recognition strategies can be classified into two broad categories: *statistical* or *structural*. Statistical methods generally require quantitative features extracted from perceptual data, while structural methods rely on primitives (such as edges in an image) and the inherent relationship among the primitives. Primitives are often symbolic (such as horizontal line) and do not have a numerical value. This is illustrated for the character recognition problem shown in ➜ *Fig. 24.4*. Structural recognition uses a grammar that describes the relationship among the primitives. Statistical recognition, on the other hand, utilizes statistical parameters like mean and variance of features to characterize a given class.

In certain applications, it is possible to use both quantitative (statistical) and structural techniques to improve the accuracy of recognition. This hybrid approach has been used very effectively to achieve high accuracies in recognition of handwritten characters [1]. In subsequent sections, only statistical methods will be discussed. Illustrative examples from character recognition and machine vision will be presented to highlight specific strategies. Typical recognition problems involving machine vision are shown in ➜ *Fig. 24.5*. In ➜ *Fig. 24.5a*, the application is the recognition of the stamped text on a gas bottle. Here it is required to design a classification system that can read all stamped text, irrespective of their orientation. In ➜ *Fig. 24.5b*, the problem is verification of the texture of rattan weaving and the detection of any anomalies [22, 23].

24.2 Statistical Methods

Statistical pattern recognition is typically grounded on Bayesian decision theory. In this approach, it is assumed that the underlying probabilistic structure of the features associated with the pattern is known or can be computed. The Bayesian approach is best illustrated with



■ **Fig. 24.4**
Structural features for the character 5, consisting of horizontal line, vertical line, and semicircle

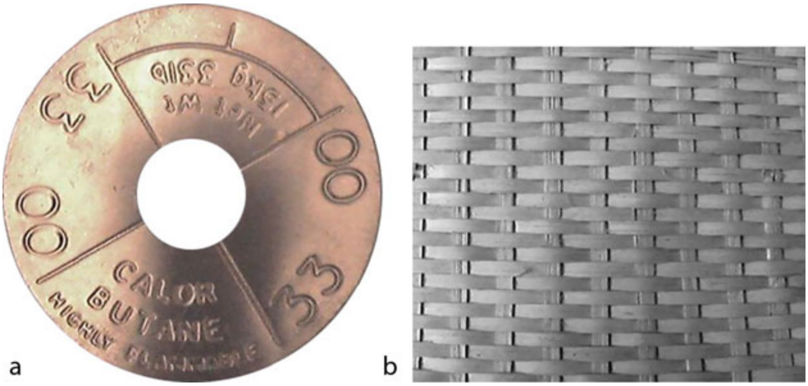


Fig. 24.5
(a) Recognition of stamped text on a gas bottle. (b) Verification of rattan texture

the following example. Suppose we have only two classes, denoted ω_1 andω_2. Here, for a given input feature vector **x**, the task is to decide if **x** belongs to class ω_1 or class ω_2.

Let $P(\omega_1)$ and $P(\omega_2)$ be the prior probabilities of ω_1 and ω_2. If only this information is available, then one assigns **x** to ω_1 if $P(\omega_1) > P(\omega_2)$. On the other hand, if $P(\omega_2) > P(\omega_1)$, then **x** is assigned to ω_2. In the case that $P(\omega_2) = P(\omega_1) = 0.5$, either class can be chosen. Note that here, the classifier always outputs the same class!

Now, suppose we have additional probabilistic information about the classes: the *conditional probability densities* $P(\mathbf{x}|\omega_1)$ and $P(\mathbf{x}|\omega_2)$. Baye's Theorem allows us to write the following expressions:

$$P(\omega_1|\mathbf{x}) = \frac{P(\mathbf{x}|\omega_1)P(\omega_1)}{P(\mathbf{x})} \tag{24.1}$$

$$P(\omega_2|\mathbf{x}) = \frac{P(\mathbf{x}|\omega_2)P(\omega_2)}{P(\mathbf{x})}$$

where

$$P(\mathbf{x}) = \sum_{i=1}^{2} P(\mathbf{x}|\omega_i)P(\omega_i)$$

Hence, the Bayesian decision rule takes the following form: Given a test pattern **x**, assign the pattern as follows:

$$\text{class}(\mathbf{x}) = \begin{cases} \omega_1 \text{ if } P(\omega_1|\mathbf{x}) > P(\omega_2|\mathbf{x}) \\ \omega_2 \text{ if } P(\omega_2|\mathbf{x}) > P(\omega_1|\mathbf{x}) \end{cases} \tag{24.2}$$

It is easily seen that $P(\omega_1|\mathbf{x}) > P(\omega_2|\mathbf{x})$ if

$$\frac{P(\mathbf{x}|\omega_1)P(\omega_1)}{P(\mathbf{x})} > \frac{P(\mathbf{x}|\omega_2)P(\omega_2)}{P(\mathbf{x})}$$

Since the denominator is the same on both sides of the inequality, the decision strategy is simply

$$\text{class}(\mathbf{x}) = \begin{cases} \omega_1 \text{ if } P(\mathbf{x}|\omega_1)P(\omega_1) > P(\mathbf{x}|\omega_2)P(\omega_2) \\ \omega_2 \text{ if } P(\mathbf{x}|\omega_2)P(\omega_2) > P(\mathbf{x}|\omega_1)P(\omega_1) \end{cases} \tag{24.3}$$

In the equally likely case $P(\omega_1) = P(\omega_2)$, we get a simpler decision strategy:

$$\text{class}(\mathbf{x}) = \begin{cases} \omega_1 \text{ if } P(\mathbf{x}|\omega_1) > P(\mathbf{x}|\omega_2) \\ \omega_2 \text{ if } P(\mathbf{x}|\omega_2) > P(\mathbf{x}|\omega_1) \end{cases} \tag{24.4}$$

If accurate estimates of $P(\mathbf{x}|\omega_i)$ can be derived and prior probabilities $P(\omega_1)$ are known, one has an efficient strategy for classifying test pattern **x** as belonging to class ω_1 or ω_2 using (➲ 24.3) or (➲ 24.4). It is straightforward to extend this strategy to recognizing patterns that may come from more than two classes. Assuming that there are p distinct classes ω_i, $i = 1, 2, \ldots, p$ then one assigns a test pattern **x** to ω_k, if

$$P(\mathbf{x}|\omega_k) > P(\mathbf{x}|\omega_i), \quad i = 1, 2, \ldots, p \quad \text{and} \quad i \neq k \tag{24.5}$$

In summary, Bayesian decision theory uses the conditional densities $P(\mathbf{x}|\omega_i)$ as well as the prior probabilities$P(\omega_i)$. This is also one of the challenges of Bayesian decision theory, as in many practical applications, the conditional densities are not readily available. Another factor affecting the Bayesian approach is the nature of the feature vector **x**. **x** could be a scalar in simple applications, or a vector in more complex applications. Also **x** could be continuous or discrete depending on specific recognition applications. A common density function that lends itself to analytic approaches is the normal (Gaussian) density function. The *multivariate normal density* in n dimensions is written as

$$P(\mathbf{x}) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left[-\frac{(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)}{2}\right] \tag{24.6}$$

Here, Σ is the covariance matrix of the feature vector **x, $\boldsymbol{\mu}$** is the mean vector and n is the dimension of **x**.

The unique feature of the Gaussian density is due to its dependence on just two parameters: $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. All relevant measures including higher order moments can be derived in terms of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. In the context of pattern recognition, the density function is evaluated over each class ω_k $k = 1, 2, \ldots, p$. Thus we define

$$p(\mathbf{x}|\omega_k) = \frac{1}{(2\pi)^{n/2}|\Sigma_k|^{1/2}} \exp\left[-\frac{(\mathbf{x}-\mu_k)^T\Sigma_k^{-1}(\mathbf{x}-\mu_k)}{2}\right] \tag{24.7}$$

Here μ_k and Σ_k are the mean vector and covariance matrix for class ω_k. Assuming that the density function of the feature vectors are Gaussian, pattern class assignment described in (➲ 24.5) can be restated using a discriminant function $g_i(\mathbf{x})$. Thus one assigns a test pattern **x** to ω_k when

$$g_k(\mathbf{x}) > g_i(\mathbf{x}), \quad i = 1, 2, \ldots, p \text{ and } i \neq k \tag{24.8}$$

In the case where **x** is described by a Gaussian density function, a suitable form for $g_k(\mathbf{x})$ is the logarithm of the density function:

$$g_k(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\mu_k)^T\Sigma_k^{-1}(\mathbf{x}-\mu_k) - \frac{1}{2}\ln(\Sigma_k) - \ln P(\omega_k) \tag{24.9}$$

If all classes are equally likely, then the last term in (➲ 24.9) may be omitted.

The principal difficulties with using Bayesian classification are the underlying assumptions that prior probabilities $P(\omega_k)$and conditional densities $P(\mathbf{x}|\omega_\mathrm{k})$ are known or can be estimated. In most real world applications, one will have knowledge of design samples for each class and some general knowledge about the classification issues. When $\mathbf{x}$ is a high-dimensional feature vector, the covariance matrix Σ is often ill-conditioned, due to the high correlation among the components of the feature vector. This results in serious numerical errors when computing $\boldsymbol{\Sigma}^{-1}$. Alternative approaches have been proposed to alleviate these errors. For example in (Kimura et al. 1989), the authors recommend the use of a *modified quadratic discriminant function* (MQDF).

It is also worth noting that the discriminant function $g_k(\mathbf{x})$ defined in (24.9) is frequently employed, even if the underlying statistics are not Gaussian, due to the ease with which the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ can be computed.

While statistical methods have found many practical applications, the need for large amounts of training data has often been the primary difficulty in machine vision applications. Neural networks present some advantages in this regard. This will be discussed in the next section.

24.3 Statistical Recognition Case Study: License Plate Recognition

In this section the recognition of license plates used in the Netherlands will be presented. Typical images of license plates are shown in *Fig. 24.6*. A false image that was captured by the camera is also shown in this collection, as well as the license plate of a foreign car. The objective here is to recognize the string of characters that constitute the license plate. In the Netherlands, license plate characters have some predefined structure (Shridhar 1997). The sequence is typically XY-23-CD or XY-PQ-43. There is always one pair of numerals in the license plate string and they usually occur in the second or third set. All license plate strings have six characters in them. For historical reasons, the combinations of SS and SA are not used. Since 1978, the vowels AOEIU and the letters C, Q and W are not used. The principal advantage is that a more robust and accurate license plate recognition system can be developed.

Figure 24.7 shows the actual picture taken by an overhead camera of a vehicle as it moves along on the road.

The recognition of character strings has been the subject of extensive study in the last decade (Fujisawa et al. 2002). Recognition systems based on Maximum Likelihood (ML) or Hidden Markov Model (HMM) remain the most popular techniques for string recognition

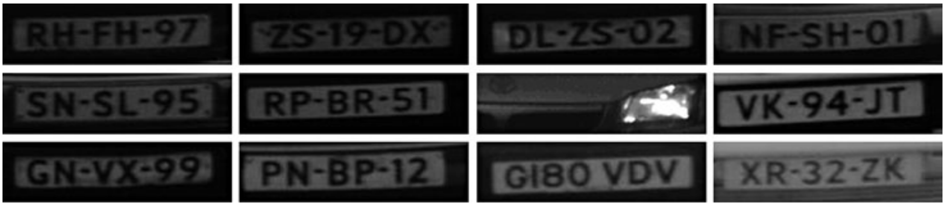


Fig. 24.6
Images of license plates from the Netherlands

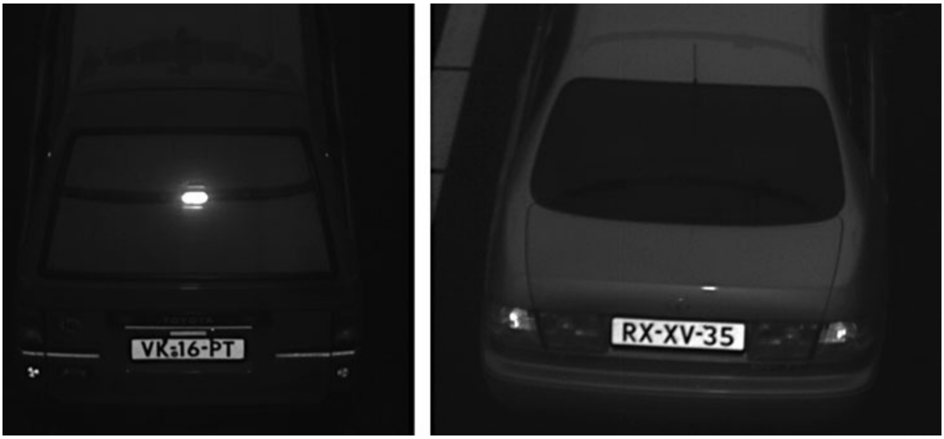


Fig. 24.7
Input images for the license plate recognition problem

[19, 20]. However, these techniques were developed to meet the challenges of handwriting analysis and recognition.

In the case of license plates one observes that characters are for the most part well separated. The authors also observed that at most only two characters were touching or connected. Also, only upper case alphas and numerals are present in the string. The additional character is used: the hyphen. These characteristics suggested that a simplified maximum likelihood technique would be adequate for recognition of the string. Maximum Likelihood (ML) technique was used to achieve the recognition [21]. A 64-dimensional digit classifier and a 64-dimensional alpha classifier, designed with NIST samples were used to derive likelihood measures. Contextual information was used to penalize strings that were not inherent to the data set; for example, the hyphen was not allowed to be located in the second or fourth position in the string. Also, strings whose length exceeded known limits were penalized. ➲ *Figures 24.8* and ➲ *24.9* shows the recognition process used for this study.

The processing involves the following steps:

- Illuminate acquired image through log transformation
- Perform grayscale morphological closing operation on log transformed image
- Subtract closed image from the original image
- Eliminate all non-character blobs in image
- Apply grayscale morphological closing to image that contains only character size blobs. This will cause the character string in the license plate to form a rectangular blob with appropriate aspect ratio
- Extract this region of image containing the license plate and its characters
- Apply string recognition to extract the license plate string.

A limited study with 700 images of license plates captured at 72 dpi and 256 gray levels indicated that an acceptance rate of 80% or higher and an error rate less than 1% is achievable. Some problems remain to be resolved. These problems include ambiguities between similarly shaped characters such as (B – 8), (O – o – 0), (I – 1), (A – 4) and confusion errors from the character pairs like (C – G), (D – O), and (K – X). Some typical results are shown in the table

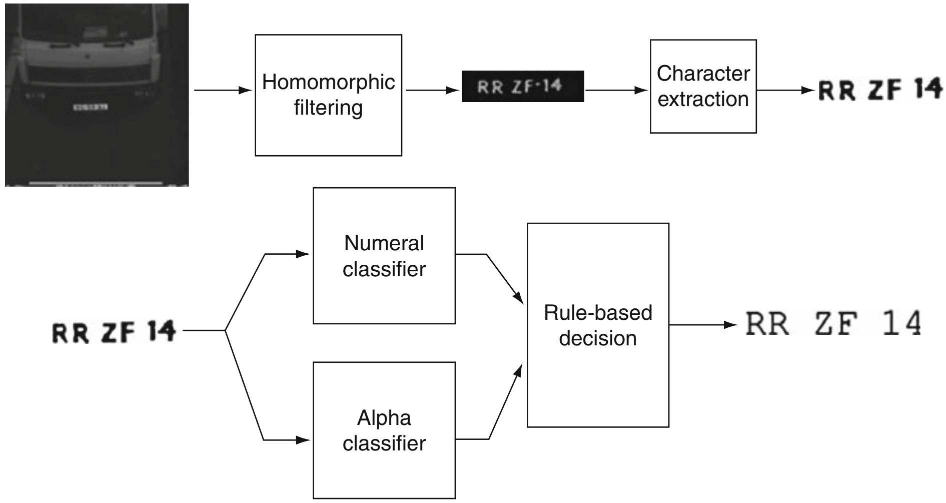


Fig. 24.8
Schematic flow of the license plate recognition process

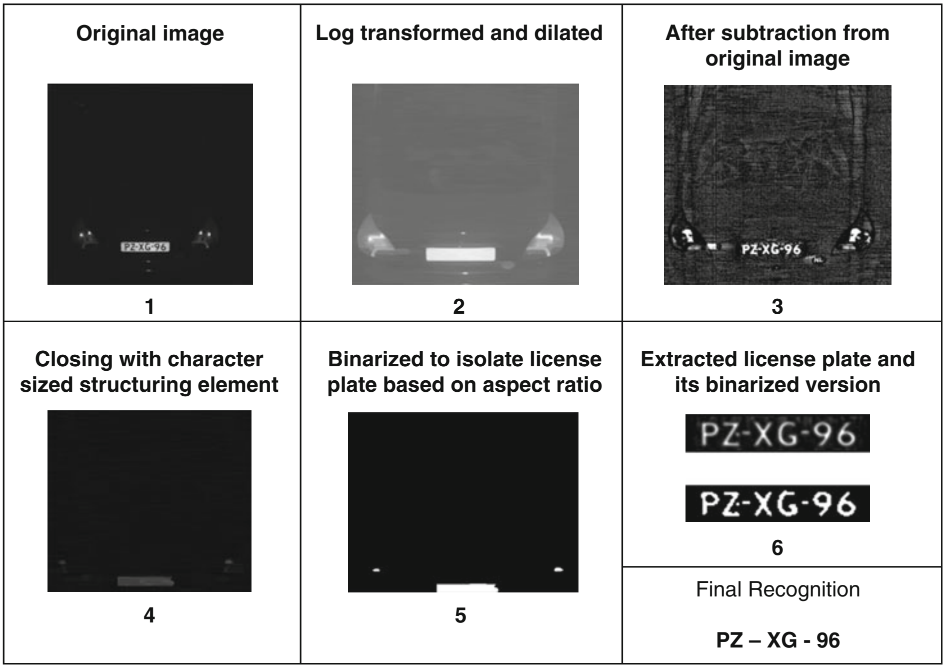
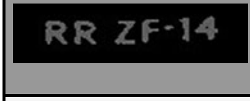

Original image
PZ-XG-96
1
Log transformed and dilated
2
After subtraction from original image
PZ-XG-96
3
Closing with character sized structuring element
4
Binarized to isolate license plate based on aspect ratio
5
Extracted license plate and its binarized version
PZ-XG-96
PZ-XG-96
6
Final Recognition
PZ – XG - 96

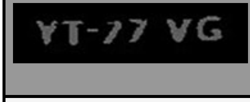
Fig. 24.9
The process of extracting the license plate from the captured image of the car

that follows. The numbers following the recognized character represent the confidence values (or likelihood) generated by the classifier.

Optimum string+ 	Optimum recognition[a]	Recognition from alpha classifier[a]	Recognition from numeral classifier[a]
	4	A (not used)	**4–61**
	1	I – 76 (not used)	**1–79**
	F	**F – 61**	5–56
	Z	**Z – 73**	2–68
	R	**R – 81**	8–59
	R	**R – 78**	2–53

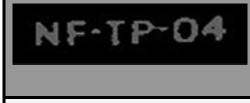
Total likelihood = 433; average likelihood = 72

[a]Correct recognition is shown in bold type; + Hyphen gets removed as noise, depending on its size. The letter "O" is automatically converted to the numeral "0," since "O" is not used in Dutch license plates. Similarly the letter "I" is replaced by the numeral "1"

Optimum string+ 	Optimum recognition[a]	Recognition from alpha classifier[a]	Recognition from numeral classifier[a]
	G	C (**71** –replace with "G")	6–59
	V	**V – 74**	4–45
	7	Y	**7–73**
	7	I	**7–89**
	-	-	-
	T	**T – 87**	7–71
	Y	**Y – 69**	4– 52

Total likelihood = 463; average likelihood = 77

[a]Correct recognition is shown in bold type; + Hyphen gets removed as noise, depending on its size. The letter "O" is automatically converted to the numeral "0," since "O" is not used in Dutch license plates. Similarly the letter "I" is replaced by the numeral "1"

Optimum string+ 	Optimum recognition[a]	Recognition from alpha classifier[a] with confidence	Recognition from numeral classifier[a] with confidence
	4	A – 57 (not used)	**4–69**
	0	**O – 93** (replace with 0)	0–85
	P	**P – 83**	8–56
	T	**T – 88**	8– 61
	F	**F – 79**	5–43
	N	**N – 91**	4–55

Total likelihood = 503; average likelihood = 83

[a] Correct recognition is shown in bold type; + Hyphen gets removed as noise, depending on its size. The letter "O" is automatically converted to the numeral "0," since "O" is not used in Dutch license plates. Similarly the letter "I" is replaced by the numeral "1"

Fig. 24.10
An example frame from the driver video

The characters extracted from the image of the license plate are sent to an alpha classifier as well as a numeric classifier, each yielding a confidence value for the top choice. Using contextual information described previously, the errors due to ambiguities can be resolved

The rule based decision strategy shown in *Fig. 24.10* utilizes the results from the alpha and numeral classifiers and uses the license plate structure to derive three potential strings along with their confidences as shown below:

RR – ZF – 14 with total confidence of 433
RR – 25 – IA with total confidence of 286 (I and A are not used – Reject)
28 – ZF – IA with total confidence of 236 (I and A are not used – Reject)

The total likelihood (confidence) is obtained by adding the confidences associated with each character in the string. This is illustrated by considering the first string RR – ZF – 14. Each of the characters in this string have the confidence values R – 78, R – 81, Z – 73, F – 61, 1 – 79, 4 – 61, which when added together will yield a total confidence of 433 and an average likelihood value of 72.

The three strings above represent the best letter and number pairs with the letter pairs and the number pair in various permissible positions. It is to be noted that strings such as 4B – XY – 8Z are not permitted. In the example above, the string RR – ZF – 14 is chosen as the final recognition.

24.4 Non-parametric Approaches to Pattern Recognition

24.4.1 Eigenfaces

The eigenface algorithm employed here is based on the formulation given in [11, 15].

Briefly, we start with a *database* or *training set* containing sample images of each of the classes of interest. In this problem, there are seven classes of interest. Let m_i represent the number of samples of class i, $i = 1, 2, \ldots, 7$, and let M denote the total number of training samples: $M = m_1 + m_2 + \ldots + m_7$. The training images will be denoted $\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_M$ and each training image is assumed to be an N-dimensional vector of grayscale pixel data. In addition, let t_j represent the facial pose for each training image $\mathbf{I}_j$ so the training set has the following form: $(\mathbf{I}_1, t_1), (\mathbf{I}_2, t_2), \ldots, (\mathbf{I}_M, t_M)$.

An $N \times M$dimensional matrix of zero mean images $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_M]$ can be constructed with column entries $\mathbf{x}_j = \mathbf{I}_j - \mathbf{I}_{ave}$, where $\mathbf{I}_{ave}$ is the average of all the images: $\mathbf{I}_{ave} = \frac{1}{M}\sum_{j=1}^{M} \mathbf{I}_j$.

The covariance matrix of $\mathbf{X}$ is then given by $\mathbf{C} = \mathbf{X}\mathbf{X}^T$. In general, $M << N$ and so rather than compute the eigenvectors of the $N \times N$ matrix $\mathbf{C}$, we compute the eigenvectors of the $M \times M$ matrix $\mathbf{X}^T\mathbf{X}$. The eigenvectors $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_M$ (or eigenfaces) of the covariance matrix $\mathbf{C}$ are then computed as $\mathbf{X} \bullet \mathbf{eig}(\mathbf{X}^{\mathrm{T}}\mathbf{X})$. For each training image, an eigen representation $w_{ij} = \mathbf{e}_j{}^T\mathbf{x}_i$, $j = 1, 2, \ldots, M$ is obtained by projecting the corresponding $\mathbf{x}_i$ onto each of the eigenfaces $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_M$. So each database image is now represented by a M-dimensional vector of eigen coefficients $\mathbf{w}_i$. For $M << N$, this amounts to a significant dimensionality reduction. In addition, the dimension can be further reduced by only considering the eigenvectors which have sufficiently large eigenvalues. We will denote the number of eigenvectors used by M'.

24.4.2 Fisherfaces

While the eigen approach allows for a low-dimensional representation of the training images, it implicitly reduces the discrimination between training images. While this is a useful property for compactly coding images, it may have some undesirable consequences when it comes to classification. The Fisher representation on the other hand, provides better discrimination between classes while preserving the reduced discrimination property of the eigen representation within each class [4].

In the Fisherfaces algorithm [14], we compute the average image within each of the seven classes. That is, for each $i = 1, 2, \dots, 7$

$$\mu_i = \frac{1}{m_i} \sum_{x_j inclassi} \mathbf{x}_j$$

is the average of all the memory set images which are in class i.

The within-class scatter matrix $\mathbf{S}_W$ is an $N \times N$ matrix which provides a measure of the amount of variation among the memory set images within each class and is given by

$$\mathbf{S}_W = \sum_{i=1}^{7} \sum_{xinclassi} (\mathbf{x} - \mu_i)(\mathbf{x} - \mu_i)^T$$

The between-class scatter matrix $\mathbf{S}_B$ is also $N \times N$ and measures the amount of variance between the means of each class, and is computed by:

$$\mathbf{S}_B = \sum_{i=1}^{7} m_i(\mu_i - \mathbf{I}_{ave})(\mu_i - \mathbf{I}_{ave})$$

The Fisher criterion is to choose a linear projection such that the distance between the projected class means is maximized, but the selection is normalized by the within-class scatter. This amounts to computing the eigenvectors of the matrix $\mathbf{S}_W^{-1}\mathbf{S}_B$. Again, since this is an $N \times N$ matrix, the computation of the eigenvalues of $\mathbf{S}_W^{-1}\mathbf{S}_B$ is computationally inefficient. Hence rather than directly computing the eigenvectors, we first project the memory set data onto a lower dimensional space using eigenfaces.

If we let $\bar{\mu}_i = \mu_i - \mathbf{I}_{ave}$, $\mathbf{A} = [\mu_1, \mu_2, \dots, \mu_7]$, $\mathbf{B} = [m_1\mu_1, m_2\mu_2, \dots, m_7\mu_7]$, and $\mathbf{S}_B = \mathbf{BA}^T$, then we can give an effective measure of $\mathbf{S}_W$ and $\mathbf{S}_B$ in the eigenspace coordinates:

$$\bar{\mathbf{S}}_W = \sum_{i=1}^{7} \mathbf{W}_{pca}^T \mathbf{A}_i \mathbf{A}_i^T \mathbf{W}_{pca}$$

and

$$\bar{\mathbf{S}}_B = \mathbf{W}_{pca}^T \mathbf{B} \mathbf{A}^T \mathbf{W}_{pca}$$

where $\mathbf{W}_{pca}$ is the eigenfaces linear projection operator. Finally, the Fisher projection is computed as the eigenvectors of $\bar{\mathbf{S}}_W^{-1}\bar{\mathbf{S}}_B$.

$$\mathbf{W}_{fld} = eig(\bar{\mathbf{S}}_W^{-1}\bar{\mathbf{S}}_B)$$

Once the projection matrix is computed, and just like in the eigenfaces method, the Fisher projection matrix is used to project each of the database images $\mathbf{I}_i$ onto an M'-dimensional vector $\mathbf{w}_i$ of Fisher coefficients.

24.4.3 Nearest Neighbor Classifier

To classify an input image $\mathbf{I}$, we first compute the corresponding zero mean image $\mathbf{x} = \mathbf{I} - \mathbf{I}_{avg}$ and then project the image onto the eigen space: $\mathbf{w} = \mathbf{W}_{pca}\mathbf{x}$ or the Fisher space: $\mathbf{w} = \mathbf{W}_{fld}\mathbf{x}$.

Various classifier designs can be used to map **w** to one of the seven poses (for example, a nearest neighbor classifier or radial basis function classifier, etc.).

In the nearest neighbor classifier, we compute the distance between **w** and the eigen or Fisher representation vectors for each of the database images:

$$\begin{aligned} d_1 &= \|\mathbf{w} - \mathbf{w}_1\| \\ d_2 &= \|\mathbf{w} - \mathbf{w}_2\| \\ &\ldots \\ d_{M'} &= \|\mathbf{w} - \mathbf{w}_{M'}\| \end{aligned}$$

The coefficient vector $\mathbf{w}_i^*$ which gives the smallest such distance is then determined, and **x** is classified to the pose associated with database pattern $\mathbf{w}_i^*$.

24.5 Case Study: Driver Pose Estimation

The problem addressed in this section is stated as follows: Given a video sequence consisting of a driver as he or she drives a vehicle, determine the pose of the driver for each frame in the sequence. An example of a single video frame is shown in ➲ *Fig. 24.10*, and shows the driver looking straight at the road ahead.

Determining the pose of a driver means determining where the driver is looking. And although there are an endless number of possibilities, the National Highway Traffic Safety Administration has formulated seven standard driver pose classes, which are of importance to the ITS community:

Pose 1: Looking over left shoulder
Pose 2: Looking in left rear view mirror
Pose 3: Looking at road ahead
Pose 4: Looking down at radio/instrument panel
Pose 5: Looking at center rear view mirror
Pose 6: Looking at right rear view mirror
Pose 7: Looking over right shoulder

Hence given a video sequence, the problem here is to assign the appropriate class label (1–7) to each frame of the video.

The main reason why the intelligent transportation systems (ITS) community is interested in the pose estimation problem is that the way a driver behaves behind the wheel can give an indication as to the driver's attentiveness and/or fatigue [13, 14, 16]. For example, one study has shown that drivers who are falling asleep at the wheel tend to blink at a faster rate than drivers who are fully awake.

Of course, one way to determine driver pose is by hand, and have a human operator look at the video and mark those frames where the driver changes pose and then, by visual inspection, classify the resulting pose sequences. This process, however, is very tedious and prone to errors. For example, to partition and classify one 20-min video took a student assistant 6 h! Even so, the resulting ground truth contained a number of missed pose transitions and sequences that were misclassified. Clearly, an automatic and computer-based classification method is needed.

In this section, we compare the performance of two non-parametric techniques to determine driver pose from video data: eigenfaces and Fisherfaces. The pose of an input facial image can be determined by first computing the appropriate representation (using a linear projection to project the image onto a smaller dimensional space) and then comparing it to the representation of each standard pose.

24.5.1 Post Classification Filtering

Experimental results [17] indicated that, when presented with temporally correlated data, the nearest neighbor classifier suffers from isolated errors. Meaning, in a sea of correct classifications, there may be a few sporadic misclassifications. To eliminate such misclassifications, we post-process the temporal record of the classifier results. A simple majority rule filter – choosing the pose that appears a majority of the time in a symmetric sliding window of fixed width – eliminates many of these misclassifications without introducing too many new ones. In some cases, the improvements in classification rates are significant.

24.5.2 Experimental Results

MPEG videos of two test subjects were obtained from Transportation Research Center in East Liberty, Ohio: Subject 11 (11092002.mpg) and Subject 46: (46107002.mpg). Images obtained from the MPEG video were cropped to extract the upper left-hand quadrant containing the facial image – the image size was 150(W) × 115(H). Sample poses from the Subject 11 database are shown in *Fig. 24.11*.

The ground truth for each database was obtained by hand-classifying each frame of the video (the same person truthed both videos). For the Subject 11 video, the total number of frames available for testing and training the classifier was 31,000, and for the Subject 46 video, there were 25,000 frames available.

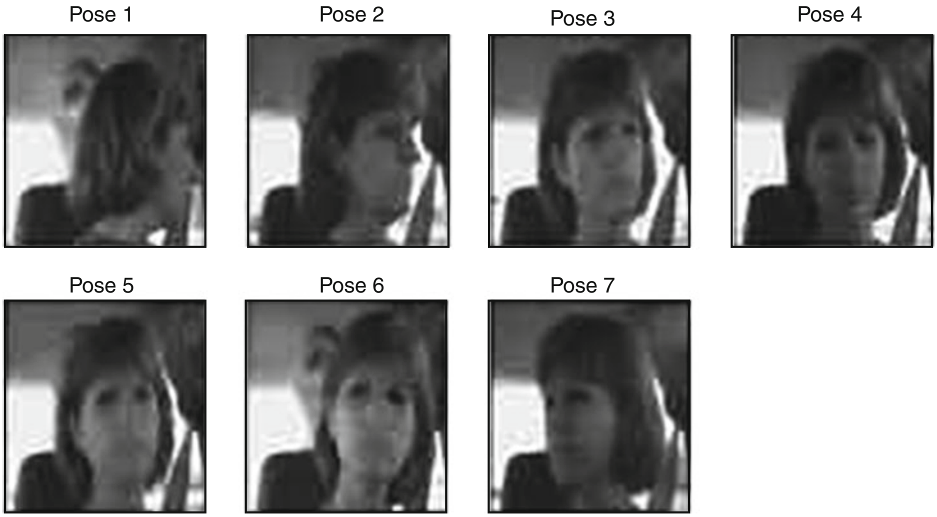


Fig. 24.11
The 7 driver poses

For all of the results given below, the memory set consisted of 50 images per pose – except for pose 3 (the straight pose) which had 150 samples. The reason that more memory set samples were taken for pose 3 is that there were many more samples of pose 3 in the database than any of the other poses. It is to be expected (and hoped for!) that the driver will primarily be looking straight while driving.

For the post-processing majority rule filter, the filter width was fixed at 10 frames.

➲ *Table 24.1* shows the percent correct classification for the Subject 11 database using the eigenface approach and ➲ *Table 24.2* shows the results for the Fisherface approach. ➲ *Tables 24.3* and ➲ *24.4* shows the corresponding results for Subject 46. For these sample experiments, the Fisherface algorithm clearly outperforms the eigenface algorithm in terms of overall percent correct classification. In fact for many other experiments that we conducted (and due to space limitations we cannot report them here), the Fisherface algorithm outperformed the eigenface algorithm. And hence we conclude that the Fisher representation is more suitable for this pose estimation problem.

Table 24.1
Results for Subject 11: Eigenface method. Overall: 82.4% correct

	True pose							
Classified Pose	#1	#2	#3	#4	#5	#6	#7	#8
#1	**93.6**	3.2	1.3	0.3	0.0	0.0	0.0	2.5
#2	0.0	**88.4**	2.4	0.0	3.4	1.7	0.0	2.9
#3	6.4	4.3	**83.9**	13.5	17.0	3.4	2.3	7.8
#4	0.0	1.1	4.5	**84.1**	5.1	0.0	0.0	9.8
#5	0.0	0.0	5.1	1.7	**61.4**	0.0	4.5	3.9
#6	0.0	0.0	1.5	0.0	10.8	**91.9**	4.1	3.3
#7	0.0	0.0	0.4	0.0	0.9	3.0	**89.1**	2.8
#8	0.0	3.0	0.9	0.4	1.4	0.0	0.0	**67.1**

Table 24.2
Results for Subject 11: Fisherface method. Overall: 85.3% correct

	True pose							
Classified pose	#1	#2	#3	#4	#5	#6	#7	#8
#1	**95.7**	3.9	1.0	0.0	0.2	0.0	0.0	1.9
#2	0.0	**88.4**	2.3	0.0	1.6	0.3	0.0	1.5
#3	4.3	3.9	**87.0**	18.1	18.8	3.4	2.3	7.8
#4	0.0	0.0	3.0	**81.0**	1.3	0.0	0.0	7.5
#5	0.0	0.0	3.8	0.7	**70.2**	4.4	4.9	7.0
#6	0.0	0.0	1.8	0.1	6.2	**86.9**	3.4	1.5
#7	0.0	3.9	0.4	0.0	0.5	5.0	**89.5**	2.9
#8	0.0	0.0	0.8	0.1	1.2	0.0	0.0	**69.9**

Table 24.3
Results for Subject 46: Eigenface method. Overall = 84.2%

Classified pose	True pose							
	#1	#2	#3	#4	#5	#6	#7	#8
#1	**96.7**	1.4	0.8	0.0	4.8	0.0	4.8	0.0
#2	0.0	**77.9**	1.6	0.0	0.7	0.0	0.0	3.5
#3	1.6	15.8	**84.6**	0.0	21.6	5.3	7.2	16.9
#4	0.0	0.0	1.9	**100.0**	0.4	0.0	0.5	0.0
#5	0.0	0.0	3.9	0.0	**71.5**	0.8	5.0	0.0
#6	1.6	0.0	1.8	0.0	0.7	**94.0**	1.8	0.0
#7	0.0	0.0	2.3	0.0	0.0	0.0	**80.8**	0.0
#8	0.0	4.9	3.2	0.0	0.4	0.0	0.0	**79.6**

Table 24.4
Results for Subject 46: Fisherface method. Overall: 89.4% correct

Classified pose	True pose							
	#1	#2	#3	#4	#5	#6	#7	#8
#1	**95.7**	3.6	0.8	0.0	0.0	0.0	5.7	0.0
#2	0.0	**87.9**	2.7	0.0	0.4	0.0	0.0	1.8
#3	1.6	8.5	**90.3**	5.8	18.3	4.5	11.8	18.9
#4	0.0	0.0	0.4	**94.2**	0.0	0.0	2.0	0.0
#5	0.0	0.0	2.4	0.0	**78.4**	0.0	4.1	0.0
#6	2.7	0.0	1.7	0.0	0.9	**95.5**	5.4	0.0
#7	0.0	0.0	0.8	0.0	2.0	0.0	**70.6**	0.0
#8	0.0	0.0	0.7	0.0	0.0	0.0	0.5	**79.3**

24.6 Neural Networks

Artificial neural networks are modeled (loosely) after the neural structure of the brain. The brain learns from repeated exposure to specific events. New practical machine vision applications utilizing neural networks have begun to appear in the market; for example, consider the *Eyebot* from Sightech Machine Vision, which can learn up to 20 million features a second (http://www.sightech.com/). Unlike the quadratic discriminant functions associated with statistical methods, neural networks can be trained to generate complex discriminating surfaces.

24.6.1 Artificial Neurons

The basic computational element of a neural network is a single neuron, often called a node. It receives input from external sources (or possibly from other neuron) and computes a single output. Let the n-dimensional input be denoted $\mathbf{x} = [x_1, x_2, \dots, x_n]^T$. Associated with each input is a real-valued parameter w_i called a *weight*. The output of the neuron y is obtained by first computing a weighted sum of inputs:

$$net = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n$$

And then applying an *activation function* f (typically a nonlinear function):

$$y = f(net)$$

A schematic diagram of a neuron is shown in ➲ *Fig. 24.12*.

A few commonly used activation functions are given below and shown in ➲ *Fig. 24.13*.

1. Logistic activation

$$f(net) = \frac{1}{1 + e^{-net}}$$

2. Hyperbolic tangent activation

$$f(net) = \tanh(net) = \frac{e^{net} - e^{-net}}{e^{net} + e^{-net}}$$

3. Linear activation function

$$f(net) = \beta net, \text{ where } \beta \text{ is the slope,}$$

4. Threshold activation

$$f(net) = \begin{cases} 1 \text{ if } net \geq 0 \\ -1 \text{ otherwise} \end{cases}$$

The choice of the activation function depends on the application and the nature of the required data processing. For example, if a real-valued output between 0 and 1 is required, then a logistic activation function is appropriate; if the output is required to be bipolar binary, then a threshold activation is appropriate. Note that for backprop learning, a smooth activation function is required. Oftentimes, the network will be trained with sigmoidal functions and then when training is done, the activation function of the output units will be replaced with threshold functions.

The computation performed by a single neuron is rather simple. However, by interconnecting these simple processors in a network structure can vastly increase the computational power. The most common network structure is the 2-layer model shown in

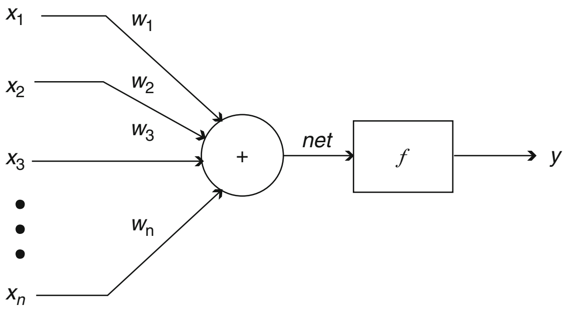


Fig. 24.12
Model of a neuron

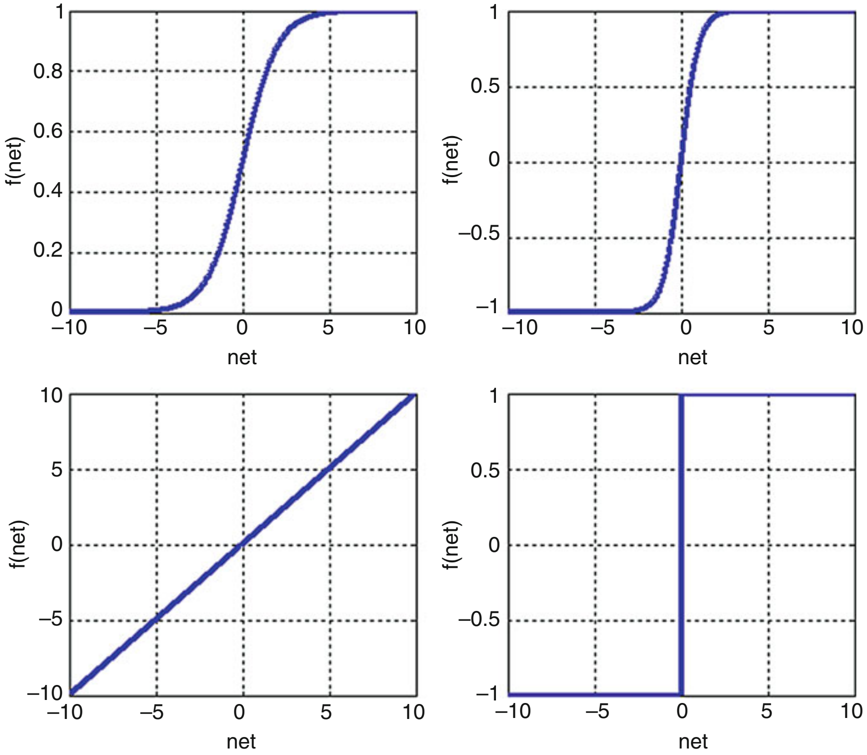


Fig. 24.13
Commonly used activation functions

Fig. 24.14. Here, the input **x** is connected to each neuron in the first layer, called the *hidden layer*. Each hidden unit computes a weighted sum, applies an activation function f_h and produces an output. Let the hidden unit outputs be denoted $\mathbf{z} = [z_1, z_2, \ldots, z_J]^T$. An expression for **z** can be written as $\mathbf{z} = F_h(\mathbf{Vx})$, where $F : R^J \rightarrow R^J$ is a vector mapping of all the hidden unit activation functions: $F_h(\mathbf{net}) = [f_h(net_1), f_h(net_2), \ldots, f_h(net_J)]^T$. The hidden unit outputs are inputs to the output layer of neurons. The final network outputs $\mathbf{y} = [y_1, y_2, \ldots, y_K]^T$ are given by $\mathbf{y} = F_o(\mathbf{Wz})$.

The design of a neural network for pattern classification applications is a non-trivial task, and requires addressing the following issues:

- Selection of external inputs (features) to the neural network
- Construction of a training set of representative examples
- Choice of activation function
- Size of the hidden layer and the number of hidden layers to use
- Learning algorithm to determine the weights

Determination of the weights usually follows an iterative process, called the *learning phase*. A widely used algorithm for determining the weights is called the *backpropagation*.

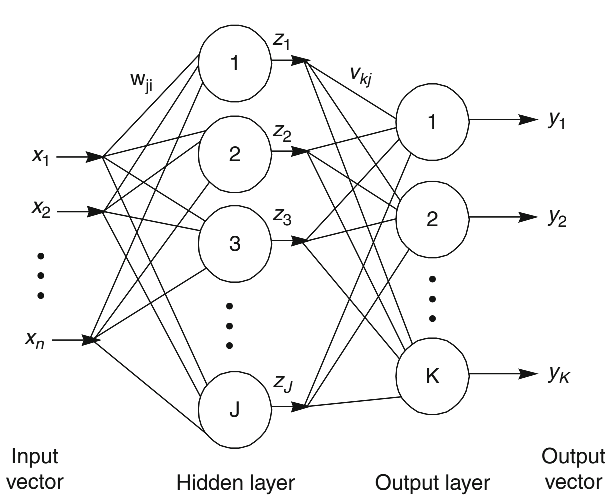


Fig. 24.14
A two layer artificial neural network

24.6.2 Backpropagation Algorithm

The backpropagation algorithm is a method for supervised training of multilayer neural networks. In this case, we assume we have a training set of inputs and associated desired or *target* outputs: $\{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_m, \mathbf{y}_m)\}$. There are two phases of neural network operation: *training* and *testing*. The training phase consists of presenting a pattern to the input and passing the signals through the network in order to yield outputs from the output units. The weights are updated according to the chosen learning rule (by using the known target values). In the test phase, a different set of data is used to asses the performance of the system. In this case, test inputs are processed, but no weight adjustment is done (called feedforward operation).

Given a network topology (i.e., given the values of n, J, and L) and given the hidden layer weights $\mathrm{W} = [w_{ji}]$ and the output layer weights $V = [v_{kj}]$, the output of the network $\mathbf{y}$ can be computed for any input vector $\mathbf{x}$ using (24.3–24.6). The process of computing the output given an input vector is called a *forward pass* of the network because of its left-to-right chain of computations: starting at the left with the input $\mathbf{x}$, we compute the hidden unit weighted sums net_j, the hidden unit outputs, z_j, the output unit weighted sums net_k, and then finally the network outputs y_k.

A feedforward artificial neural network may be viewed as a multiple input-multiple output system which maps an n-dimensional input vector $\mathbf{x}$ to a K-dimensional output vector $\mathbf{y}$. The actual input-output mapping implemented by the neural net depends on its architecture (that is, the values of n, J, and K) and the setting of its internal parameters (that is, the value of the weights).

One of the most popular and extensively studied learning algorithms for such feedforward networks is called error backpropagation, or simply *backprop* [10], and is briefly explained below. In this case, we start with the *sum-squared error function* (SSE):

$$E = \frac{1}{2}\sum_{m=1}^{M} \| d^m - y^m \|$$

This SSE error function is taken as the objective function of an unconstrained optimization problem. In particular, the problem is to find network weights w_{ji}^* and v_{kj}^* such that E is minimized. This optimization problem can be solved using the gradient descent approach:

$$w_{ji}^{t+1} = w_{ji}^{t} - \rho \frac{\partial E}{\partial w_{ji}}$$

$$v_{kj}^{t+1} = v_{kj}^{t} - \rho \frac{\partial E}{\partial w_{kj}}$$

Here, the gradient descent step size $\rho > 0$ is called the *learning rate*. If ρ is set too small, then learning will take a long time. On the other hand, if ρ is too large, the gradient descent equations become unstable and learning reduces to random search. The learning rate is typically adjusted empirically to balance between these two extremes. In this learning scheme, given a training pair (**x, d**) randomly chosen from the training set, the output layer weights are modified by the following amount:

$$\Delta v_{kj} = \rho(d_k - y_k) f'(net_k) z_j \tag{24.10}$$

where ρ is a small positive constant called the learning rate, and $f'(net_k)$ is the derivative of the activation function $f(net)$ evaluated at net_k; that is, $f'(net)|_{net=net_k}$.

It is easy to verify that the derivative of the activation functions in ➲ *Fig. 24.1b* is given as follows:

1. $f'(net) = f(net)(1 - f(net))$ for the logistic activation,
2. $f'(net) = [1 - f^2(net)]$ for the hyperbolic tangent activation.
3. $f(net) = \beta$ for the linear activation, and
4. Since the threshold function is not differentiable (more precisely, not differentiable at the origin), it is not suitable for backprop learning.

The hidden layer weights are modified by the following amount:

$$\Delta w_{ji} = \rho \left(\sum_{k=1}^{K} (d_k - y_k) f'(net_k) w_{kj} \right) f'(net_j) x_i \tag{24.11}$$

Training proceeds by randomly choosing a training pattern from the training set, computing the required weight changes using (➲ 24.10) and (➲ 24.11), and then updating the hidden and output layer weights. There are various ways to check for convergence of the algorithm. The simplest way, though, is to choose a maximum number if iterations C and keep training until this number of training steps is achieved. The entire algorithm is summarized below.

Backprop algorithm

1. Choose the network structure, i.e., choose values for n, J, and K. Initialize all hidden and output layer weights to small random numbers, select a small learning rate $\rho > 0$, choose a desired number of learning iterations C, and initialize the iteration counter: $t = 0$.
2. Increment the count: $t \leftarrow t + 1$. Select an input pattern $\mathbf{x}^t$ from the training set and compute the following quantities:

$$net_j \text{ and } z_j \text{ for each } j = 1, 2, \ldots, J;$$
$$net_k \text{ and } y_k \text{ for each } k = 1, 2, \ldots, K.$$

3. Using the desired target $\mathbf{d}^t$ associated with $\mathbf{x}^t$ employ Equations (➲ 24.10) and (➲ 24.11) to compute the output layer weight changes Δv_{ij} and hidden layer weight changes Δw_{ji}.
4. Update all output layer weights: $v_{kj} \leftarrow v_{kj} + \Delta v_{kj}$ and hidden layer weights: $w_{ji} \leftarrow w_{ji} + \Delta w_{ji}$.
5. Test for convergence: $t \geq C$. If convergence criterion is met, then end; otherwise, go to step 2.

The above process of adaptively adjusting the weights of the network is called the *training phase*. Once the training phase is complete, the weights of the network are frozen at their final values and the network can be tested with patterns from the training set as well as additional patterns which were not used during training; this is called the testing or *testing phase*. The ability of the neural net to supply reasonable outputs for entirely new inputs (i.e., inputs not "seen by" or "presented to" the network during training) is called *generalization*.

References

1. Kimura F, Shridhar M (1991) Handwritten numeral recognition based on multiple algorithms. Pattern Recognit 24(10):969–983
2. Ball P (2009) Shapes: nature's patterns: a tapestry in three parts (nature's patterns). Oxford University Press, Oxford
3. Beck R (2007) World history: patterns of interaction. Holt McDougal, Evanston
4. Belhumeur P, Hespanha J, Kreigman D (1997) Eigenfaces vs. fisherfaces: recognition using class specific linear projection. IEEE Trans Pattern Anal Mach Intell 19(7):711–720
5. Eyebot Vision Processor (2010) SIGHTech vision systems Inc http://www.diffley-wright.com/eyebot.htm
6. Hassoun M (1995) Fundamentals of artificial neural networks. MIT Press, Cambridge
7. Patel A (2008) Music, language, and the brain. Oxford University Press, Oxford
8. Pinker S (2007) The language instinct: how the mind creates language (P.S.). Harper, New York
9. Rosnow R, Rosenthal R (2006) Beginning behavioural research: a conceptual primer, 6th edn. Thompson, Belmont
10. Rumelhart DE, Hinton GE, Williams RJ (1986) Learning internal representations by error propagation. In: Rumelhart DE, McClelland JL (eds) Parallel distributed processing: explorations in the microstructure of cognition, vol 1. The MIT Press, Cambridge, pp 318–362
11. Sirovich L, Kirby M (1987) Low-dimensional procedure for the characterization of human faces. J Opt Soc Am 4(3):519–524
12. Taleb N (2007) The black swan: the impact of the highly improbable. Random House, New York
13. Tijerina L, Johnston S, Parmer E, Winterbottom M, Goodman M (2000) Driver distraction with wireless telecommunications and route guidance systems. *National Highway Traffic Safety Administration, Report No. DOT HS 809–069*
14. Tomoaki N, Sugiyama K, Mizuno M, Yamamoto S (1998) Blink measurement of image processing and application to warning of driver's drowsiness in automobiles. In: Proceedings of IEEE/ICTV 2
15. Turk M, Pentland A (1991) Eigenfaces for recognition. J Cog Neurosci 3(1):71–86
16. Tyeruna L, Gleckler M, Stoltzfus D, Johnson S, Goodman M, Wierville W (1999) A preliminary assessment of algorithms for drowsy and inattentive driver detection on the road. *DOT Technical Report*
17. Watta P, Lakshmanan S, Hou Y (2007) Nonparametric approaches for estimating driver pose. IEEE Trans Intell Transp Syst 56(4):2028–2041
18. Kimura F, Shridhar M (1992) Segmentation-recognition algorithm for zip code field recognition. Mach Vision Appl 5:199–210
19. Kundu A, Yang He, Barl P (1989) Recognition of handwritten word: first and second order hidden markov model based approach. Pattern Recognit 22(3):283–297
20. Kimura F, Miyake Y, Shridhar M (1995) Zip code recognition using lexicon free word recognition algorithm. In: Proceedings of 3rd ICDAR, Montreal, pp 906–910
21. Shridhar M, Miller J, Houle GF, Bijnagte L (1999) Recognition of license plate images: issues and

perspectives. In: Proceedings of ICDAR 99, Montreal, pp 17–20

22. Smith G, Burns I (1997) Measuring texture classification algorithms. Pattern Recognit Lett 18(14):1495–1501
23. Tsatsanis MK, Giannakis GB (1992) Object and texture classification using higher order statistics. IEEE Trans Pattern Anal Mach Intell 14(7):733–750
24. Kimura F, Takashina K, Tsuruoka S, Miyake Y (1987) Modified quadratic discriminate functions and its application to Chinese character recognition. IEEE Trans Pattern Anal Mach Intell 9(1):149–153
25. Cheng-Lin Liu, Hiroshi S, Hiromichi F (2002) Performance evaluation of pattern classifiers for handwritten character recognition. Int J Document Anal Recognit 4:191–204

25 Implementing Machine Vision Systems Using FPGAs

Donald Bailey
Massey University, Palmerston North, New Zealand

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_25,

Abstract: Field programmable gate arrays (FPGAs) offer a convenient and flexible platform on which real-time machine vision systems may be implemented. This chapter discusses how FPGA based design differs from conventional software design for implementing image processing algorithms. Simply porting from software to hardware can give disappointing results. The process of transforming an algorithm for mapping it onto an FPGA is described in some detail, illustrating how a range of low-level image processing operations may be efficiently implemented. Systems integration issues such as interfacing, debugging and tuning are also discussed.

25.1 Introduction

As outlined earlier in this book, machine vision is the integration of vision, or image processing, as part of the control system for a machine. This requires that the system capture images and automatically analyse them to obtain the necessary control data for directly controlling the machine to perform its required task [34]. Throughout this process, the timeliness of the whole process is critical. Machine vision generally focuses on real-time operation. Here, real-time is defined as the ability to provide the data extracted from the images in sufficient time for the processing task to be performed effectively.

This section considers the real-time requirements of machine vision systems, and how these can be overcome by exploiting the parallelism inherent within image processing algorithms. In ➲ Sect. 25.2, FPGAs are presented as a potential platform for the flexible implementation of parallel algorithms in hardware. ➲ Section 25.3 outlines the design process when using FPGAs for image processing. Particular attention is given to the differences between software and hardware based implementations of algorithms. Many of the transformations required in mapping an algorithm onto hardware are demonstrated in ➲ Sect. 25.4. The examples chosen illustrate some of the key techniques, with some of the tradeoffs for a range of low level and intermediate level image processing operations discussed in ➲ Sect. 25.5. ➲ Section 25.6 describes how external devices may be interfaced to the FPGA, with particular focus on image capture, display, and interacting with the user where necessary. In any complex system, debugging and algorithm tuning are of particular importance. The design issues associated with these are elaborated in ➲ Sect. 25.7. The final section brings everything together, and analyses some of the advantages and limitations of using FPGAs as the implementation platform for machine vision systems.

25.1.1 Real Time Processing

There are two aspects to real-time operation that are important to distinguish between. The first is the processing latency. This is defined as the time from when an image capture is triggered until the time that the processed output is available. The second is the throughput. This corresponds to the average rate at which images are processed and the required data is produced. Note that if there is a wide variation in the processing time for individual images then significant storage may be required. In this situation, the worst case processing rate may be a more relevant measure than the average. On a conventional computer the processing rate and latency are intimately related because the processor can only do one thing at a time. However, as will be seen shortly, in a parallel processing system the processing rate and latency can be decoupled. Different machine vision applications may place different constraints on the latency and throughput.

When using vision to provide direct feedback for a control task, the latency is usually a critical constraint. Delays within the feedback path complicate the design of a control system, and can directly lead to instability. However, in an inspection task, latency may not necessarily be the critical constraint. When grading, for example on a conveyor, the critical constraint is the time from when the image is captured to the first actuation point. This may be arranged at some distance (hence time) after the image capture, resulting in a very loose latency constraint. On the other hand, a system for 360° inspection may perform the image capture while the item is in freefall, and a decision must be reached to control a compressed air jet to deflect the item before it lands. In this case, there is a tight latency constraint.

In control tasks, the sample rate of the system governs the throughput. This is usually taken to mean that all of the processing must be completed within the time between samples [27] (usually the video frame rate). However, in a parallel system only each processing step needs to be completed at this rate if the data can be passed on to another processing stage. The sampling rate (or frame rate) must be sufficiently high to accurately capture any high speed phenomena. From the Nyquist sampling theorem, there must be at least two samples or frames within the highest frequency variation for it to be captured accurately. This applies a lower limit to the frame rate and hence throughput. For inspection applications, the throughput is governed by the rate at which items are presented for inspection.

Both the latency and throughput place severe constraints on the image processing system. Many vision applications require high performance computing to operate at the required rates. To achieve the required processing rates, it is often necessary to exploit the parallelism inherent in image processing algorithms [27]. Although this problem has to some extent been alleviated by the ever increasing processing power as a result of Moore's law, the complexity of the problems and expectations on the system have also increased, requiring some form of parallel solution.

25.1.2 Parallelism in Image Processing

When accelerating an algorithm by exploiting parallelism, it is important to consider how much of the algorithm is inherently sequential, and how much of the algorithm is parallelisable. The serial portions of the algorithm cannot be sped up. Therefore, if 20% (for example) of an algorithm is constrained to run serially, then the minimum latency is 1/5 of the original processing time, or a maximum of only five times speedup can ever be obtained. Such a speedup can only be achieved if the remaining 80% of the algorithm can be implemented in parallel with the serial 20% (on multiple processors if necessary). This is known as Amdahl's law of parallelism [6]. To achieve any significant speedup, the algorithm must have a high proportion of its operations that are parallelisable, and that implementing them in parallel does not introduce any significant additional overhead, from communication or intermediate storage. Note that the speedup here is in terms of the latency; it is always possible to increase the throughput, even if by using a separate processor for each image. Fortunately, there are several forms of parallelism inherent within the processing of images [18]. These will be described in turn.

At the highest level, an image processing algorithm consists of a sequence of image processing operations. These operations are independent in the sense that the processing performed by each operation is independent of the processing performed by each of the other operations. The only dependencies are the flow of data from one operation to the next through the processing chain. The captured image is input at the start of the pipeline and as it

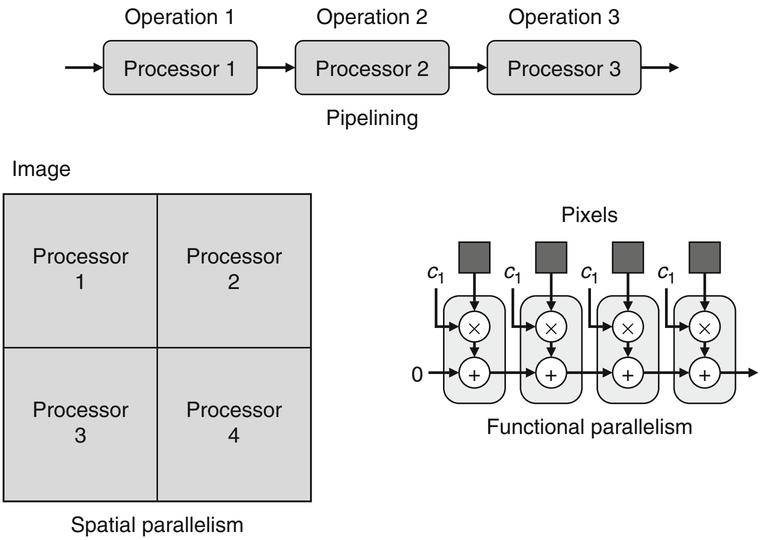


Fig. 25.1
Different types of parallelism within image processing

flows through the sequence of operations, the data is transformed until at the output the required data is produced. This temporal parallelism may be exploited by pipelining – a separate processor is used for performing each operation within the algorithm, as shown in *Fig. 25.1*. Such pipelines work like assembly lines. At a given stage, once the input is available, the operation may be performed on the image with the results passed on to the next stage in the chain. It may then begin processing the next input, without having to wait until the processing for the previous image is complete. The independence of the operations allows each processor to work independently of the others, subject to the flow of data between the operations. The latency is determined by how long it takes for an image to pass completely through the pipeline. The throughput, however, is governed by the one operation that takes the longest time to complete. Pipelining, therefore, decouples the throughput and latency because processing may begin on a new image before the processing is completed on a previous image.

At the next level, many image processing operations perform the same operation for each pixel within the image. In software, this is represented by the outermost loop iterating through the pixels in the image. This therefore corresponds to spatial parallelism, which may be exploited by partitioning the image among several independent processors, as illustrated in *Fig. 25.1*. The main factor in determining the best partitioning is to minimise the communication between processors, because any such communication is an overhead and reduces the effective speedup that can be achieved. Minimising communication usually corresponds to minimising the data required from other partitions. Such partitioning is therefore most effective when only local processing is required. This is one of the drivers for investigating separable algorithms, because separating row and column processing provides a partitioning that has inherently low communication overheads.

Stream processing is another way in which spatial parallelism may be exploited. Bandwidth constraints mean that images are usually passed sequentially from one operation to the next in a pipeline. Serialising the image data, using a raster scan, converts spatial parallelism into temporal parallelism. In a stream processing system, each stage of the processing pipeline operates one pixel at a time, rather than a whole image at a time. One simple example of such a system is the use of a lookup within an image capture card to provide contrast adjustment on an image as it is captured. Since each pixel is only provided once on the input, it is necessary for any operation that uses more than one pixel to buffer the input pixels until they are required. Stream processing therefore works best when only local information is required and the processing time for each pixel is constant. It is also the natural processing mode for data being streamed from a camera or to a display. Note that stream processing adds a throughput constraint, generally one pixel per clock cycle. However, the advantage is a significantly reduced latency, because each operation in a pipeline can begin processing as soon as pixels start arriving, without having to wait until the complete image is available.

At the lowest level is logical or functional parallelism. This is where a single function block is reused many times within an operation. For example, a linear window filter implemented in software would consist of an inner loop which iterates through the pixels within a window, multiplying each pixel by the corresponding filter coefficient, and adding the result to an accumulator. The content of the inner loop – the multiply and accumulate – is repeated many times, but could readily be parallelised by building several multiply and accumulate blocks (see ➲ *Fig. 25.1*). Since these inner loops are where most of the processing time is spent, from Amdahl's law, parallelising these can give the largest processing speedup, and can be the most effective at minimising the latency.

In the past, to accelerate the image processing algorithms by exploiting parallelism, it was necessary to develop dedicated image processing hardware. While such systems were fast, they were both expensive and relatively inflexible. As long as the task did not change, it could be performed well, but it was difficult to reconfigure the system in the event of even relatively minor changes to the task. The introduction of field programmable gate arrays (FPGAs) in the 1980s allowed hardware to be more flexibly reprogrammed. While the early devices were too small for implementing image processing algorithms, modern FPGAs now have sufficient resources to enable complete systems to be implemented on a single chip.

25.2 FPGAs

In its most basic form, a field programmable gate array is essentially a sea of programmable logic gates, combined with programmable interconnect, as illustrated in ➲ *Fig. 25.2*. The logic is performs the required computation, while the interconnect is used to connect the data from one computation point to another. The two main vendors for FPGAs are Xilinx and Altera, although there are a number of other smaller players.

This section will explore the architecture of currently available FPGAs (the latest generations are the Spartan 6 [40] and Virtex 6 [42] from Xilinx, and Cyclone IV [4] and Stratix IV [5] from Altera) and how this can support the parallel implementation of image processing algorithms. This section concludes with a discussion of some of the different approaches to programming FPGAs.

25.2.1 FPGA Architecture

The basic logic cell in most FPGAs is a small, single bit wide, lookup table (LUT). This enables it to implement any arbitrary Boolean function of the inputs, with propagation delay

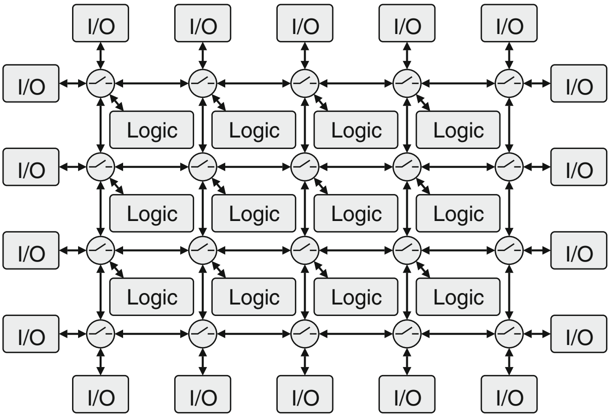


Fig. 25.2
Basic structure of an FPGA

independent of the actual function. More complex functions requiring more inputs can be created by having multiple levels of LUTs. In this way any arbitrarily complex logic function can be implemented.

Previous generations of FPGAs were based around 4-input LUTs, although this has increased with reducing fabrication technology scale. The basic logic element in the latest Xilinx FPGAs is a 6-input LUT, which can be split into two separate 5-input LUTs with the same inputs [42]. Altera have taken a slightly different approach. Their adaptive logic module [5, 25] combines up to 8 inputs with two adaptive LUTs that can vary between 3 and 6 inputs each. Each LUT also has a flip-flop on the output that may be optionally bypassed. These may be used as registers for holding intermediate results, or to demarcate the stages within a pipelined design. In addition to the LUTs, modern FPGAs also incorporate additional logic to simplify and reduce the propagation delay required to implement common operations. These include dedicated logic for carry propagation to speed addition, and support for wide multiplexers. The arrangement of logic cells within a Spartan 6 is illustrated in *Fig. 25.3*.

Each logic block is made up of a number of logic cells (8 in the case of Xilinx, 10 for Altera). Grouping logic cells in this way enables higher speed direct connections between cells within the same logic block. This reduces the propagation delay when implementing more complex logic functions. Otherwise, connections between blocks are via the programmable interconnect. The interconnect network is typically arranged in a grid structure, with switches at intersections enabling signals to be routed between the horizontal and vertical lines. Each interconnection segment and each switch introduces propagation delay. In a typical design, approximately half of the propagation delay comes from the logic, with the remaining delay from the interconnect.

For digital signal processing (DSP), a common operation is multiplication. It is expensive (both in terms of the logic required and the propagation delay) to build multipliers out of logic blocks. Therefore, in addition to the fine-grained logic blocks, most modern FPGAs also have dedicated multiply and accumulate or DSP blocks. DSP blocks can also include dedicated word-wide shift registers to simplify the implementation of finite impulse response filters.

In addition to the flip-flops associated with each logic cell, FPGAs also include on-chip memory blocks for storage of larger volumes of data. While the large, high-end FPGAs have sufficient storage for one or more images, this is usually poor value for money, and it

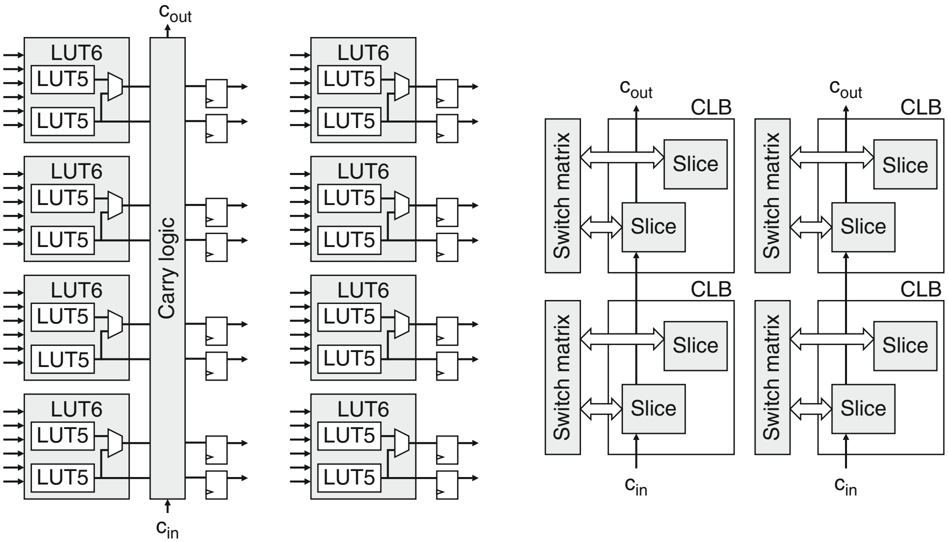


Fig. 25.3
Logic cells within a Spartan 6. *Left*: the arrangement of LUTs within slices. *Right*: the arrangement of slices as logic blocks, showing the linking of the carry chains. Simplified from [41]

is usually better to use off-chip memory for bulk storage. However the on-chip memories enable high bandwidth intermediate storage of data. The memory is implemented in two ways: either by adapting the logic blocks to use the LUTs as small memories, or as separate dedicated memory blocks.

Xilinx allow up to 50% of the logic elements to be used as separate 32-bit memories (giving up to 256 bits per logic block) for both the Spartan 6 and Virtex 6. These may be used as ROM, RAM (either single port or dual port), or programmable length shift registers. In their Stratix devices, Altera require that a whole logic block be used either for logic or as a memory, giving 640 bits per logic block. This memory is simple dual-port (one port is read/write, and the other is read only). These small memories are useful for short buffers for example for synchronising different parts of a design, or for storing coefficients.

In addition to the small blocks, FPGAs also have larger blocks of dual-port memory. In the Spartan-6, these memory blocks are 18 Kbits each (configurable in aspect ratios from 16 K × 1 up to 512 × 36). In the Virtex-6 they are larger at 36 Kbits each (32 K × 1 to 512 × 72), although they can also be configured as two independent 18 K memories. In the Cyclone IV, the blocks are 9 Kbits each (8 K × 1 to 256 × 36). The Stratix IV also has 9 Kbit blocks, but also adds larger 144 Kbit blocks (16 K × 8 to 2 K × 72). On the high end devices (the Virtex-6 and Stratix IV), the blocks may also be configured as FIFO buffers, with memory addressing provided by dedicated hardware. The 72-bit wide memories can also be configured to use 8 of the bits for hardware error detection and correction with 64 bit wide data. The block memories are reasonably plentiful, with each one accessed independently, allowing them to be used for row buffers and wide bandwidth caches.

The input and output blocks provide buffers and drivers to connect between internal logic and external devices. General purpose I/O pins can be configured to operate with a wide range of signalling standards, and include programmable termination for high speed signalling.

Other standards are also supported, including interfaces to double data rate (DDR) memories, and low voltage differential signalling (LVDS). High end devices also support multi-gigabit serial communication standards with built in serialisation and deserialisation logic to enable the FPGA core to run at a lower speed.

As synchronous devices, FPGAs require one or more clock signals to control their timing. Delay locked loops are used to synchronise clocks with external sources, and to minimise clock skew across the FPGA. Phase locked loops are used to synthesise new clock frequencies by multiplying a reference clock by an integer ratio. A series of dedicated clock lines and clock buffers distribute the clock signals across the FPGA and manage the high fanout while minimising skew between different parts of the chip.

A typical layout of the various logic components within the Xilinx FPGAs is shown in ➔ *Fig. 25.4*. The different elements are arranged in columns within the FPGA, with the I/O logic around the edges. The structure of the Altera devices is a little different, but follows the same principle of interspersing the DSP and RAM blocks throughout the logic.

All of the configurable components of the FPGA (LUTs, flip-flops, interconnect, DSP, RAM options) are controlled by static memory (SRAM) cells. Since these are volatile, the configuration must be reloaded after powering on before any processing can begin. This is achieved by loading a configuration file from an external ROM into the internal control memory (some devices contain an internal configuration ROM to reduce the component count). Some devices allow compression of the configuration file to reduce configuration time, and encryption of the configuration data to provide some protection against piracy or reverse engineering of the intellectual property. All of this uses dedicated logic on the FPGA to decompress, decrypt, and direct the configuration bits to their corresponding SRAM cells. An FPGA can be reconfigured at any time. This requires suspending the current execution, and loading a new configuration file. Since everything on the FPGA (including RAM contents) is changed by the new configuration, it is necessary to save any state information off the FPGA during this process. The Xilinx devices allow partial reconfiguration, which allows part of the device to be reconfigured without stopping execution on the remainder of the FPGA.

In summary, modern FPGAs consist of a homogenous mixture of fine grained logic and flip-flops, and coarser grained DSP blocks and memory, with a programmable interconnect that allows these to be configured to implement parallel hardware systems.

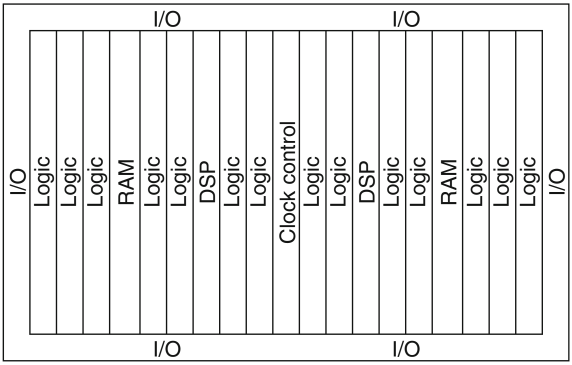


Fig. 25.4
Typical layout of Xilinx FPGAs

25.2.2 FPGAs for Image Processing

The parallel nature of FPGAs allows them to implement parallel algorithms and to exploit the parallelism inherent in images. Since each image processing operation is different, the very nature of hardware requires that separate operations be implemented by separate hardware, unless efforts are made to explicitly reuse various components. Implementation of an image processing algorithm on hardware therefore directly lends itself to coarse-grained pipelining.

In hardware, it is relatively simple to partition an image or other data structure over several parallel function blocks. This requires designing one function block, and replicating it. In the same way, functional parallelism can also be exploited – the only difference is the size and complexity of the function block. Additional logic is required to direct the appropriate pixels to their corresponding function blocks and managing any communication required between the blocks. If necessary, uneven processing rates may be smoothed through the use of FIFO buffers built from on-chip RAM.

Streaming can feed data serially through a series of function blocks. Again, streaming is natural for raster scanned image data. The complexity comes from ensuring that the data is processed at the pixel rate. This requires designing an appropriate architecture to ensure that all of the data is available when it is required. On-chip memory can be used to cache data until it is required. The abundance of registers facilitates the fine grained pipelining required to perform complex operations at the pixel clock rate. Operating a streamed pipeline is straight forward when in the middle of an image. However, pipelines require priming at the start of the image and at the start of each row. At the end of the row and image, it is necessary to keep the pipeline running until processing completes for the valid data remaining within the pipeline. The logic required for priming and flushing is often more complex than the normal processing logic.

Image processing, or at least the low-level pixel processing, is ideally suited to implementation on an FPGA. The operations are generally quite simple and inherently parallel. The reprogrammability of FPGAs allows the hardware implementing the algorithm to be redesigned and adapted if the nature of the machine vision task changes. FPGAs therefore give the best of both worlds: the flexibility and reprogrammability that normally comes with software, along with the speed and parallelism that comes from a hardware implementation.

25.2.3 Programming FPGAs

The configuration of an FPGA represents a hardware based logic circuit. Traditionally, electronic circuits are represented by schematic diagrams, although for large complex designs, these can quickly become unwieldy. In the 1980s, text based hardware description languages were developed to provide a standard representation of the specification and behaviour of a logic circuit. The two most commonly used languages are VHDL and Verilog, both of which are specified by IEEE standards. Since they must describe parallel systems, they are primarily concurrent languages. There are two main styles of programming with hardware description languages. The first is the structural style, which defines a design as a hierarchical series of functional blocks and their interconnections. This corresponds directly with the schematic type representation of a design. The alternative style is through a behavioural description of the operation of the circuit. This describes how the circuit or system should respond to changes on its inputs. This is therefore at a higher level of abstraction than a structural description, making it more appropriate for representing algorithms.

Programming FPGAs with VHDL or Verilog is a little like programming in assembler. The developer has full flexibility and control, enabling them to even explicitly use particular features of the target FPGA. The limitation however, is that the developer must also manage all of the fine detail and low-level control for implementing their algorithm. To overcome this, there has been significant research recently on synthesising hardware directly from higher level software based languages, mainly based on C. The goal of this is two-fold. First it makes the design more accessible to software engineers [3], enabling them to work in a familiar language. Second, the higher level of abstraction should make development more efficient by being closer to an algorithmic representation and removing much of the necessity of managing all of the low level details.

There are several problems with synthesising hardware directly from software languages [19]. First, software is primarily a sequential representation whereas hardware is naturally concurrent. It is often necessary to augment the language with new constructs for representing parallel execution. While some of this may be extracted automatically by the compiler through a dataflow dependency analysis, current compilers are limited in what they are able to extract. Modern compilers, usually in conjunction with operating system support, can implement concurrency with a threaded computation model, this is too coarse-grained for working with hardware. Strongly related to concurrent execution is the need for communication and synchronisation between different threads or different parts of the hardware. Semaphores, mutexes, channels, mailboxes, or other synchronisation mechanisms require a direct representation to simplify the design of the concurrent systems. A second aspect is that hardware implementation usually requires an explicit representation of time. This is particularly so for stream based processing where actions are synchronised with a data clock. In software, timing is only implicit in the order of operations. Third, in software, native data types are 8, 16, 32 and 64 bits wide. In hardware, efficient implementation requires tailoring the data width to the particular task. This requires the language to be able to explicitly specify the width of each variable and data path. Finally, software assumes a monolithic memory architecture, with single large memory used to store all data. In C, memory is commonly accessed via pointers, with a single address structure allowing pointer arithmetic to be used for efficient software implementation. In contrast to this, FPGAs have many small independent memory blocks which can be exploited to increase the bandwidth. Even external to the FPGA, the memory may consist of several independent parallel memory banks. Pointer arithmetic is not applicable to FPGAs.

One language we have found particularly useful for programming FPGAs is Handel-C. Originally developed in the Hardware Compilation Research Group at Oxford University [31], it was commercialised by Celoxica, and is now provided by Mentor Graphics [30]. Handel-C extends ANSI C with constructs that address the issues identified in the previous paragraph. The basic data type is an integer (floating point is not directly supported), with directly specified bit-width. A range of bit manipulation operations (extraction, concatenation) allow low-level data manipulation. In hardware, many operations are implemented simply by wiring the bits to the appropriate places. The normal sequential execution is augmented by a par{} construct, with all of the statements within the par block executed concurrently. Explicit timing is provided by each assignment statement taking exactly one clock cycle to execute. The complexity of the right hand side of an assignment determines the propagation delay, hence the maximum clock rate achievable. Concurrent constructs such as semaphores and channels are implemented as native constructs within the language.

While closely resembling standard C, it is important to remember that Handel-C is actually a hardware description language with each statement compiling to separate hardware. The compiler also builds an implicit finite state machine which controls the execution of each statement, and manages the sequential flow, including the appropriate control paths for looping and branching constructs. Structural design is supported through function calls, although because each function call connects to hardware, recursion is not supported (there is no programme counter, and no stack). While the language looks like C, it is still important for the developer to design the appropriate hardware and represent it using Handel-C. Simply writing C and expecting an efficient hardware implementation does not work! However provided that the FPGA architecture is kept in mind, the higher, algorithmic, level of representation does speed the development of efficient hardware.

Having chosen a hardware description language, and represented the design in the chosen language, there are several steps required to translate this into a form for operating the FPGA. The first step is synthesis, which translates the structural or behavioural description into a device or gate level net-list representing the hardware to be constructed on the FPGA. From a structural representation, this is a one-to-one mapping, although for a behavioural description the synthesis tools must determine the logic required to implement the specified behaviour. In VHDL or Verilog, this requires that the behavioural description conform to a register transfer level coding style. Synthesis tools can optimise a design for speed or area by selecting appropriate circuits for each task. The next step is mapping, which determines how the logic is implemented by the specific components and resources available on the FPGA. In particular, the logic is mapped onto the LUTs, merging related logic to fit within a single LUT, or partitioning complex logic over multiple LUTs. Once mapped, place and route tools place each component within specific logic blocks, and determine the particular routing resources required to connect between the components, and to external pins. Place and route is usually an iterative process, with components repositioned and rerouted to minimise connection lengths and meet timing constraints. The final stage is the generation of the configuration file that is loaded into the FPGA to implement the design.

25.3 Design Using FPGAs

So far, we have looked at FPGAs and how they can potentially accelerate machine vision algorithms by exploiting the parallelism inherent in processing images. In this section, the design process for an FPGA implementation will be compared with that for conventional software based design. A good software design does not necessarily correspond to a good hardware design for the following reasons [21]:

- The optimal processing mode in hardware is usually different from software [23]. Random memory access is efficient in software, especially with memory caching used by modern processors. On an FPGA, random access to image memory can be inefficient because it is usually stored off the FPGA. Software is often memory bound with intermediate variables stored within memory. On an FPGA, intermediate variables and data can be stored in dedicated registers or on-chip memory buffers, giving a much higher bandwidth. Stream based processing is better suited for manipulating images where possible.
- FPGA clock speeds are typically an order of magnitude slower than high performance serial processors. This is because the general purpose nature of FPGA logic, combined with the overhead of routing delays cannot compete with custom designed logic circuits. It is necessary to gain an order of magnitude acceleration through exploiting parallelism just to break even with serial processors. A parallel design is therefore essential rather than relying on a high clock speed.

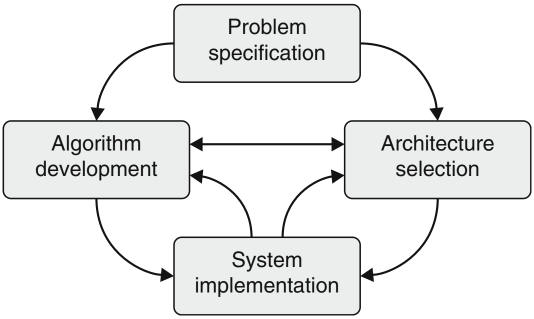


Fig. 25.5
Stages of FPGA based image processing design

- The concurrent nature of hardware complicates arbitration for resources. On a serial processor, this is significantly reduced because the processor can only do one thing at a time.
- The lack of an operating system on an FPGA system means that memory and other system resources must be managed or scheduled manually rather than via an operating system. It is also necessary to develop hardware drivers for any peripherals used by the application.

The design process needs to take these factors into account. There are four key stages of developing an FPGA based image processing system [21], as illustrated in *Fig. 25.5*.

25.3.1 Problem Specification

The problem specification stage is the same as for conventional software engineering. It is necessary to clearly define the image processing problem, identify assumptions that may be made about the images to be processed, and specify the set of functional and engineering constraints that must be met for a solution to be considered successful. For image processing, this requires clarifying and formalising the often vague description of the task to be performed by the image processing module. The resulting engineering specification must be sufficiently concise to enable any potential solution to be evaluated for correctness and completeness.

Some of the key issues to consider here are: the timing constraints; how knowledge of the task performed can be used to simplify the processing; the selection of representative sets of images for developing and testing the algorithms; the selection of appropriate features for performing any classifications; the specification of minimum success rates in terms of classifications; the robustness of the complete system to errors or misclassifications; and whether or not an FPGA based system is actually required. It is important to remember that image processing is only one module within the overall machine vision system [9] and success requires that it be integrated with the rest of the system.

25.3.2 Algorithm Development

Algorithm development requires determining the sequence of image processing operations that perform the task and meet the performance specifications. Algorithm development typically involves two phases. The first is to find an algorithm that meets the specifications.

This typically the same process as for a software based solution, and requires an interactive image processing algorithm development environment. This must be performed in a standard software based environment because an FPGA does not have the interactivity and a large range of image processing commands available for the development to take place directly on the FPGA system.

The second phase, once a suitable algorithm has been found, is to adapt it for parallel implementation. This often requires significant changes to the algorithm to exploit parallelism and to make best use of the resources available on the FPGA. It is at this stage that the design process diverges from the design for a conventional processor. The algorithm optimisation goals are different for software and hardware processors, although in both cases the design is usually iterative, with the algorithm adapted and refined until the processing goals and constraints are met.

25.3.3 Architecture Selection

On a software based system, the computational architecture is fixed. For an FPGA based machine vision system, very little is predefined. It is necessary to develop both the architecture as well as the algorithm. The concurrency of hardware opens a wide range of possibilities to tailor the architecture to the application.

The architecture needs to be designed at multiple levels. At the highest level is the system architecture. While low-level image to image operations are well suited to a hardware implementation, the higher level operations, such as recognition tasks, are harder to implement efficiently in custom hardware. Tasks which only run occasionally, or have complex data dependent flow, if implemented in hardware tend to require large amounts of hardware to account for the different situations, and leave much of the hardware idle for long periods of time. Such tasks also tend to have significantly lower data volumes, making them better suited to a conventional serial processor.

In many machine vision applications, such as inspection and grading, the low-level operations tend to dominate, and the high-level vision tasks are relatively simple. Such applications can be implemented solely on an FPGA, operating in standalone mode. In this architecture, everything is implemented on the FPGA, although it may be connected to a host computer during the initial development and for debugging. In the standalone configuration, the FPGA is responsible for everything, including image capture, display, and any interaction with the user.

Applications with more complex high-level processing may require partitioning the computation between hardware and software processors. Two possibilities for implementing the composite hardware/software system are to have a separate computer subsystem, or to implement the serial processor from the logic of the FPGA. In the latter case, both Xilinx and Altera provide processor cores (MicroBlaze and NIOS II respectively) and associated development environments.

The system architecture also includes how peripheral devices are connected to the system. The goal of machine vision is to autonomously control some activity, so the controllers must also be integrated into the system.

It is also necessary to design the computational architecture. The computational architecture for a serial processor consists of an arithmetic logic unit (ALU) and a set of general purpose registers. A series of machine instructions fetch data from registers or memory, determine the operation performed by the ALU, and store the results back into registers or memory. Many of the components are reused from one instruction to the next, and the data is passed through the ALU many times in the course of the processing.

In contrast with this, in a hardware system the computational architecture is designed to accelerate the processing by exploiting parallelism. Consequently, the processing tends to be more distributed, with the data passing through each processing block only once in the course of the algorithm. Each of the parallel processing modes described earlier can be used as the basis for designing a computational architecture.

Stream processing processes an image as a sequential stream of pixels. It is well suited to processing image data on the fly as it comes from a camera or is output to a display. It works with a fixed clock rate, usually one pixel per clock cycle, which constrains memory bandwidth. Stream processing is well suited to low-level image processing operations where only local information is required for each output pixel. Other operations may require significant algorithm redesign to fit within a stream processing framework. Algorithm acceleration is gained through pipelining, with low-level pipelining used to meet timing constraints. Stream processing may be extended through the use of systolic arrays, where a one or two-dimensional array of processors operates on data as it is streamed through. Systolic arrays effectively unroll a recursive processing loop over a set of parallel concurrent processors and use simple communication between the processors to maintain state as the data is passed through with each clock cycle.

Random access processing allows pixels to be accessed anywhere from a frame buffer as needed by an operation. The lack of explicit data access requirements make it much closer to the software approach to processing, although simply porting a software algorithm to hardware will usually be memory bound, with memory bandwidth limiting the performance. While random access processing relaxes the hard timing constraint of stream processing, acceleration can be gained by maximising the clock rate by reducing the combinatorial delay through low level pipelining. It is also necessary to identify parallel branches to distribute the processing effectively. Image partitioning can be used here where the algorithm allows. Memory bandwidth limitations are then overcome by associating local memory with each branch and reducing the number of accesses required to the global frame buffer. In the extreme case, a massively parallel architecture uses a simple processor for each pixel in the image. While fine in theory, such massively parallel architectures are resource hungry, and require large expensive FPGAs to be effective.

It is important to obtain a good match between the algorithm and the computational architecture. A poor match limits the acceleration available, and makes inefficient use of available resources. Therefore several iterations between algorithm development and architecture selection are often required before a good match is found.

25.3.4 System Implementation

The final stage is system implementation. This is where the algorithm is mapped onto the architecture. This task differs significantly from software based design. While the initial algorithm is developed as software, the mapping process does not simply port the software algorithm to hardware unless it was initially developed with a hardware implementation in mind. The function of the software is mapped, rather than the specific algorithm.

Image processing operations are often expressed in parallel, for example a filter applies an operation on the pixel values within a local neighbourhood or window in an image. This is implicitly parallel, since the operation is applied for every possible position of the window.

A software implementation represents this as a serial algorithm optimised for such a serial computational architecture. It is not always an easy task to convert or map such serial algorithms back onto parallel architectures. Therefore, it is necessary to remap the operation rather than simply translate the algorithm. Consequently, not all software algorithms map well to hardware. Implementation often requires significant changes to the actual algorithm, and sometimes to the computational architecture to make full use of parallelism and obtain an efficient implementation.

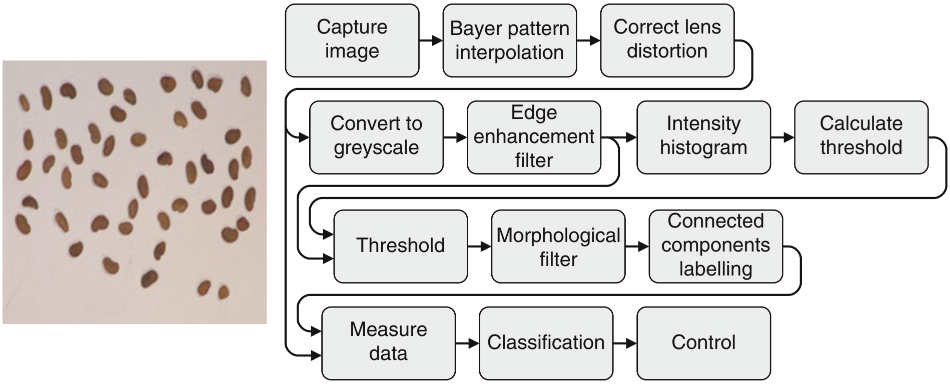
Another implementation task is to develop device drivers and interfaces for any peripherals. These are usually provided by an operating system in software based systems, but are not generally available in hardware. They therefore must be developed by the system designer.

It is important to remember that an FPGA based implementation requires hardware design. Even though the implementation language may look like software, an efficient design requires a hardware mindset.

25.4 Transforming an Algorithm for Hardware Implementation

The design process outlined in the previous section will be illustrated with an example application (see ➲ *Fig. 25.6*). This example is somewhat contrived to show how a range of different image processing operations are mapped onto an FPGA, and to demonstrate the mapping process from software to hardware. The application consists of grading seeds on the basis of colour and size. It will also be assumed that the latency of the algorithm must be minimised, hence the need to use an FPGA implementation.

A colour image is captured using a single-chip CMOS colour sensor using a Bayer colour filter array. From the camera, each pixel will either be red, green or blue depending on the filter associated with that pixel. To obtain a full colour image, it is necessary to interpolate between the pixels available to obtain the missing components. Since the control output needs to be able to precisely locate each seed, it is necessary to calibrate and correct for any geometric distortion introduced by the lens. Segmenting the seeds from the background does not require full colour, so the processing can be simplified and reduced by converting the colour image to greyscale. The edge enhancement filter sharpens edges between the seeds and the background to enable



◘ Fig. 25.6
Example machine vision algorithm, with a sample input image

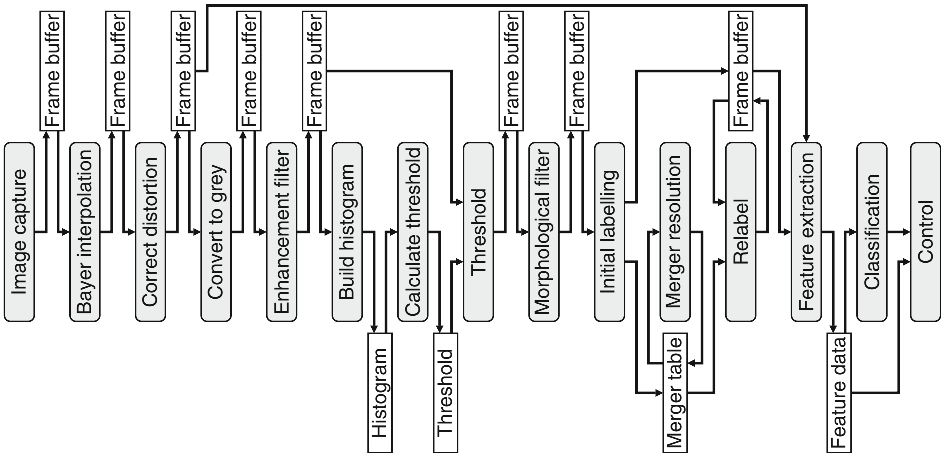


Fig. 25.7
Dataflow of software algorithm

more accurate segmentation. A simple global threshold is used to convert the image to binary. The threshold level is determined by analysing the intensity histogram of the image. After thresholding, a binary morphological filter is used to remove isolated misclassified pixels. Connected components labelling is then used to assign each binary blob (corresponding to a seed) a unique label. Then for each seed the centre of gravity, the area, and the average colour are measured. The area and colour are used to assign each seed into one of several classes, with the class and seed position passed to the control system to control an actuator that sorts the seeds on the basis of class.

The data flow through the algorithm is shown in ➲ *Fig. 25.7*. In this the connected components labelling operation is expanded out, using a standard two-pass labelling algorithm [33]. In the first pass, as the image is scanned, the previously processed neighbourhood of a pixel is examined. If none of the neighbouring pixels are labelled, a new label is assigned, otherwise the existing label is propagated. At the bottom of "U" shaped objects, two differently labelled regions merge, so the equivalence of those labels is stored in a merger table. At the end of the frame, the sets of equivalent labels are resolved, creating a mapping table from the initial label to the unique final labels. A second pass through the image uses the mapping as a lookup table to produce the output image. In software, this second pass reuses the same frame buffer for input and output.

The data flow clearly shows that a simple port of the software algorithm to hardware would not be effective. Eight separate frame buffers (or image arrays) are used, as are several other memory based data structures. In a software system, all of these data structures (along with other control variables) are maintained within a single global memory. The fact that every block is accessing memory means that the memory bandwidth will limit the extent to which the algorithm can be parallelised. Through the memory accesses, it is constrained to be a serial algorithm. To overcome the memory bandwidth problem, one simple (but not particularly practical) solution would be to use a separate memory block for each major data structure. An efficient parallel implementation requires that the algorithm be transformed, rather than simply ported [12].

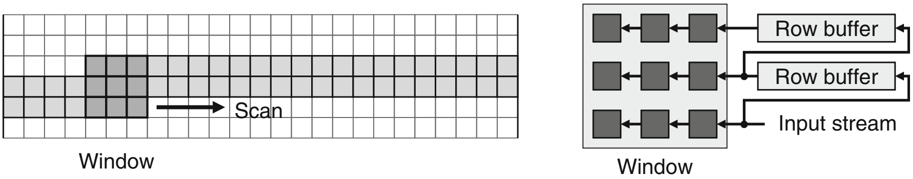
25.4.1 Pipelined Stream Processing

The main problem with the software approach is that it is memory bound. Each operation reads its input from memory, processes it, and writes the results back to memory. Stream processing can overcome this bottleneck by pipelining the operations. Rather than write the results to memory, they are passed directly on to the next operation in the pipeline, eliminating at least two memory accesses per pipeline stage. Stream processing requires that a downstream operation consumes data at the same rate and in the same order that data is produced by the operation immediately upstream. Since the outer loop of many image processing operations performs a raster scan through the image, this will be the case for those operations. Pipelined stream processing is effectively merging the outer loops of the pipelined operations, replacing a series of separate raster scans with a single raster scan but pipelining several operations within that single loop.

Point operations, where the output depends only on the corresponding input pixel, can easily be pipelined. This includes the conversion from colour to greyscale, thresholding, and the relabeling pass of connected components analysis.

Filters, which use only local information, can also be streamed since both the input and output follow the same raster scanned access pattern. However since each output pixel requires more than one pixel on the input, an operation specific cache is required to hold the input data until it is needed. Consider a 3 × 3 window filter – each output pixel value is a function of the nine pixel values within the window. In software, the input pixel values are accessed though a pair of nested loops. These loops can be unrolled and the required pixels accessed in parallel though the use of an appropriate cache. Without caching, each pixel must be read nine times as the window is scanned through the image. Pixels adjacent horizontally are required in successive clock cycles, so may simply be buffered and delayed in registers. This reduces the number of reads to three pixels every clock cycle. A row buffer caches the pixel values of previous rows to avoid having to read these pixel values from external memory again (see ➔ *Fig. 25.8*). A 3 × 3 filter spans three rows: the current row and two previous rows; a new pixel is read in on the current row, so two row buffers are required to cache the pixel values of the previous two rows. The row buffers are typically implemented using either FIFOs or circular memory buffers made from the dual-port RAM resources on the FPGA. This principle can be readily extended to larger window sizes or to other operations that require data from the previous rows of either the input or the output.

In the seed sorting algorithm, the edge enhancement and morphological filters can readily be placed in a streamed pipeline. The Bayer pattern interpolation is also a filter, although the actual filter function depends on location (which colour channels are missing); it may also be streamed. The initial labelling pass of connected components labelling propagates the output labels from the previous row. These may be cached using a row buffer, enabling this operation to be streamed as well.



■ **Fig. 25.8**
Row buffering caches the previous rows so they only need to be read once

Geometric transformations (such as that required for lens distortion correction) are a little more complex to pipeline. There are two approaches to implementing a geometric transform. In software, the usual approach is to use a reverse mapping. This determines for each output pixel where the input comes from in the image. It is therefore able to produce a streamed output, but the input requires random access, so cannot readily be streamed. The alternative is to use a forward mapping, which specifies where each input pixel maps to in the output image. This is able to operate with a streamed input, but the output does not generally follow a raster pattern. Therefore a geometric transformation will require a frame buffer on either the input or the output depending on the transformation approach used. In this application, since the output must be accessed later to measure the average colour, first analysis suggests that the forward mapping approach with a frame buffer on the output is the most appropriate.

Accumulating the histogram can be streamed, although the threshold level cannot be derived from the histogram until the complete frame has been accumulated. In many applications the histogram, and the threshold level calculated from it, will not change significantly from one frame to the next [29]. In this case, the threshold level determined from the previous frame can be used for the current image. This will reduce the latency of the algorithm by one frame, and enable the thresholding to be streamed directly from the output of the edge enhancement filter.

The input from the camera is naturally streamed, so can feed directly into the Bayer pattern interpolation without needing a frame buffer. The algorithm structure after transforming to a streamed architecture is shown in ➲ *Fig. 25.9*. It significantly reduces the number of frame buffers needed by eliminating buffers between pipelined stages. The buffer within the connected components analysis is required because the merger resolution step requires data from the complete frame. Pipelining also requires that a separate output buffer be used for relabeling the mergers. This is because the image cannot be relabelled in place because the first frame buffer is required for storing the initial labelling of the next image. Data from the lens distortion correction may actually require multiple buffers to prevent the colour pixel data from being overwritten before it is used by the feature extraction block.

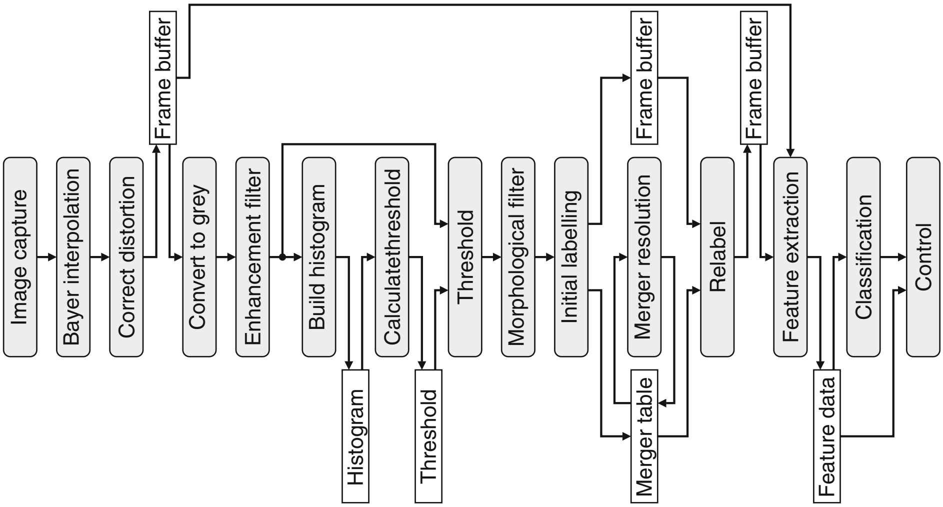


Fig. 25.9
After transformation to use a pipelined streamed architecture

The other effect of converting to a streamed pipeline is that everything becomes synchronised with the pixel clock rather than simply running as fast as possible. The reduction in clock frequency is made possible by eliminating the many unnecessary memory accesses, and exploiting parallelism through pipelining. It also has the potential to reduce the latency, although in this case it is not necessarily realised because of the remaining frame buffers. While every image is processed at the pixel clock rate, the latency is approximately three frame periods.

25.4.2 Strip Mining and Strip Rolling

The feature extraction step typically iterates through the labelled image extracting data for each label in turn. This iteration is a bottleneck within the connected components dataflow and would require an unrealistic clock rate to process all of the labels within a single frame period. However, since each iteration is processing a separate label, this step is readily parallelisable by having multiple feature extraction units. Unrolling and parallelising such loops is called strip mining. In this case, though, it creates another problem: all of the feature extraction processors would be competing for access to the single shared frame buffer. Using a streamed access would allow the frame buffer to be shared among all of the feature extraction processors. Many (but not all) useful object features can be accumulated using a raster scan through the image. This includes the centre of gravity, area, and average colour required in this application.

In this instance, however, strip-mining only partially solves the problem. Potentially there can be a large number of components within the image, requiring a large number of parallel processors to handle all of them in parallel. Images with only a few labels would leave most of the hardware sitting idle, resulting in inefficient hardware utilisation. Conversely, if only a few processors are constructed, then images with a large number of labels would require multiple passes to process all of the objects. This would require either a higher clock rate for the feature extractors, or a lower (and possibly variable) frame rate.

A second look at this problem results in the observation that each pixel can only have a single label. This implies that if stream processing is used, a single feature extraction processor may be shared for all of the iterations (strip rolling). Separate data registers must be maintained for each component with the particular registers used for processing each pixel multiplexed by the component label. Thus for each pixel, the label is used to select the data associated with the corresponding object. This data is then updated according to the pixel added, and saved again in the corresponding register. In this way, a single feature extraction processor is able to measure the features of an arbitrary number of objects in a single pass through the image.

The data register multiplexing may be readily implemented using a memory-based data table. The multiplexing hardware then comes for free, being implemented through the built in memory addressing logic. This transformation also makes the frame buffer between the relabelling and feature extraction modules redundant. Since both use stream processing, the final labels may be streamed directly into the feature extraction module, reducing both the memory requirements and latency.

25.4.3 Algorithm Optimisation

A number of other optimisations are possible. For example, the data volume and processing may be reduced by using appropriate data coding. In this application, run-length encoding may be used with connected components labelling [7] to reduce the size of the associated frame buffer. The second relabeling pass can then process whole runs of pixels as a unit, significantly reducing the latency. In this application, since the colour associated with each object pixel must be accessed, the object pixels must be processed one at a time. However, each run of background pixels can be can be processed as a unit, allowing the background to be quickly skipped. This can result in a significant reduction in the latency of the second pass as the image is typically 80–85% background for this task.

Another optimisation is to rearrange the order of the operations to simplify the processing complexity, or even eliminate some steps. A classic example of this is greyscale morphological filtering followed by thresholding. This sequence of operations is equivalent to thresholding first, followed by binary morphological filtering which uses considerably less logic.

With connected components analysis, the feature extraction step can be applied to the data during the initial labelling pass [13]. This has two significant effects on the algorithm. First, relabeling the pixel data is no longer required and the frame buffer storing the initial labelling can be eliminated. Second, the feature data table may now contain multiple entries for each connected component, corresponding to the multiple labels that are assigned during the initial labelling. The algorithm for merger resolution must now be modified to merge the corresponding separate entries within the feature data table.

A further algorithm rearrangement is to merge the feature data when the mergers are first identified. The reason merger resolution was deferred in the original algorithm was because a merger could result in relabeling a large number of pixels. However, now that the algorithm is working with regions rather than pixels, it is practical to reintegrate the merger resolution step with the initial labelling pass. Whenever two regions merge, the corresponding entries in the data table can also be merged [10].

Elimination of the frame buffers associated with connected components labelling implies that the image after distortion correction does not necessarily need to be buffered. The frame buffer may be replaced by a synchronising buffer to account for the latency between the distortion correction and feature extraction modules. Since all of the intervening operations are streamed, the delay through the operations is a constant number of clock cycles. A FIFO buffer may be used to delay the colour pixel value so that the corresponding colour value is available for the feature extraction step.

Revisiting the distortion correction step, the simpler reverse mapping may be used to give a streamed output. With lens distortion, the input will follow approximately a raster scan, implying that only a few rows rather than the whole image needs to be buffered on the input [2]. Alternatively, rather than apply distortion correction to each pixel, the distorted image can be processed, with the correction applied to the features extracted. The centre of gravity will need to be adjusted to correct for the distortion, and the area scaled to account for the fact that the magnification is not uniform throughout the image.

The result of these optimisations is shown in ➋ *Fig. 25.10*. The algorithm has been transformed so that all of the steps share a common processing mode, and are architecturally compatible. All of the frame buffers have been eliminated, reducing the latency to approximately one frame period.

25.5 Mapping of the Image Processing Operations

The previous section developed the system architecture for an example application. This section will explore the computational architecture of each operation in the resultant

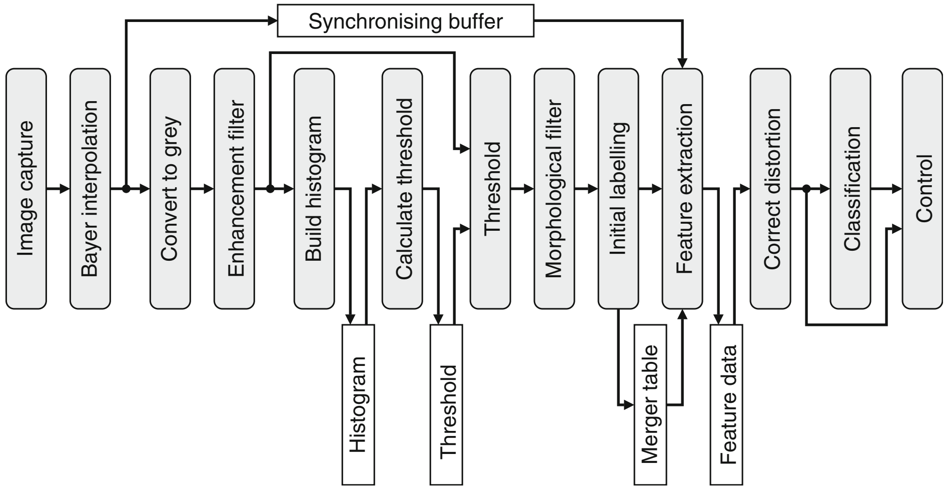


Fig. 25.10
Final stream based algorithm architecture

algorithm. The original algorithm was chosen so that the operations used are representative of the range of operations typically used in machine vision applications. Many of the points discussed here generalise to other similar operations.

25.5.1 Point Operations

Point operations are the simplest of image processing operations to implement, since the output value depends only on the corresponding input pixel value. Simple operations may be implemented directly, while for more complex operations, the mapping from input pixel value to output pixel value may be represented using a lookup table. For 8 bit images, such a lookup table requires only 256 entries and may be implemented using a single memory block within the FPGA.

The thresholding operation in the seed grading algorithm is dynamic, in the sense that the threshold level may vary from frame to frame. The simplest implementation is to subtract the threshold level from the input pixel values as they are streamed through. The sign bit of the result indicates whether the input is greater than or equal to the threshold (sign bit equals 0) or less than the threshold (sign bit equals 1). Since the background is lighter than the seeds, the sign of the difference may be used directly to indicate the presence of absence of a seed.

There are many ways to convert from a colour image to greyscale. In this application, the particular colour space is not important, so the simplest conversion is to use weights which are powers of two, since the multiplications may be implemented by bit shifting which is free in hardware:

$$\text{Greyscale} = \frac{1}{4}\ \text{Red} + \frac{1}{2}\ \text{Green} + \frac{1}{4}\ \text{Blue} \qquad (25.1)$$

Higher weight is given to green because with the Bayer pattern input, there are twice as many green pixels as red or blue.

25.5.2 Local Filters

Implementing a filter requires row buffers to cache the input stream for later use when calculating the output on subsequent rows. *Figure 25.11* shows the linear edge enhancement filter. The filter weights can be chosen to be powers of two to avoid the need for multiplications. In a direct implementation of a 3 × 3 linear filter, eight operations are required to add the nine pixel values within the window together. However, symmetries within the window allow the computation to be factorised, reducing it to five operations, as shown in *Fig. 25.11*. If the filter is separable, this generally provides the greatest opportunities for factorising the calculation.

The morphological filters can be implemented in a similar manner to linear filters, except using a logical AND of pixel values within the window for erosion, and a logical OR for dilation. Again, gains can be made if the window can be decomposed or is separable. Even if the window is not separable in the conventional sense, it can be made separable using the SKIPSM paradigm [36, 37]. Another alternative for small windows is to use a lookup table to evaluate the function. The bits within the window are simply concatenated and looked up to produce the corresponding output bit.

In this application, scattered noise pixels may be removed by using a 3 × 3 opening to remove false object pixels, followed by a 3 × 3 closing to remove small holes. The opening is performed first, to reduce the chance of objects becoming linked by noise in the background. The concatenation requires a series of three filters: a 3 × 3 erosion, followed by a 5 × 5 dilation, followed by a 3 × 3 erosion. All three filters are directly separable, simplifying their calculation.

The third filter in this application is to interpolate the missing colour components from the Bayer colour filter array. Each incoming pixel will either be red, green, or blue depending on the particular colour filter associated with that pixel (see *Fig. 25.12*). There is a wide range of algorithms for interpolating the missing colour values. Nearest neighbour and bilinear interpolation, while relatively simple and low cost, are prone to artefacts particularly around edges and in regions of fine detail. Many of the more complex algorithms are designed to reduce these artefacts, although this comes at the cost of significantly increased computational complexity, and the number of row buffers required can be quite large. One compromise described in [24] uses a 5 × 3 window requiring only two row buffers. The improvement in quality is obtained by detecting edge directionality and interpolating along, rather than across, an edge. A local gain term is also introduced to enhance fine details. The computational requirements are modest, requiring only 3 subtractions, 10 additions, 2 divisions, and 3 multiplications.

The difference between the interpolation filter and conventional filters is that the filter function varies from pixel to pixel, depending on the colour and context of the incoming pixel. This may be managed by a simple state machine.

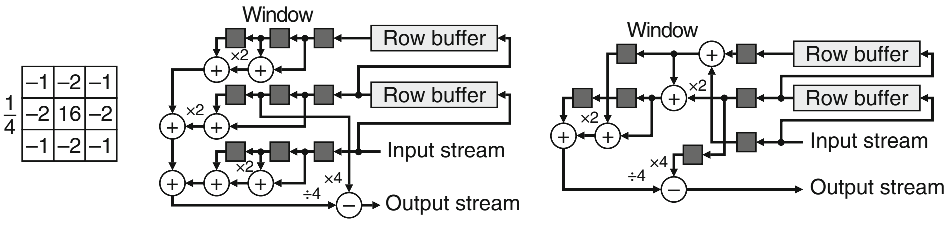


Fig. 25.11
Linear edge enhancement filter: filter weights, direct and factorised implementations

B	G	B	G	B
G	R	G	R	G
B	G	B	G	B
G	R	G	R	G
B	G	B	G	B

Fig. 25.12
Arrangement of pixels with a Bayer colour filter array

One factor that has not been discussed is how to handle the edges of the image where the window extends past the edges. Additional logic is required to extend the image and correctly manage these situations. Further implementation details on managing the edge effects and related pipeline priming and flushing implications are beyond the scope of this brief overview, but are discussed in some detail in [8].

25.5.3 Histograms

Accumulating a histogram requires maintaining a series of counters, one for each possible pixel value, and incrementing the appropriate counter as the pixels are streamed in. With a set of parallel counters, the decoding logic can be quite resource intensive. Since only one counter is incremented in any clock cycle, the parallel counters may be strip-rolled into a memory structure. Each clock cycle, the memory entry corresponding to the incoming pixel value is read and incremented. On the following clock cycle, the new count is written back to memory. A dual-port on-chip memory allows the two simultaneous accesses. A problem arises when two successive incoming pixels have the same value. For the second pixel, the count read from memory will be the old count because it has not yet been updated. Therefore it is necessary to test for this condition, and use the previous count rather than that read from memory. This process is illustrated in *Fig. 25.13*.

The next step is to calculate the threshold level from the histogram. While there are many possible approaches that could be taken, in this case since the contrast is good, the simplest approach is to use the mean of means [32]. This method refines an existing threshold as the midpoint between the mean of the dark pixels (less than the threshold) and the mean of the light pixels (greater than or equal to the threshold):

$$\text{Threshold}_{n+1} = \frac{\mu_{I<\text{Threshold}_n} + \mu_{I\geq\text{Threshold}_n}}{2} \tag{25.2}$$

This calculation requires a counter to step through the histogram, and circuitry to accumulate the count and weighted count from which the two means are calculated. Equation 25.2 is normally iterated until it converges (typically three or four iterations). However, in this application, there should only be a small change (if any) from one image to the next. Therefore, a single iteration suffices. During the iteration to calculate the threshold level, the write port may be used to reset the histogram to zero in preparation for the next image.

Since only a single pass through the histogram is required, it is actually simpler to directly accumulate the pixel values and count above and below the current threshold level. The threshold block from the processing algorithm is reused to determine which set of accumulators is used, and the intermediate histogram is not actually required.

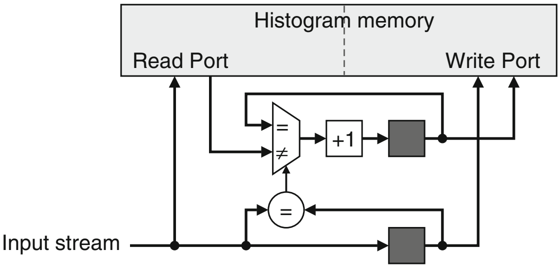


Fig. 25.13
Using dual-port memory for accumulating the histogram

25.5.4 Geometric Transformation

There are two main approaches to performing a geometric transformation. The forward mapping determines where each input pixel goes to in the output image, whereas the reverse mapping determines where each output pixel comes from in the input.

A common problem with the forward mapping is holes in the output, which result from output pixels that have no corresponding input pixels mapping to them [38]. This may be overcome by separating the transformation into two passes [16]. The first pass operates on each row, and warps the pixels to get them into the correct column. The second pass then operates on each column to get the pixels into the correct row. The advantage of the separable transformation is that the resulting one-dimensional transformations are much simpler than directly performing a two-dimensional transformation. Example hardware for performing each one dimensional transform is described in [39]. Rather than performing the two passes through the image separately, they can be pipelined [11]. In the second pass, all of the columns can be processed in parallel, with a memory based data buffer to hold the data for all of the columns. Then, when a pixel is output from the first pass, it is passed to the appropriate column in the second pass. Since the first pass scans along a row, the outputs will be for successive columns. However, as a result of the warping, the outputs from the column processing may not necessarily be successive memory locations, requiring a random access frame buffer on the output.

The alternative is to use a reverse mapping. This produces pixels in streamed order, but requires random access on the input. As a result of the mapping, the input may not fall exactly on an input pixel location, so interpolation may be used to give more accurate results. With interpolation, the integer part of the pixel location selects the nearest pixels, which are then weighted based on the fractional part of the location. This is much the same as image filtering, with the added complication that the path through the input image is curved. Since only one pixel can be read from the input frame buffer in each clock cycle, caching pixels that are likely to be reused becomes more complex (see for example [20]) because the path through the image does not, in general, follow a raster scan. If the input pixel values are stored in on-chip memory, this bandwidth limitation may be alleviated by using separate blocks of memory for both the odd and even row and column addresses. This allows four pixels to be read simultaneously for bilinear interpolation.

One optimisation that may be used with either streamed input or output is to perform incremental calculations. Rather than calculate the mapping independently for each pixel, in many cases it is possible to reuse the part of the result from the previous pixel with an incremental adjustment. In the application considered here, one model for lens distortion correction is [22]

$$r_u = r_d\left(1 + \kappa r_d^2\right) \tag{25.3}$$

where r_u and r_d are the radius from the centre of distortion in the undistorted and distorted images respectively, and κ is the distortion parameter. ➋ Equation 25.3 may be used directly with a forward mapping, or the corresponding equation for the reverse mapping is

$$r_d = r_u M\left(\kappa r_u^2\right) \tag{25.4}$$

where $M()$ is a magnification function determined through a lookup table. In either case, calculation of $r^2 = x^2 + y^2$ may be simplified through incremental calculation [22]. Along any row, y^2 is constant, and x^2 may be calculated from

$$(x+1)^2 = x^2 + 2x + 1 \tag{25.5}$$

A similar calculation may be used to update y^2 when moving from one row to the next. Another optimisation is that the size of the lookup table for determining $M()$ may be significantly reduced through using interpolation [22] (requiring a single multiplication), or bipartite tables [35] (requiring only an addition).

In the implementation of the algorithm shown in ➋ *Fig. 25.10*, the distortion correction is deferred until after the features have been measured. Correction at this stage requires calculating the both the position and area magnification functions for each object using the forward mapping. This position magnification $\left(1 + \kappa r_d^2\right)$ is used to scale the position, whereas the area magnification $\left(1 + \kappa r_d^2\right)\left(1 + 3\kappa r_d^2\right)$ is used to scale the area.

25.5.5 Connected Components Analysis

The architecture of the single-pass connected components algorithm is shown in ➋ *Fig. 25.14* [13, 26]. The label selection block determines the label for the current pixel based on the labels of its neighbours, with the labels on the previous row cached using a row buffer. Label selection follows the same approach as the classic two-pass algorithm for propagating the label. When two regions merge, the smaller of the labels is propagated. The merger control block records the equivalence by updating the merger table so that the larger label points to the smaller. Whenever a new label is created, the merger table is initialised by adding a new entry that points to itself. The role of the merger table is to ensure that only the current label for a component is used.

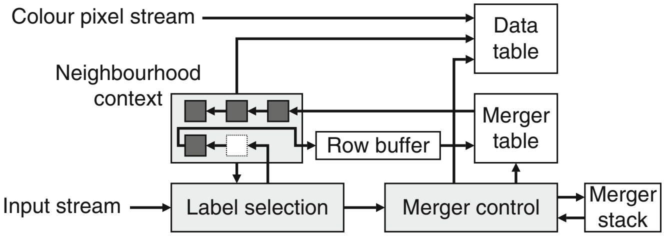


Fig. 25.14
Single pass connected components analysis computational architecture

Any old merged labels cached in the row buffer are translated to their equivalent label before reloading into the neighbourhood context.

When there is a series of mergers with the smaller label on the right, each merger requires updating several equivalent labels. This cannot be completed in the single clock cycle required. These chains may be resolved with a reverse scan of the row during the horizontal blanking period between rows. To accomplish this, pairs of equivalent labels where the smaller label is on the right are pushed onto the merger stack and at the end of the row they are popped off in reverse order for updating the merger table [13].

As the image is scanned, the data table accumulates the raw data for calculating the features of each connected component. The data table is indexed by the label of the current pixel, with the entry updated to include the current pixel within the region. Whenever two regions merge, the corresponding entries in the data table are also merged.

In the seed grading algorithm, six accumulators are maintained for each region: pixel count (for the area), sum of x and sum of y (for the centre of gravity), and the sum of each of the red, green and blue components (for the average colour). When two regions merge, the sums for each of the regions are added together to give the sum for the new composite region.

The size of both the merger and data tables depend on the number of labels that are allowed. To handle the worst case (one that will correctly process every possible image), the number of labels is one quarter of the number of pixels within the image [14]. (In this application, the use of the morphological filter reduces the worst case to 1/36 of the number of pixels in the image, but this is still proportional to the area of the image.) For typical images, such as those that will be encountered in this application, the number of labels needed is much smaller. A pragmatist would argue that as long as it handles the number required then that is sufficient, whereas the purist would require that it handle anything that could be given to it.

An observation is that at any time, the maximum number of labels in use depends only on the width of the image. This means that it should be possible to reuse the labels of completed objects. The next optimisation is to recycle labels from one row to the next [14, 28]. Since a labelled image is not required, the labels on each row can be allocated almost independently of the labels on previous rows as long as the labels are consistent with the connectivity as determined to that point. Therefore, assuming that the labelling is consistent for the previous row, it is only necessary to process the objects sufficiently to obtain a consistent labelling at the end of the current row. As each pixel is only visited once, it is necessary to have data structures to record merging of regions on the current row. The simplest label allocation scheme is to allocate labels sequentially on each new row. This requires that labels assigned on the previous row be translated to those used on the current row. The resulting optimised architecture is shown in ➲ *Fig. 25.15*. Its operation will be outlined here, with more details available in [28].

The label equivalence table is now split into two tables, with the merger table reflecting equivalences between labels on the previous row, and a translation table to update labels from the previous row to the current row. Two merger tables are required: one to look up mergers that have taken place in the previous row, and one to record new equivalences in the current row. At the end of each row, the table for the current row becomes the lookup table for the previous row, and the old table is reset for the new row. The translation table is built as the row is scanned, and is discarded at the end of the row.

Label selection logic and merger control are now slightly more complex than before. A new label must now be assigned each time a region is first encountered on a row. Old row labels on the previous row in the neighbourhood are updated with the new label, and the translation recorded in the translation table. If there happen to be two different previous row labels, then

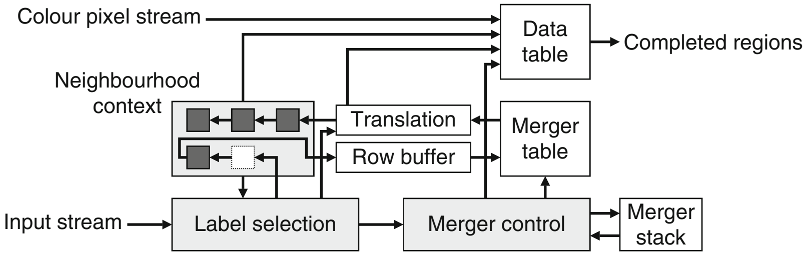


Fig. 25.15
Optimised computational architecture for connected components analysis

this merger is reflected in the state of the translation table. A merger is only recorded in the merger table when two current row labels in the neighbourhood merge. Otherwise the merger control block operates in much the same way as before.

Since the labels are changed with each new row, the data table must be replaced as each row is processed. Two data tables are required: one indexed by the current row labels, and one indexed by the previous row labels. Whenever a new translation is entered into the translation table, the feature data is copied to the current table and deleted from the previous table. Mergers entered into the merger table merge the data in the current table as before. At the end of the row, any entries remaining in the previous feature data table correspond to completed regions. The region feature data may be passed out at this stage rather than waiting until the end of the image.

Although two merger tables and two data tables are now required, these are both much smaller than before, being proportional to the width of the image rather than the area. This can significantly reduce the size of the memory required, even for a system that handles only the typical rather than worst case [14]. Another advantage is that the region data is available one row after the region is completed, rather than having to wait until the complete frame has been processed. This can significantly reduce the latency, especially for objects near the top of the image.

25.6 Interfacing

In any machine vision system, it is necessary to have input from a camera or other sensor. Since the primary purpose of a machine vision system is to autonomously perform an activity, it is usually necessary to control some form of actuator. It is often also necessary to have some form of display output, even if only to demonstrate to an operator that the system is working. In many applications, a user friendly display is important. Depending on the context, it may also be necessary to accept control input from a user, and to interface with a software processor. This section will briefly outline some of the possibilities and issues with interfacing between a vision algorithm on an FPGA and external devices.

Connecting to an analogue video camera requires appropriate decoding of a composite video signal, and high speed A/D converters for each colour channel. While this is simpler for a monochrome camera, the easiest and tidiest solution is to use an appropriate codec chip. This performs all the necessary filtering and conversion of the analogue signal to digital, and directly provides a pixel clock and the end of line and end of frame timing signals. Configuration of such codec chips is typically through some form of low speed serial interface. The communication protocol of such interfaces is usually sufficiently simple that a small control block can be built with FPGA logic to set up and control the device.

More modern commodity cameras have a USB or Firewire interface. Although they are digital, they are actually harder to interface to because they are based on a more complex communication protocol. The USB protocol requires a central controller which makes USB cameras quite difficult to interface to FPGAs. Firewire is a peer-to-peer protocol, reducing the complexity of the interface. Rather than interface directly, it is advantageous to use an external physical layer controller to manage the physical communication, timing, and arbitration logic. Similarly, using an external link layer controller simplifies the interface by correctly forming the packets and providing a higher level interface, including an integrated FIFO buffer. Integrated physical layer and link layer devices are also available that can be connected through a PCI express endpoint on high-end FPGAs.

An alternative is to interface directly with the CMOS imaging sensor if integrating the FPGA into the camera itself. Such cameras typically provide the pixel data using a parallel digital output which can be easily interfaced directly to the FPGA. If using an external camera, an industrial camera with a CameraLink interface is another way of connecting the digital video signal to the FPGA. The CameraLink standard [1] uses 28 bits to represent 24 bits of pixel data and 4 bits for synchronisation signals. The data is serialised over five LVDS pairs, four of which are used for data, and one for the clock. The signals may be interfaced directly with the LVDS I/Os available on FPGAs, and the communications protocol is sufficiently straightforward to require minimal control and decoding logic.

While most machine vision systems do not need a display for normal operation, having a video display is essential for algorithm debugging and tuning, and setting up the system. A driver for a VGA type display requires two counters: one for the counting the pixels across the row, and one for counting the rows in a frame. From these, the synchronisation pulses and blanking signals may be generated at the appropriate times. Producing an analogue signal for display may be achieved simply with a resistor divider network for low resolution displays. Higher resolution displays would either require high speed D/A converters (available as codecs integrating the blanking and synchronisation signals) or use of the digital visual interface standard [17]. This encodes each of the red, green and blue channels using a 10-bit transition minimised code, with the signals sent serially over four differential pairs (one for each colour and one for the clock). Signals for lower resolution displays may be produced using the general purpose I/O pins of the FPGA, although high resolution modes may require the built in serialisation of high end FPGAs to achieve the required bit rates.

For simple displays, it is possible to synchronise the display with the processing data stream, allowing images to be displayed on the fly without the need for a frame buffer. More complex display layouts, for example showing several images, will require an external memory to buffer the display contents. Using a frame buffer has the advantage that the display contents can be frozen to enable debugging. Additional logic is also required to direct the appropriate image data to the corresponding location in the frame buffer. Arbitration logic and FIFO buffers may be required to manage conflicts between writing to the frame buffer and reading data from the buffer for display. Any text written to the display will require character generation logic either when writing to the frame buffer, or as an overlay when producing signals for the display.

The logic required to control an actuator obviously depends on the type of actuator controlled. This may vary from sending high-level data to an external controller (for example using a serial port) through to implementing the controller logic on the FPGA, and providing low-level actuation signals directly to the actuator. In systems with an associated software processor or microcontroller (either separate or embedded within the FPGA) this may be used for implementing more complex control algorithms.

For debugging and tuning, some form of control input from a user is usually required. Interfacing with a PS/2 keyboard or mouse is straight forward because the associated serial protocol is relatively simple and low speed. The logic required to act on the commands received may however be more complex. In simple cases, this may be built using a relatively simple finite state machine in the logic of the FPGA. More complex logic would require considerable resources on the FPGA which would be underutilised. In this case, using a software based processor can more efficiently manage the associated operations.

Similarly, more complex high-level vision processing operations may be more efficiently performed in software than in hardware on the FPGA. Interface between a serial processor and the FPGA may take place in a number of ways. At one extreme, the FPGA may be mapped into the memory address space of the serial processor. This may be implemented using one or more dual-port memory blocks as shared memory, with one port accessible from the logic implemented on the FPGA, and the second port reserved for use by the serial processor. Shared memory is well suited for transferring large volumes of data between the systems. Alternatively, the FPGA can provide one or more control and status registers to the serial processor, with these mapped into either the I/O or memory address space. Providing such an interface to an external processor requires relatively little decoding logic unless a large number of registers are required. For looser coupling between the FPGA and external processor, a standard serial communication link could be used. Such a link is best suited when only low data volumes must be transferred between systems.

25.7 Algorithm Tuning and Debugging

Many algorithms have threshold levels and other parameters that must be tuned to give the best results. Where possible, the algorithm should be developed so that it is self-tuning as much as possible, because this reduces the opportunities for an inexperienced operator to mistune the algorithm. However, this makes the algorithm more complex and is not always practical.

One solution is to tune the algorithm once when it is initially developed, and hard wire the corresponding parameter values into the algorithm. Changing any parameter then requires recompiling and remapping the complete design. A more practical alternative is to store the parameters in registers where they may be modified either from a separate software process or through user interaction implemented in hardware. In the latter case, additional logic will also be required to load the initial parameter values from an external flash memory or other non-volatile storage.

When running with a host computer system, many of the tuning functions can be performed and controlled by the host computer. When running in a standalone configuration, it is necessary to build all of the required user interface logic into the FPGA [15]. Sometimes the additional logic required may be more extensive than the actual application. This may be managed by building multiple configurations: one for running the application, and one or more for tuning. The tuning configuration may not necessarily require all of the application logic, for example the actuator control logic can often be omitted. During tuning, it may be necessary to display additional information, such as images that show the effects of changing each control parameter. Once tuned, it is necessary to save the parameters in non-volatile storage before loading the configuration file with the actual application. This is because reconfiguration will replace everything within the FPGA, including the state of latches and the contents of any memories.

The requirements for system and algorithm debugging are similar to those for tuning. It is often important to view the results at various stages throughout the algorithm to check that it is operating correctly. Debugging an algorithm in hardware is much more difficult than in software because it is very difficult to stop a hardware algorithm and single step through it. In software, only one thing is happening at a time, but in hardware all of the components operate concurrently with the consequence that the system can change state considerably with each clock cycle. This is also complicated on an FPGA based design because almost all of the signals are internal to the FPGA and are therefore not even accessible for testing. In simple applications, key control signals can be routed to I/O pins to enable them to be tested.

Many of these problems may be reduced by thoroughly testing the algorithm in software before implementing it in hardware. It is important that these tests are performed on the algorithm after transformation for FPGA implementation, and not just the original software algorithm. This ensures that the algorithm is correct and the final FPGA implementation step becomes one of verifying that the algorithm is behaving as designed and tested in software. Incremental development techniques can help identify and eliminate bugs in this mapping process. Each operation is implemented one at a time, with the design mapped and placed onto the FPGA and tested before adding the next operation. While rebuilding the FPGA configuration after adding each operation is quite time consuming, it can reduce time overall by simplifying the time required for debugging and testing.

Unfortunately it is not possible to test every possible case before implementing the design. This is especially so when dealing with high speed, real time systems such as required for machine vision. During the initial development and early deployment, it may be necessary to capture the images that cause the algorithm to fail for more thorough testing offline. This requires saving each image before it is processed so that it is available later. Two approaches may be used here. One is to run the system slower, capturing the input image into a frame buffer and from there saving it to an external storage device before executing the algorithm. This may require substantial changes to both the design and operation of the system. A less invasive approach is to have a parallel system that is capable of both recording and playing back a video stream at the full processing rates. While requiring considerably more development effort, such a tool would allow testing at the full operating speed enabling complex dynamic bugs to be investigated.

If the design must be debugged in hardware, one approach may be to route key signals to unused I/O pins to make them accessible from outside the FPGA. This still requires connecting an external logic analyser or oscilloscope to the pins to monitor the signals. Both Xilinx and Altera provide logic analyser cores that may be embedded within a design. These record selected signals in an internal memory block, from where they can be read out and analysed by an external programme.

Whatever debugging and tuning techniques are used, it is important for them to be integrated into the design process and not just added on at the end.

25.8 Summary and Conclusions

Field programmable gate arrays have the potential to significantly accelerate the image processing component of a machine vision system. Most machine vision algorithms are dominated by low and intermediate level image processing operations, many of which are inherently parallel. This makes them amenable to a parallel hardware implementation on an FPGA.

Compared with a software implementation, gains are made in two ways. First, a software implementation is constrained to executing only one instruction at a time. Although the phases of the instruction fetch/decode/execute cycle may be pipelined, and modern processors allow different threads to be executed on separate cores, software is fundamentally sequential. A hardware implementation, on the other hand is fundamentally parallel, with each operation or "instruction" implemented on separate hardware. In fact a hardware system must be explicitly programmed to perform operations sequentially if necessary. If an algorithm can be implemented in parallel to efficiently make use of the available hardware, considerable performance gains may be made.

Second, a serial implementation is memory bound, with data communicated from one operation or instruction to the next through memory. A software processor therefore spends a significant proportion of its time reading its input data from memory, and writing the results of each operation (including intermediate operations) to memory. On an FPGA system, each operation is implemented in parallel, on separate hardware. This allows data to be passed directly from one operation to another, significantly reducing or even eliminating this memory overhead. Memory bandwidth is also considerably expanded, because rather than operating with a single monolithic memory model as software does, multiple small memory blocks are available to be used independently and in parallel.

Despite efforts to make programming FPGAs more accessible and more like software engineering [3], efficient FPGA programming is still very difficult. This is because programming FPGAs is hardware design, not software design. Although many of the languages look like software, the compilers use the language statements to create the corresponding hardware. Sequential algorithms ported from software will by default run sequentially, usually at a significantly lower clock speed than high end CPUs. Relatively simple changes to the algorithm, such as pipelining and loop unrolling enable it to exploit some of the parallelism available, and this is often sufficient to compensate for the lower clock speed of FPGAs. However, the algorithm is still largely sequential unless it has been specifically adapted for parallel execution. The software-like languages are in reality hardware description languages, and efficient design requires a hardware mindset.

The operations within a software based image processing environment have been optimised for serial implementation within that environment. Simply porting a software algorithm onto an FPGA will generally give relatively poor performance, because it will still be predominantly a serial algorithm. Such an FPGA implementation will generally give disappointing results. This is because the performance of an FPGA based design is very sensitive to the quality of the implementation [23]. It is necessary to transform the algorithm from a software based implementation to one that can efficiently exploit the parallelism available from hardware. Such transformation requires three stages: algorithm analysis, computational architecture design, and implementation. Algorithm analysis looks at the whole algorithm to determine the structure. It is desirable, where possible, to make all of the algorithm steps share a common processing mode. This may change the underlying algorithm to one that is more readily able to map efficiently to a hardware implementation. The second step involves designing the computational architecture that is implied by the algorithm. This may require further transformation of the algorithm to result in a resource efficient parallel architecture. The final step is to then map the algorithm onto the architecture to obtain the resultant implementation. This process can be iterative with the algorithm and architecture mappings being updated as different design decisions are made. Good algorithm transformation is currently beyond the capabilities of automated development tools (although pipelining and loop unrolling can be automated to some extent). The design process requires a human in the loop who understands the detailed operation of the whole algorithm and is thinking in terms of efficient hardware design.

This algorithm transformation process has been demonstrated with a relatively simple classification example. While the high-level image processing algorithm is relatively unchanged, at a lower implementation level, it bears little resemblance to its software counterpart. In particular, the memory requirements are significantly reduced; so much so that in this case the design has been reduced from requiring eight separate frame buffers to one where the image is not actually stored anywhere during the processing. The whole processing is performed on the data as it is streamed from the camera. This not only affects the processing speed, but also significantly reduces the latency, with most of the results being output even before the image has been completely input.

The benefits of an FPGA based implementation are obvious, even from this relatively simple example. What is less obvious are some of the pitfalls and limitations of FPGA based design. First, as has been emphasised in this summary, it requires a different skill set to conventional software based image processing design. In itself, this is less critical with the design of machine vision systems as a wide range of both hardware and software skills are required within the development team [9]. Second, the design, development, and debugging of concurrent hardware systems is both more difficult and time consuming than developing the equivalent software based system. This is partly because the system is effectively developed twice, first as software, and then as hardware. Therefore, if a software based system is effective for the task, it is probably not worth considering an FPGA based implementation. Third, the clock rates are typically one to two orders of magnitude lower in an FPGA design than on a high end CPU. Therefore the parallel design must accelerate the algorithm by at least 10–100 times just to break even. Amdahl's law requires that between 90% and 99% of the algorithm be efficiently parallelisable just to run at the same speed as a serial implementation on a high end CPU. To gain any significant real acceleration or reduction in latency, an even higher percentage of parallelism needs to be exploited.

Fortunately, the low and intermediate level image processing operations typically used in a machine vision algorithm are readily parallelisable. An FPGA implementation results in a smaller and significantly lower power design that combines the flexibility and programmability of software with the speed and parallelism of hardware.

Acknowledgements

I would like to acknowledge my research students (David Johnson, Kim Gribbon, Chris Johnston, Aaron Bishell, Andreas Buhler, and Ni Ma) who helped shape my thinking as we struggled together to work out efficient ways of implementing image processing algorithms.

References

1. AIA (2004) Camera link specifications, vol 1.1. Automated Imaging Association, Ann Arbor, MI
2. Akeila H, Morris J (2008) High resolution stereo in real time. In: Robot vision, Auckland, New Zealand, 18–20 Feb 2008. Lecture notes in computer science, vol 4931, Springer-Verlag, Berlin, pp 72–84. doi:10.1007/978-3-540-78157-8_6
3. Alston I, Madahar B (2002) From C to netlists: hardware engineering for software engineers? IEE Electron Commun Eng J 14(4):165–173. doi:10.1049/ecej:20020404

4. Altera (2010) Cyclone IV device handbook, vol CYIV-5 V1-1.1. Altera Corporation, San Jose, CA
5. Altera (2010) Stratix IV device handbook, vol SIV5V1-4.0. Altera Corporation, San Jose, CA
6. Amdahl GM (1967) Validity of the single processor approach to achieving large scale computing capabilities. In: AFIPS spring joint computer conference, Atlantic City, 18–20 April 1967, vol 30. ACM, New York, pp 483–485. doi:10.1145/1465482.1465560
7. Appiah K, Hunter A, Dickenson P, Owens J (2008) A run-length based connected component algorithm for FPGA implementation. In: International conference on field programmable technology, Taipei, 7–10 Dec 2008, pp 177–184. doi:10.1109/FPT.2008.4762381
8. Bailey DG (2011) Image border management for FPGA based filters. In: 6th IEEE international symposium on electronic design, test and applications, Queenstown, 17–19 Jan 2011, pp 144–149. doi:10.1109/DELTA.2011.34
9. Bailey DG (1988) Machine vision: a multidisciplinary systems engineering problem. In: Hybrid image and signal processing, Orlando, 7–8 April 1988, vol SPIE 939, pp 148–155
10. Bailey DG (1991) Raster based region growing. In: 6th New Zealand image processing workshop, Lower Hutt, 29–30 August 1991, pp 21–26
11. Bailey DG, Bouganis CS (2009) Implementation of a foveal vision mapping. In: International conference on field programmable technology (FPT'09), Sydney, 9–11 Dec 2009, pp 22–29. doi:10.1109/FPT.2009.5377646
12. Bailey DG, Johnston CT (2010) Algorithm transformation for FPGA implementation. In: 5th IEEE international symposium on electronic design, test and applications (DELTA 2010), Ho Chi Minh City, 13–15 Jan 2010, pp 77–81. doi:10.1109/DELTA.2010.17
13. Bailey DG, Johnston CT (2007) Single pass connected components analysis. In: Image and vision computing New Zealand (IVCNZ), Hamilton, 5–7 Dec 2007, pp 282–287
14. Bailey DG, Johnston CT, Ma N (2008) Connected components analysis of streamed images. In: International conference on field programmable logic and applications (FPL 2008), Heidelberg, 8–10 Sep 2008, pp 679–682. doi:10.1109/FPL.2008.4630038
15. Buhler A (2007) GateOS: a minimalist windowing environment and operating system for FPGAs. Master of Engineering Thesis, Institute of Information Sciences and Technology, Massey University. hdl:10179/667
16. Catmull E, Smith AR (1980) 3-D transformations of images in scanline order. ACM SIGGRAPH Comput Graph 14(3):279–285. doi:10.1145/965105.807505
17. DDWG (1999) Digital visual interface (DVI), vol 1.0. Digital Display Working Group, Vancouver, WA
18. Downton A, Crookes D (1998) Parallel architectures for image processing. IEE Electron Commun Eng J 10(3):139–151. doi:10.1049/ecej:19980307
19. Edwards SA (2006) The challenges of synthesizing hardware from C-like languages. IEEE Des Test Comput 23(5):375–383. doi:10.1109/MDT.2006.134
20. Gribbon KT, Bailey DG (2004) A novel approach to real time bilinear interpolation. In: 2nd IEEE international workshop on electronic design, test, and applications (DELTA 2004), Perth, 28–30 Jan 2004, pp 126–131. doi:10.1109/DELTA.2004.10055
21. Gribbon KT, Bailey DG, Bainbridge-Smith A (2007) Development issues in using FPGAs for image processing. In: Image and Vision Computing New Zealand (IVCNZ), Hamilton, 5–7 Dec 2007, pp 217–222
22. Gribbon KT, Johnston CT, Bailey DG (2003) A real-time FPGA implementation of a barrel distortion correction algorithm with bilinear interpolation. In: Image and Vision Computing New Zealand (IVCNZ'03), Palmerston North, 26–28 Nov 2003, pp 408–413
23. Herbordt MC, VanCourt T, Gu Y, Sukhwani B, Conti A, Model J, DiSabello D (2007) Achieving high performance with FPGA-based computing. IEEE Comput 40(3):50–57. doi:10.1109/MC.2007.79
24. Hsia SC (2004) Fast high-quality color-filter-array interpolation method for digital camera systems. J Electron Imaging 13(1):244–247. doi:10.1117/1.1631443
25. Hutton M, Schleicher J, Lewis D, Pedersen B, Yuan R, Kaptanoglu S, Baeckler G, Ratchev B, Padalia K, Bourgeault M, Lee A, Kim H, Saini R (2004) Improving FPGA performance and area using an adaptive logic module. In: 14th international conference on field programmable logic and applications, Antwerp, 29 Aug–1 Sep 2004. Lecture notes in computer science, vol 3203. Springer-Verlag, Berlin, pp 135–144. doi:10.1007/b99787
26. Johnston CT, Bailey DG (2008) FPGA implementation of a single pass connected components algorithm. In: IEEE international symposium on electronic design, test and applications (DELTA 2008), Hong Kong, 23–25 Jan 2008, pp 228–231. doi:10.1109/DELTA.2008.21
27. Kehtarnavaz N, Gamadia M (2006) Real-time image and video processing: from research to reality. In: Synthesis lectures on image, video and multimedia processing. Morgan & Claypool, San Rafael, CA. doi:10.2200/S00021ED1V01Y200604IVM005
28. Ma N, Bailey D, Johnston C (2008) Optimised single pass connected components analysis. In: International conference on field programmable technology, Taipei, 8–10 Dec 2008, pp 185–192. doi:10.1109/FPT.2008.4762382

29. McCollum AJ, Bowman CC, Daniels PA, Batchelor BG (1988) A histogram modification unit for real-time image enhancement. Comput Vis Graph Image Process 42(3):387–398. doi:10.1016/S0734-189X(88)80047-1
30. Mentor (2010) Handel-C language reference manual, release v5.3_2. Mentor Graphics Corporation, Wilsonville, OR
31. Page I (1996) Closing the gap between hardware and software: hardware-software cosynthesis at Oxford. In: IEE colloquium on hardware-software cosynthesis for reconfigurable systems (Digest No: 1996/036), Bristol, 22 February 1996, pp 2/1–2/11. doi:10.1049/ic:19960221
32. Ridler TW, Calvard S (1978) Picture thresholding using an iterative selection method. IEEE Trans Syst Man Cybernet 8(8):630–632. doi:10.1109/TSMC.1978.4310039
33. Rosenfeld A, Pfaltz J (1966) Sequential operations in digital picture processing. J Assoc Comput Machinery 13(4):471–494. doi:10.1145/321356.321357
34. Schaffer G (1984) Machine vision: a sense for computer integrated manufacturing. Am Mach 128(6): 101–129
35. Schulte MJ, Stine JE (1997) Symmetric bipartite tables for accurate function approximation. In: 13th IEEE symposium on computer arithmetic, Asilomar, 6–9 July 1997, pp 175–183. doi:10.1109/ARITH.1997.614893
36. Waltz FM (1994) Separated-kernel image processing using finite-state machines (SKIPSM). In: Machine vision applications, architectures, and systems integration III, Boston, 31 Oct–2 Nov 1994, vol SPIE 2347, pp 386–395. doi:10.1117/12.188749
37. Waltz FM, Garnaoui HH (1994) Application of SKIPSM to binary morphology. In: Machine vision applications, architectures, and systems integration III, Boston, 31 Oct–2 Nov 1994, vol SPIE 2347, pp 396–407. doi:10.1117/12.188750
38. Wolberg G (1990) Digital image warping. IEEE Computer Society Press, Los Alamitos, CA
39. Wolberg G, Sueyllam HM, Ismail MA, Ahmed KM (2000) One-dimensional resampling with inverse and forward mapping functions. J Graph Tools 5:11–33
40. Xilinx (2009) Spartan-6 family overview, vol DS160 (v1.0). Xilinx Inc, San Jose, CA
41. Xilinx (2010) Spartan-6 FPGA CLB user guide, vol UG384 (v1.1). Xilinx Inc, San Jose, CA
42. Xilinx (2010) Virtex-6 family overview, vol DS150 (v2.2). Xilinx Inc, San Jose, CA

26 Very Low-Cost In-Process Gauging System

John W. V. Miller · V. Shridhar · E. Wicke · C. Griffith
The University of Michigan-Dearborn, Dearborn, MI, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_26,

Abstract: A vision system for gauging two types of automotive parts is described in this article. One of the part types is a power steering connector, in which the depth and width of the groove, and the distance between the start of the groove and the end of the power steering line are gauged. For the second type of part, a crimped connector attached to a brake hose, measurements of interest are the two diameters of the crimp and the bell length, where the hose is inserted into the connector. A standard video camera is used to acquire the image of a back-illuminated part which is digitized and captured with a frame grabber. The basic hardware to accomplish the gauging tasks consists of a standard video camera, light source, frame grabber and industrial personal computer. In order to minimise hardware costs, a standard 50 mm C-mount camera lens and extension tube was used with the video camera. Consideration had been made to use more expensive telecentric optics so that parts placement would not cause a change in magnification with a resulting loss of accuracy. With the 50 mm lens, however, magnification effects were lessened due to the greater standoff distance between camera and part. Combined with a modern personal computer, rapid image acquisition and analysis provides 100% inspection at full production rates. Since the gauging rate exceeds the production rate by a significant factor, a single computer and frame grabber with camera multiplexer can process data in real time from up to four measurement stations simultaneously.

26.1 Introduction

With advances in microelectronics, it is possible to assemble a vision system for less than $1000 with some applications. New personal computers have processing capabilities that were only available in much more expensive workstations several years ago. Advances in multimedia hardware have provided very inexpensive image acquisition capabilities, and improved image analysis techniques have reduced the need for highly regulated (and expensive) illumination sources.

Many applications, of course, cannot be addressed with this type of hardware, especially if extensive image analysis and processing must be accomplished in a short time or if special hardware is required for image acquisition. However, for suitable applications, economical solutions are possible. This chapter describes a low-cost gauging system for two families of automotive parts, a power steering connector and a brake-hose connector.

In order to gauge these parts, a video camera is used to acquire a back-illuminated image or silhouette of the part (see ➋ Chap. 41). The video image is then digitised with a frame grabber and stored in computer memory. The digitised image is processed so that key features in the image are identified and the number of pixels between these features are found and converted to physical dimensions using a calibration factor.

Both part types need to be fabricated correctly. If a given part is out of tolerance, it can significantly degrade the safety and reliability of a vehicle where it has been installed. A back-illuminated image of a power steering connector is shown in ➋ *Fig. 26.1*. Here, the feature of interest is the groove which appears as the "notch" in the narrower part of the profile. ➋ *Figure 26.2* is an image of a brake-hose crimp. The features of interest here are the minimum diameters of the two crimped regions and the location of the leftmost minimum diameter to the end of the fitting. ➋ *Figures 26.6* and ➋ *26.7* provide additional details on these measurements.

The basic hardware to accomplish the gauging tasks consists of a standard video camera, light source, frame grabber and industrial personal computer. A standard 50 mm C-mount

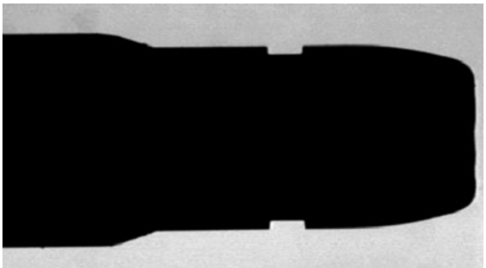


Fig. 26.1
Power steering connector

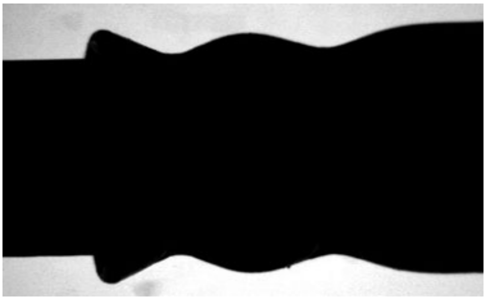


Fig. 26.2
Brake-hose connector

camera lens and extension tube was used with the video camera. For image acquisition, a low-cost frame-grabber card was selected. With a modern personal computer, rapid image analysis can be performed so that every part can be gauged at production rates. In fact, gauging speed exceeds production rates by a significant factor so that a single computer and frame grabber is able to measure parts at up to four stations simultaneously. Each station consists of a video camera and illumination source plus any necessary fixturing hardware. A camera multiplexer is used with the image capture hardware to access image data from each camera.

In order to provide a user-friendly system, a graphical user interface has been developed which displays inspection results, keeps a log file of results, and provides setup and calibration modes. A tolerance file can be created and modified for each station through the interface.

Various aspects of the system are described in the following narrative starting with optics and illumination considerations. This is followed by descriptions of the computer hardware and the gauging algorithms developed for this system. Aspects of the graphical user interface are then presented. In the last section, results and general observations are presented.

26.2 Optics and Illumination

Since the parts were to be back illuminated, it might be assumed that few lighting problems would be encountered. While this is basically true, light reflecting off of the camera lens would sometimes cause bright areas within the silhouetted part and cause measurement problems as

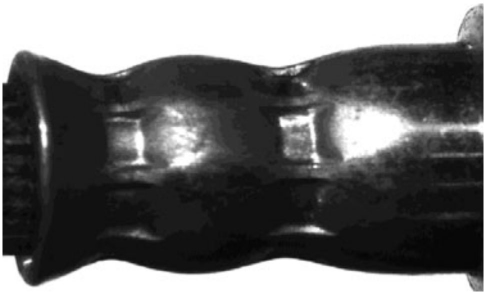


Fig. 26.3
Specular reflections

illustrated in *Fig. 26.3*. To minimise this problem, both software modifications, described later, and optical modifications were made. The classic optical solution, circular polarisation, was employed by inserting a polarizer between the object and camera lens at a 45° angle to the direction of illumination. This ensured that little light could be reflected off of the polarizer onto the part, since the polarizer greatly attenuated light reflected from the camera lens.

Rather than use a highly regulated and expensive light source, the decision was made to provide software compensation for illumination variations as described in Sect. 26.4. The main requirement was that the illumination be fairly uniform, which is not difficult given the small area of the light source. Since available space is limited, the light source needed to be compact. Initially, a fibre optics source was used, which provided excellent lighting and allowed the bulky halogen lamp and transformer to be placed some distance away from the inspection area. However, since this approach was expensive, an alternative and much less costly approach was used. Here, a low-power compact fluorescent lamp was used which is only slight larger than the fibre optics source if the ballast is located at some distance from the lamp.

In order to reduce magnification effects due to object placement relative to the camera, a 50 mm focal-length camera lens was chosen. This increased the distance between the part and camera and ensured that only relatively parallel light rays would be imaged onto the CCD array in the camera. Telecentric optics would have been a better solution but would have increased cost significantly.

26.3 Computer Hardware

As noted in the introduction, the overall system costs needed to be kept very low. Since relatively simple processing is required, reasonably high-powered PC compatible hardware can handle all of the processing. Similarly, suitable image capture hardware is required but can be satisfied with capture cards using chip sets originally developed for multimedia applications. The choice of computer processor is not critical and as noted in the discussion of the software, just about any modern personal computer will be suitable. For obvious reasons, the computer should be appropriately packaged for plant use to provide sufficient protection from the environment and ease of maintenance.

Since any standard video source should work with the image capture card, a wide range of cameras can be used for this application. However, because there is a camera for each of the four stations, costs will be raised significantly if an expensive camera model is chosen. Fortunately, there are low-cost cameras available with the necessary features such as the ability to disable gamma correction and automatic gain control. As noted previously, a camera multiplexer is used to interface the four cameras to the image capture card. USB cameras represent a newer cost-effective option but were not available during the development of this system.

In order to interface the gauging system to production equipment, a digital control card was used with optically isolated inputs and relays to provide high isolation with plant programmable logic controllers or similar hardware. When a part is in place and ready to be measured, a signal is sent to the card to initiate an image acquisition and analysis cycle. The result is sent to hardware which controls the accept/reject mechanism.

26.4 Software Development

Processing requirements for the applications of interest here are basically straightforward since the back-illuminated parts provide very high contrast and don't require complicated grey-scale operations. In order to provide adequate robustness, there are some desirable enhancements that should be provided through processing. These include the ability to handle contaminants present on the illumination source, contaminants on the actual parts, and lighting variations. The ability to accommodate lighting variations is especially important when using a low-cost source like the fluorescent lamp described previously. In addition to the image analysis software, support for accommodating up to four cameras is also required along with a user-friendly interface suitable for plant use.

26.4.1 Image Analysis Algorithms

While sophisticated image processing techniques could be used to reduce illumination dependencies, a routine to provide automatic threshold selection, based on the image histogram, was chosen. Otsu's technique works well for this application and was incorporated into the software [1]. In order to prevent serious errors if the illumination source should fail, or if no part is in the field-of-view, additional tests on the histogram are performed. These tests basically verify that there is a sufficient range of grey levels between the brightest and darkest region of the image.

Other algorithms were also optimised to improve system reliability and robustness. After obtaining the threshold, connectivity analysis is performed. Ideally, only a single large blob should be obtained, but the presence of contaminants on the light source often generates spurious blobs which must be eliminated. Several tests are used to identify the correct blob including size (pixel count) which is the most reliable predictor for blob identification.

Other concerns are that the silhouette of a part being gauged could become corrupted by a contaminant, such as a fibre, or that a new part type may have a somewhat different shape from those that were used to develop the measurement algorithms. Members of a family of parts will have similar shapes but size will vary, as well as the location for particular measurements. This implies that a given algorithm must be sufficiently robust to handle such variations. The approach used here was to convert the contour of the part being gauged into a one-dimensional signature and process this signature to identify key features.

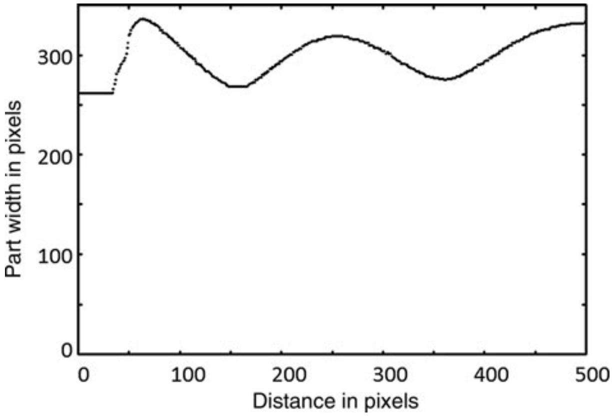


Fig. 26.4
Signature of brake-hose crimp

One-dimensional morphology is very useful both to eliminate distortion caused by contaminants and to identify the key features for measurement [2]. The basic technique for feature detection can be described by considering the signature for the brake crimp which is given in ➋ *Fig. 26.4*.

This signature was obtained by subtracting the bottom edge points from the top edge points of the binarised silhouette of ➋ *Fig. 26.2*. Since the part is symmetrical, the plot looks very similar to the top of the crimped connector. One potential problem here is that even if the part is rotated by even a small amount, the resulting lack of symmetry will adversely affect measurement accuracy. By fitting an ellipse to the binarised image of ➋ *Fig. 26.2*, the angle of the major axis can be found. A transformation is then applied to the edge points to compensate for the rotation.

Applying opening and closing operations with a small structuring element removes any distortions caused by contaminants and video noise. A closing operation is then applied to the signature with a structuring element of around 100 pixels so that the "valley" regions are filled in as shown by the horizontal lines above the signature plot in ➋ *Fig. 26.5*. Subtracting the original signature from the closed signature provides the points shown at the bottom of the plot. Here, the two "humps" correspond to the valley regions to provide an easy way to find the minimum diameters. This may seem to be overly elaborate since relatively simple search routines also work, but it is inherently more reliable over a wide range of connector sizes and shapes.

The other feature that needs to be detected for this type of part is the location where the hose and connector meet on the left side of ➋ *Fig. 26.4*. This can be found by taking the derivative of the original signature. Using the analogy of an electrical signal, this part of the connector has a "fast rise time" with a corresponding large derivative and so is easily detected.

26.4.2 Coding Considerations

These algorithms were all initially developed using the C programming language with a command-line interface. However, since the system was going to be used in a plant environment, a Windows interface was required. A relatively simple approach to achieving this goal was to write the user interface in Visual Basic and have it spawn the existing command-line

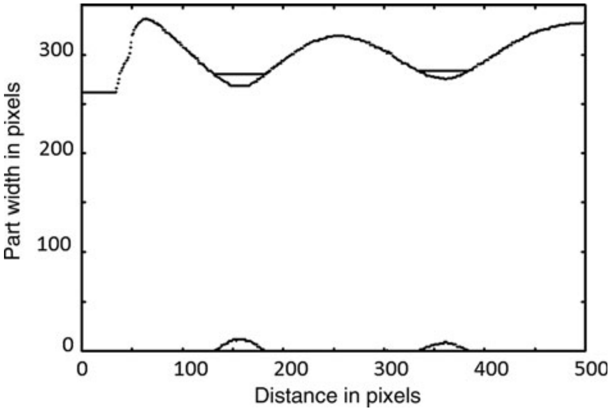


Fig. 26.5
Processing crimp signature

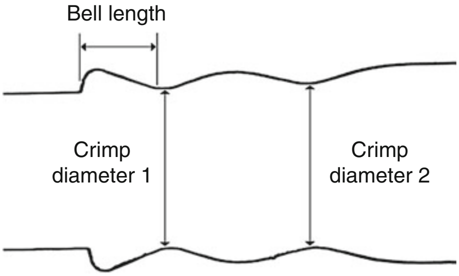


Fig. 26.6
Crimp measurements

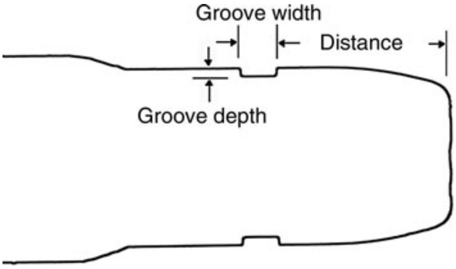


Fig. 26.7
Power steering connector measurements

executable. Visual Basic provides a straightforward means of developing Windows interfaces, and the image analysis code would not have to be rewritten.

While this approach worked, it was not very efficient given the time that was needed to spawn command-line executable code. An alternative technique was to have the analysis programme begin executing and wait to receive a signal that a part was in place which would then be gauged. Data would then be sent to the GUI so that updated results could be displayed. Unfortunately, such programmes did not share resources efficiently under a multitasking environment, so this approach was dropped.

To overcome these problems, the original image analysis code was ported to Borland's C++ Builder so that all of the software would be included in a single executable routine. The image analysis code required little modification for this, except for some C++ wrappers for interfacing to the GUI routines provided by C++ Builder. GUI development with this package was similar to Visual Basic and was straightforward considering the magnitude of work required to develop this type of interface without visual tools. The fully integrated software was much more efficient with processing times of about 1 s for a 133 MHz Pentium processor and would be correspondingly shorter with faster microprocessors. With production rates around of 5 parts/min, the system could accommodate multiple gauging stations as mentioned earlier.

However, enhancing the original gauging system so that it could handle four individual stations raised a number of software issues. Since parts in two or more stations would, on occasion, be arriving nearly simultaneously, some kind of multitasking or scheduling was required. For this reason, the approach employed used threaded code with a new thread created each time a part was ready to be gauged. It may be noted that with modern multi-core processors, very efficient and fast execution is obtained.

26.4.3 Production User Interface

A user-friendly interface was needed for setting up each of the workstations in order to control the image acquisition and processing routines as well as display the results to the user. In addition, it recorded the results in a separate inspection log for the inspected products.

The GUI contains four different forms: tolerance settings, calibration, gauging and process information. The tolerance menu is shown in ➋ *Fig. 26.8* and requires that the user sets three

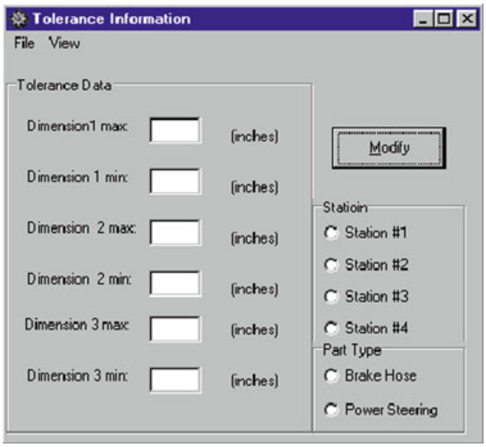
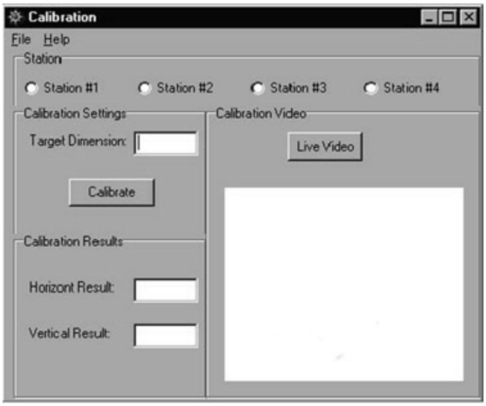


Fig. 26.8
Tolerance menu

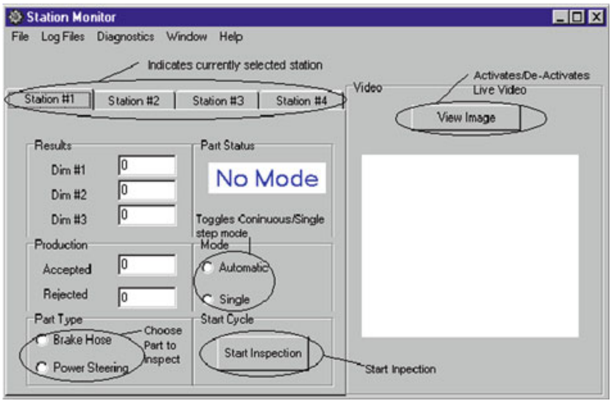
values. When the part type is selected, the text changes to identify the actual measurement parameters. The user then selects the station and enters the values which are saved by the system.

The Calibration menu, shown in ➲ *Fig. 26.9*, prompts the user to insert a calibration rod of known diameter horizontally, within the field-of-view of the camera. The system acquires the image and calculates the Y scale factor. The user is then prompted to insert the rod vertically in the field-of-view so that the X-scale factor can be calculated.

Once tolerances have been set and the station calibrated, gauging can be performed using the main interface given in ➲ *Fig. 26.10*. The user needs to select the station, part type and the mode of operation. The operating mode can be automatic for normal production use or single mode for setup and testing. When the part type is selected, the generic text for the three dimensions is replaced by the actual parameters. A running count of accepted and rejected parts is displayed and a log file is maintained which can be displayed or cleared as desired.



■ **Fig. 26.9**
Calibration menu



Dim #1 0
Dim #2 0
Dim #3 0
No Mode
Accepted 0
Rejected 0

■ **Fig. 26.10**
Main operating menu

26.5 Results and Recommendations

An evaluation of measurement accuracy indicated that consistent measurements within one or two pixels have been achieved. The system has also been tested under various conditions such as no illumination, part missing or part moving during image acquisition. The last condition is especially problematic because the two fields of the interlace can be quite different, resulting in an extremely distorted image. This affects connectivity in an especially adverse manner since a large number of irregular blobs may be generated. If the handling system is functioning properly, this should not happen but software routines to detect this fault were provided.

The interlace problem is one of the many drawbacks with RS-170 technology, although low-cost progressive scan cameras are becoming available. However, problems remain with standard analogue cameras in general, especially for gauging operations. While line-to-line repeatability is very good, progressive scan cameras still suffer from pixel jitter, unless the camera and capture hardware have synchronised pixel clocks. Achieving this is not necessarily easy or inexpensive.

A better solution is to use a digital camera. Low-cost digital cameras suitable for machine vision applications have become available since the original work was performed and would be a much better choice for this and similar applications. Such a camera will improve repeatability, and the potential exists for an order of magnitude improvement in accuracy using grey-scale interpolation techniques. While additional processing would be required, this is not likely to increase processing times given the substantially more powerful processors that are now available.

26.6 Conclusions

The vision system described here is inexpensive and can provide 100% inspection for up to four different measurement stations. It provides sufficiently accurate measurements for the intended applications and features a user interface suitable for production.

Acknowledgements

This project was funded in part by the National Science Foundation Research Experience for Undergraduates grant and by the Centre for Engineering Education and Practice at the University of Michigan-Dearborn.

References

1. Otsu NA (1979) Threshold selection method from gray-level histograms. IEEE T Syst Man Cyb 9:62–66
2. Miller JWV, Shabestari BN (1994) Detection of secondary reflections using morphology. In: SPIE proceedings of machine vision applications, architectures, and systems integration III, Boston, pp 301–306

27 Automated Handling of Coils with Flying Leads

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_27,

Abstract: During the manufacture of audio equipment, there is a need to use a robot to handle small cylindrical coils with flying leads (i.e. attached at only one end). A minimum of five cameras is needed: one overhead and four others, viewing the sides of the coil. High-contrast images can be obtained using back-lighting. Simple thresholding can then produce "clean" binary images. From the plan view of the coil, it is possible to locate the wire ends in the horizontal, [X,Y], plane.The reasoning needed to choose the best side view is well suited to PQT (➋ Chap. 23) but QT (➋ Chap. 21) would find this more difficult. From the side view, it is possible to derive one horizontal (X or Y) coordinate and the vertical coordinate (Z). Since each side view uses a different coordinate system, it is necessary to translate the side-view results into a "universal" coordinate system. Before this can be done, the system must be calibrated. (Appendix E) By combining overhead and side views, it is possible to locate the ends of both wires in 3-dimensional [X,Y,Z], space. Locating another point close to the end of a wire allows its orientation to be estimated. This is repeated for the plan and side views, giving two angular measurements, [θ,ϕ]. Five parameters [X,Y,Z,θ,ϕ] are needed by a robot to grasp the wire. (Again, a change of coordinate system is needed.) The ambiguity caused by the fact that there are two loose wires can be resolved easily by reasoning in PQT. The task of resolving real or apparent entanglement of the wires is discussed. This requires the interpretation of pairs of images and is, once again, suitable for Prolog-style reasoning in PQT. In this application, there is a high degree of uncertainty due to the flexible nature of the wires. While the image processing is trivial, the use of Artificial Intelligence techniques is essential. The matrix calculations needed to perform the coordinate-axis transformations are well suited to MATLAB and hence QT and PQT.

27.1 The Application

In this chapter, we consider the task of guiding a robot to grasp the flying leads on a small electrical coil, so that it can be connected on a printed circuit board. Once the wires have been located and grasped by the robot, they will be cut to length and then soldered to wire posts. The author encountered this application in a modern part-automated factory. The lack of robotic coil handling was the main barrier to achieving full automation in that plant.

To solve this application in practice, the system requires a far higher level of sopistication than we have space to describe here. To handle the coils properly, a robot with five degrees of freedom and at least five cameras and are needed (➋ *Fig. 27.1*). In addition, a certain degree of intelligence would be needed to interpret the images and plan the robot movements. Therefore, we shall limit our attention here to a limited but essential task: locating the ends of the wires and determining their orientation near the ends, from a single 2-dimensional view from the overhead camera.

27.2 Processing Images from the Overhead Camera

The task is to calculate three position parameters and two angles (➋ *Fig. 27.2*).

Each cameras generates a high-contrast grey-scale digital image. Converting this to binary form is straightforward. The following QT command sequence will suffice if the contrast is high enough:

```
enc,  % Enhance contrast
thr   % Threshold at mid-grey
```

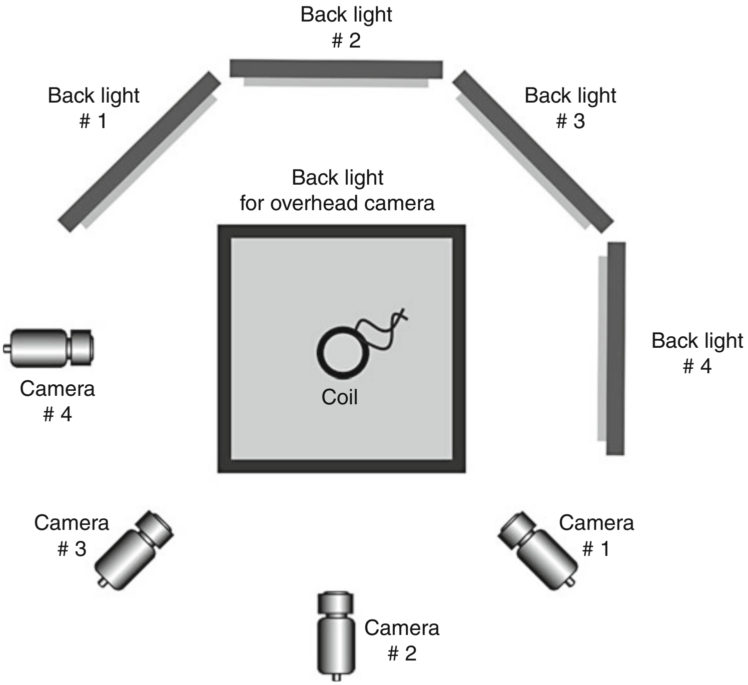


Fig. 27.1
Lighting-viewing arrangement (plan view of the coil). The grey rectangle represents the field of view of the overhead camera. From the plan view, the wire-ends are detected and located. This information is then used to choose the best side-view image. The appropriate back-light source is then switched on and the image captured from the corresponding camera (The back-light for the overhead camera would be switched off first)

If there are significant variations in the intensity of the back-ground, slightly more sophisticated processing would be required:

```
enc   % Enhance contrast
sqr   % Square all intensity values: gives greater contrast for the wires
mdf   % Median filter
crk   % Crack detector: grey-scale closing operator
```

Detecting the ends of the wires is straightforward:

```
thn   % Thinning operator
lmb   % Detect limb ends, i.e., white points with just one white 8-neighbour
```

In order to find the orientation of the ends of both of the wires, we can use the following sequence applied to the binary image (see Fig. 27.3).

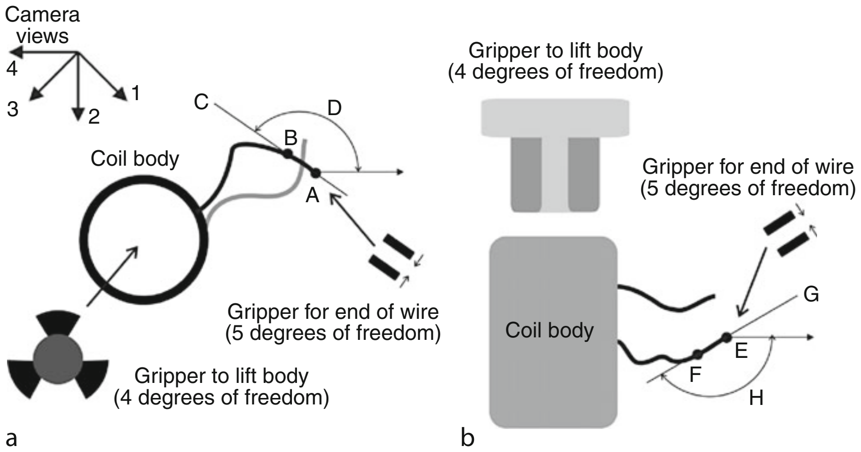


Fig. 27.2

Robotic handling of the coil. (a) *Plan view*. A is the end point of one of the wires. B is another point on the wire close to the end. C is the line joining A and B, while D is the orientation of the line AB. In this situation illustrated here, camera 1 provides the best side view. (b) *Side view*. E is the end of the wire and F is a point near its end. G is the line joining E and F, and H is its orientation. The following parameters are supplied to the robot: position coordinates, [(X_A+X_B)/2, (Y_A+Y_B)/2, (Z_E+Z_F)/2]; angles, [D, H]

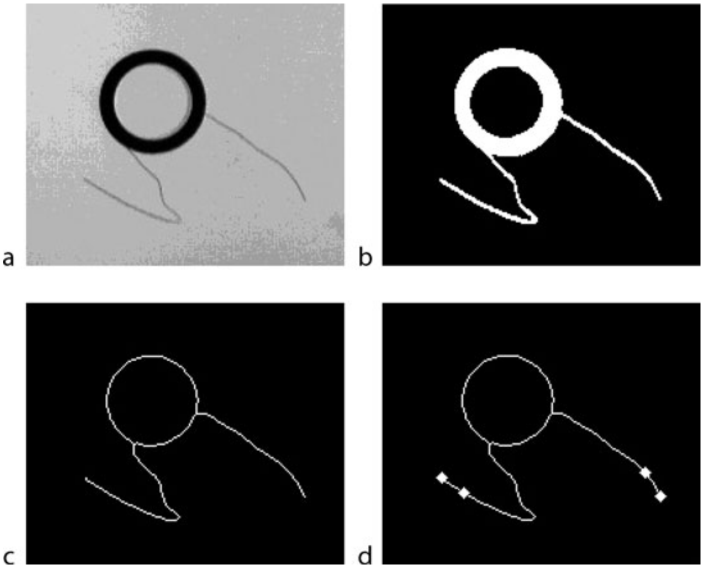


Fig. 27.3

Analysing the plan view to detect and locate the wire ends. (a) Original image. The diameter of the wires is close to the Nyquist limit, making their detection somewhat unreliable; a higher resolution would be advisable in practice. (b) Binary image. (c) After thinning. (d) End point of each wire and another point on the wire close to its end

```
thn                    % Thinning operator
spr(15)                % Recursively remove the limb-end pixels (15 loops)
exr                    % Exclusive OR - isolates the tips of the wires
big                    % Select one tip of a wire
lmb                    % Find ends of that wire tip
big                    % Select one of them (see note below)
[x1,y1]=cgr;           % Centroid
swi                    % Switch images
big(2)                 % Now, choose the other wire tip (see note below)
[x2,y2]=cgr;           % Centroid
theta=atan2(x1,y1,     % Orientation for the robot gripper
  x2,y2),
x0=(x1+x2)/2;          % X-coordinate for the robot gripper
y0=(y1+y2)/2;          % Y-coordinate for the robot gripper
```

(*Note: This naive programme does not distinguish one wire end from the other. To do so is straightforward but these details would divert us from the main point.*)

27.3 Further Analysis

The calculation described so far is only able to provide some of the information needed by the robot. It is necessary to use the position coordinates of the tip of the wire and the centre of the coil body to decide which side-view camera to use. Processing the image from the side camera will follow in a similar manner to that just described. This will provide the vertical position (Z axis) and the second angular coordinate required by the robot, as described in ➲ *Fig. 27.2*. There are three complications:

1. The overhead view must be used to decide which side view should be analysed.
2. It is not immediately obvious how the two wire ends in the plan image are related to those in the side view image (➲ *Fig. 27.4*). It may not be resolvable from a single side view.
3. The wires may appear to cross each other (➲ *Fig. 27.5*). This problem has no totally satisfactory solution, although heuristic rules might help in many cases.

These factors make it necessary to deploy a programme capable of intelligent reasoning. This type of computation is beyond the capabilities of a language like QT but is exactly the type of situation in which PQT excels. Since a full discussion of the solution is both long and tedious, we shall omit the details. Let it suffice to say that PQT does not offer a complete solution as it employs heuristics that we know will fail *occasionally*. This is not disastrous as we can often use very general (catch-all) rules to avoid situations that might otherwise lead to disasters, like machines jamming. However, such rules are often slightly inefficient/wasteful. For example, we might resort to a rule such as the following:

- *If the entanglement of the wires cannot be resolved automatically, divert the coil for reworking by hand.*

An alternative catch-all rule that we might employ is

- *If all else fails, throw the coil away.*

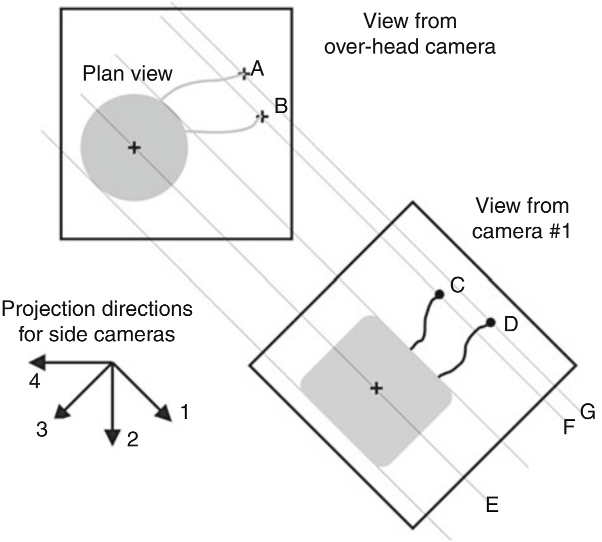


Fig. 27.4
Matching the wire ends found in the plan and side images. First, wire ends A and B are identified and located in the plan view. From this image, we first determine which camera (in this case, number 1) is the best one to use for the side view. In addition, knowing that the coil body is cylindrical allows us to predict the general form of the side-view image but with some unresolved questions. The centroids of the coil body lie along line E. Wire-end A is known to match a wire end in the side-view image somewhere along line G. Similarly, we can predict that we shall find a wire end, corresponding to B, somewhere along line F. When we examine the side-view image, we find that there is indeed a wire end (C) lying along line F and another wire end on line G. Since we can now associate A with D and B with C, we can uniquely locate both wire-ends in 3D space

Stating a rule in such a stark way is, of course, emotionally unwelcome to any factory manager. Neverthless, it may still offer a cost-effective approach, particularly if the unit cost of the coils and frequency of severe wire entanglement are both very low.

Another very serious issue that must be resolved is that of calibration. We must be able to measure image features seen in one view and compare them with measurements derived from another view. Furthermore, the robot coordinate axes must be related to those of each of the cameras. Calibration requires careful thought and planning but it is not intrisically difficult; it is simply an exercise in relating (six) different sets of coordinate axes with each other. The use of specialised calibration targets will help a lot. Their use and design requires human intelligence and we cannot supply general rules without have a specific application in mind. The subject of calibrating a simple robot vision system (overhead camera and a robot with three degrees of freedom) is covered in Appendix E.

It is worth remembering that here, as in many other applications of robot vision, a few tricks can be useful. For example, if the wires seem to cross one another, as in ➋ *Fig. 27.5c*, it might be possible to disentangle them by using the robot to place a smooth rod to pull them gently apart. (This is akin to using a single finger to unravel a loosely tangled pile of string, simply by pulling. If this manoeuvre fails, it is sensible to remove the finger and give up the

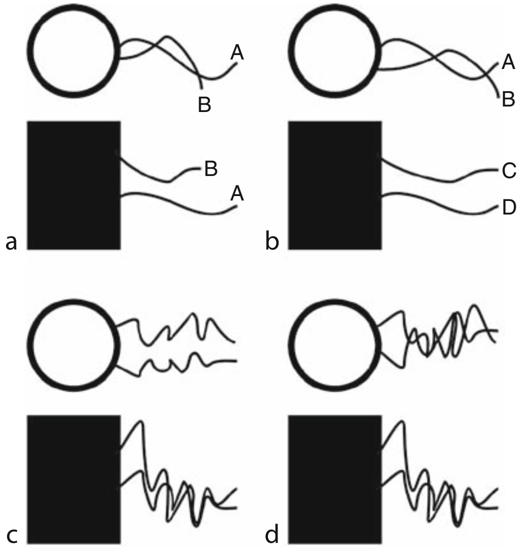


Fig. 27.5

Analysing pairs of images. (a) Fully resolvable situation. Both images show that wire-end B is markedly closer to the body of the coil than A. (b) Partially resolvable situation. The two cross-over points visible in the plan image do not cause problems, as there are no cross-overs in the side view. Since A and B are at the same distance measured horizontally from the coil body, there is an unresolved ambiguity: does A correspond to C or D? A second side view might be able to resolve this. (c) As in (b). Even though there are multiple cross-over points in the side image, the plan image shows that the wires are not tangled. (d) Unresolvable. There are multiple cross-over point in both images. In this case, the coil could be set aside for human examination and possible reworking, or simply discarded

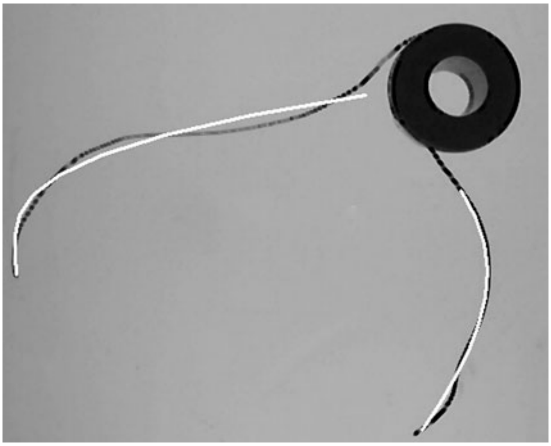


Fig. 27.6

Fitting (third order) polynomial curves (separately) to the wires

attempt.) Obviously, this stratagem will not always work, but it is certainly worthwhile trying, particularly when it is known beforehand that there is unlikely to be anything worse than very simple entanglement.

Sometimes, a wires may become severely kinked. If this is possible, it is sensible to use a curve-fitting algorithm to find a smoother representation of their general trend. For example, in ➋ *Fig. 27.6*, a third order polynomial curve has been fitted to each wire. Using this, rather than the skeleton of the wire itself, may provide a less erratic measure of the orientation of the wire at the pick-up point. In this example, the author chose to use the vertical axis for the independent variable in the polynomial series. He also defined the limits of variation of the independent variable and order of the polynomial. Clearly, these choices would all have to be automated in practice, emphasising the point that handling objects with loose wires is an exercise in Artificial Intelligence, rather than the use of standard image processing techniques.

28 A Telecentric Vision System for Broach Verification

John W. V. Miller[1] · *Spencer D. Luster*[2] · *K. D. Whitehead*[1]
[1]The University of Michigan-Dearborn, Dearborn, MI, USA
[2]Light Works, Toledo, OH, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_28,

Abstract: A common machine vision task is to verify that a product has been properly fabricated or assembled. In the following chapter, a vision system is described for confirming that a type of gear has been machined properly. The main problem in this application is the relatively large depth of the gear which results in a more complex image than the desired silhouette of the part. The resulting image shows a portion of the inner wall due to the varying magnification because some points on this wall are closer to the lens than others. Employing telecentric optics, however, greatly simplifies the problem since only light rays parallel to the optical axis create the image so that a good silhouette of the part is obtained. Gear teeth can then be isolated using mathematical morphology techniques to verify that the part has been properly broached.

28.1 Introduction

Broaching is a common cutting operation during the manufacturing of metal components including gear teeth. While this technology is very mature and produces high-quality parts, occasionally, problems are encountered which include parts that have not been broached or broached more than once. These types of problems are caused by improper parts handling in which processed and unprocessed parts are mixed. When this happens, the potential consequences are severe since improperly broached parts may not be detected and are incorporated into finished assemblies. Often the faulty part is detected at this point so that it can be replaced at additional cost. In some instances, however, the assembly functions normally, and results in a failure later with greatly increased costs and customer dissatisfaction.

Human inspectors have traditionally been used to identify this type of problem before an improperly broached part is used in an assembly. However, this approach is not sufficiently accurate since inspectors can make mistakes. A vision system that would detect if a part had been properly broached can eliminate this problem if it is reliable enough.

In principle, this is a relatively straightforward task since a back-illuminated gear provides a high-contrast image which can be analyzed by a vision system. However, because gears with internal teeth may have significant depth, the inner walls are visible which makes the vision task considerably more difficult as shown in ➋ *Fig. 28.1*. The walls are visible due to magnification which changes as a function of distance from the camera. A feature close to the camera will appear to be larger than an identical feature at a greater distance. This type of problem can be solved by using telecentric optics which provides constant magnification and reduces constraints on the working distance. Another advantage of telecentric optics is that it provides constant perspective so that the object does not have to be precisely aligned with the camera. It may be noted that a definite asymmetry exists in the gear wall of ➋ *Fig. 28.2* because of a slight misalignment between the gear and camera. There are several ways to attain telecentric results which are discussed in the next section.

28.2 Telecentric Optics

A telecentric system is one in which one or both of the pupils (object and image space) are located at infinity and the principal ray enters or leaves the system parallel to the axis. Another way to think of this is to remember the definition of a lens focal point. Rays of light that are

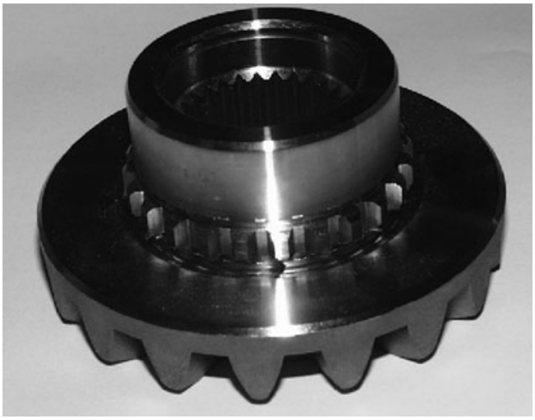


Fig. 28.1
Gear with internal teeth

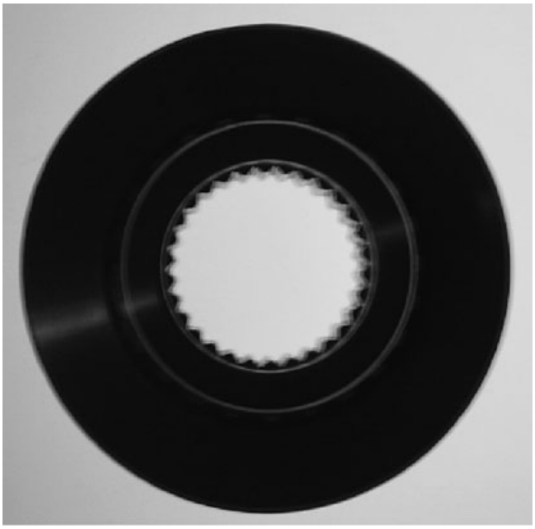


Fig. 28.2
Back-illuminated gear

parallel to the optical axis of a lens will enter the lens and converge to the focal point which is true at least for the paraxial or "thin lens" case. Also, rays of light that enter a lens after passing through the focal point will emerge parallel to the optical axis. So, if a small aperture or stop is placed at a lens focal point, only light that is nearly parallel to the optical axis can either enter or emerge from the system, depending at which focal point the stop is placed. Such an arrangement is defined to be telecentric and the aperture is the telecentric stop. ➲ *Figure 28.3* shows the three types of telecentric systems: object space, image space, and both [1].

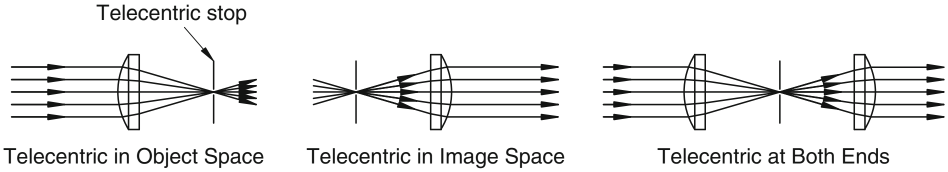


Fig. 28.3
Telecentric systems

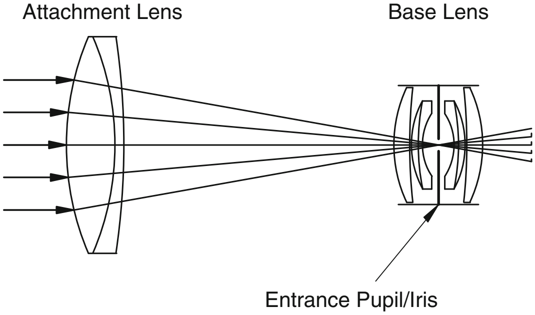


Fig. 28.4
Typical telecentric optics

28.2.1 Telecentric Approaches

There are a number of ways to build a system that is telecentric in object space. The general layout for a popular design that is commonly used in commercial and custom-designed telecentric lenses is shown in *Fig. 28.4*. Using currently popular terminology, the primary or "base" lens is placed such that its entrance pupil (iris) is at the back focus of the "attachment" lens. The pupil/iris of the base lens thus serves as the telecentric stop. This provides a compact design that is easily constructed for a large range of magnifications.

There are some other advantages that are somewhat less well covered in the literature. These include the ability to perform Schlieren-like inspection, as well as the potential for reducing edge location errors, especially for highly specular, thick objects such as gear teeth. A reasonable question concerns how "telecentric" is the system. In addition, the adjustments for obtaining better results also can be addressed. The drawing in *Fig. 28.5* contains some useful information in this regard.

Here parallel rays of light are shown entering the lens, but they are not parallel to the optical axis. Of course they still converge to a point at the focal plane of the lens. The input angle that allows parallel rays of light to converge to a point located at the edge of the telecentric stop is the acceptance angle, α, of the system. As α gets smaller, the system is more telecentric. What this means is that the depth of field increases, perspective error decreases, and magnification becomes more constant. Essentially all of the advantages of telecentric optics improve up until the point at which diffraction limits performance. Also, of course, light collection decreases because only light rays that are almost perfectly parallel with the optical axis are admitted.

28.2.2 Optics for Gear Inspection

For the application of interest here, only relatively modest performance was required but it was very important that the vision system be easy to assemble, set up, and maintain. With the object space implementation for telecentric optics, the camera lens provides the needed telecentric stop (camera aperture) and it is only necessary to position the attachment lens carefully relative to the base lens. The drawback with this approach is that only relatively small fields of view are economically accommodated since the attachment lens must be larger than the field of view and of relatively high quality.

The alternative approach is better for larger fields of view since the collimating lens for the light source can be an inexpensive Fresnel lens. The aperture must be positioned accurately relative to the lens and this is more difficult than the object space implementation to set up. Since the gear hole size was approximately 2 in., the object space implementation was selected.

Using this setup, the gear walls disappear and a clear view of the gear teeth is obtained as illustrated in ➲ *Fig. 28.6*. ➲ *Figure 28.7* shows a part which has not been broached and appears significantly different from the broached part. These images are much easier to analyze for

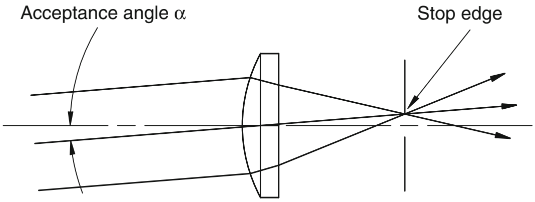


Fig. 28.5
Limits on telecentric performance

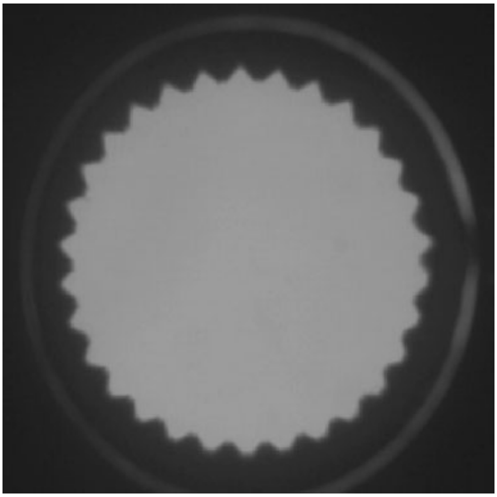


Fig. 28.6
Telecentric view of broached part

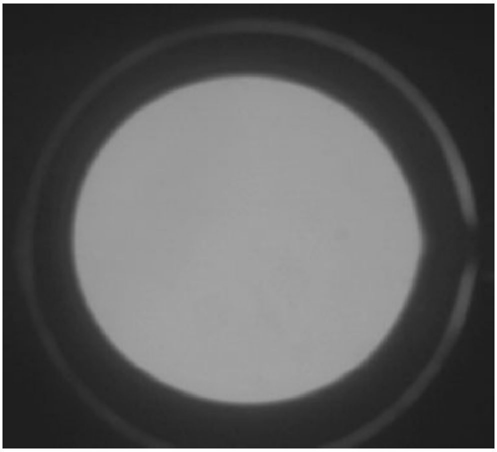


Fig. 28.7
Telecentric view of unbroached part

broach verification. It may be noted that the quality of the attachment lens was relatively low and it had significant distortion at the edges. This can be corrected either by using a better lens or a larger lens since distortion is greatest near the edges. This was not a critical consideration for the application because it was only necessary to detect the gear teeth. A better lens would be required if actual measurements were to be taken.

28.3 Image Analysis

While it is trivial for a person to tell the difference between these two images (*Figs. 28.6* and *28.7*), it is necessary to develop a robust technique for detecting the broach pattern. One such approach is based on mathematical morphology in which images are analyzed using special graphical patterns called structuring elements. The detection of gear teeth by this technique was one of the early uses of mathematical morphology and is very reliable [2]. In the approach used here, a gray-scale closing operation is used to eliminate the dark teeth.

28.3.1 Gear Teeth Detection Algorithm

The broach pattern in figure can be identified using a structuring element that is large enough to cover the bright regions between the splines. By performing a gray-scale closing operation with this structuring element, a new image is obtained which resembles the unbroached part in *Fig. 28.7*. By taking the difference between the original and processed images, the image given in *Fig. 28.8* is generated. The broach pattern is quite visible after this operation. This may be contrasted to *Fig. 28.9* which shows the results of applying this algorithm to a part that has not been broached.

Otsu's thresholding method was then used to binarize the image [3]. By using an automatic thresholding routine, the probability of problems with light level variations is minimized.

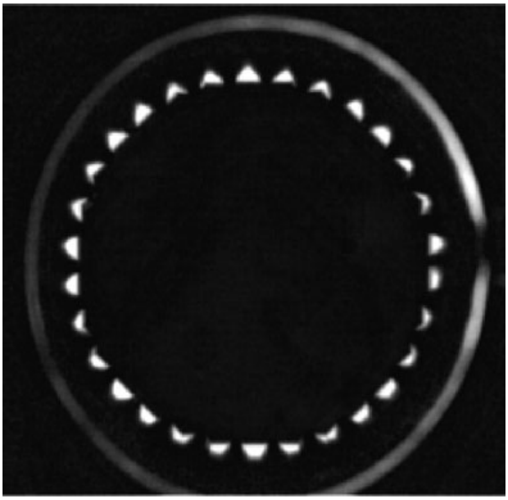


Fig. 28.8
Difference between closed and original image

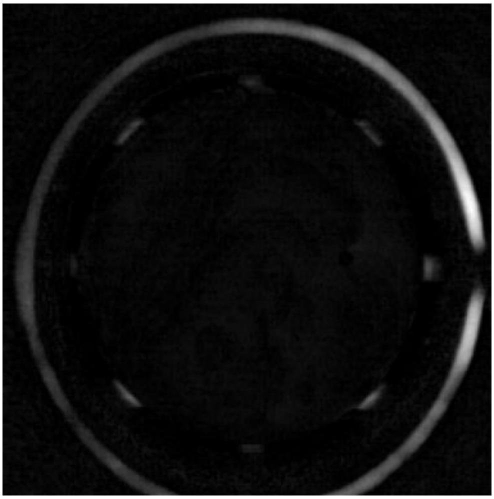


Fig. 28.9
Difference image for unbroached part

Otsu's method works well in many vision applications and always appeared to give good reliable results here as illustrated in *Figs. 28.10* and *28.11*.

Connectivity is performed to provide a list of blobs (connected groups of pixels). Two basic operations are employed here in which the image is converted to run-length format and then the runs are assigned to blobs.

A recursive operation is performed in which a run is initially assigned a label followed by a search of candidate runs on the next line to see if they should be linked to the newly created

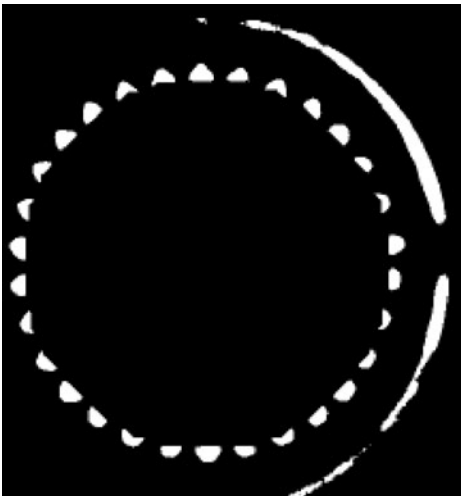


Fig. 28.10
Binarized image of broached part

Fig. 28.11
Binarized image of part that has not been broached

blob. If a run is found that should be assigned to the current blob, it is added and a new search for runs to link is initiated. When no more runs can be assigned, the next available unassigned run is assigned a new blob label and the process is repeated.

This continues until all runs have been assigned to blobs. This approach has two advantages. First, there is no need to re-label runs when two labels have been assigned to runs which ultimately will be assigned to the same blob. This is because runs are assigned to a single blob until all available runs have been properly assigned. The second advantage is that the assignment list of runs is properly ordered. This can save additional processing for some vision applications although it was not needed for the application of interest here.

After connectivity, the blob count is recorded and features are extracted from each blob to determine whether or not the given part has been properly broached. Key features here are blob area and aspect ratio since the gear teeth will be relatively similar in area and aspect ratio. The blob count is also a very good indicator of broached parts since parts that have not been broached will have relatively few blobs. After each blob is tested, the program decides whether the part has met the criterion for good parts. A portion of a ring is also detected but it is actually caused by the optical configuration and would not appear with better masking. It should be noted that the ring features are very different from teeth features. Similarly, some distortion of the teeth is also apparent and likewise would be considerably reduced less with improved optics.

28.4 Results and Conclusions

The current algorithm provides a very robust approach for determining if a part has been broached. Both the blob count and blob features are very reliable indicators with essentially total reliability at least in a laboratory setting and should be very effective on the plant floor. It is somewhat harder to characterize how well double broached parts will be identified since only a few samples were available and there can be considerable variation between parts that have been double broached. In general, teeth are considerably reduced in size when this happens so that blob area can be used to identify a part that has been broached twice. With the current optics, this should still be very reliable in most cases and using a better attachment lens will improve discrimination for marginal cases. Since double-broached parts are relatively rare anyway, even the current optics should be very satisfactory. The lack of insensitivity for part placement relative to the camera is a significant advantage of telecentric optics. The use of automatic thresholding is also expected to enhance robustness of the vision system. Voltage fluctuations and long-term light changes should not affect performance adversely within reasonable limits.

References

1. Luster S (1998) Telecentric imaging: a camera lens, a doublet, and thou. In: Applied machine vision '99 Conference, Nashville, 19–21 May 1998
2. Sternberg SR (1983) Industrial computer vision by mathematical morphology. SME technical paper #MS830413, Society of Manufacturing Engineers, Dearborn
3. http://iul.cs.byu.edu/morse/550-F95/node25.html

29 Challenges of Low Angle Metal Surface (Crosshead) Inspection

J. P. Sacha[1] · *Spencer D. Luster*[2] · *Behrouz N. Shabestari*[1] ·
John W. V. Miller[3] · *P. Hamel*[4]
[1]Edison Industrial Systems Center, Toledo, OH, USA
[2]Light Works, LLC, Toledo, OH, USA
[3]The University of Michigan-Dearborn, Dearborn, MI, USA
[4]Metal Forming and Coining Corporation, Maumee, OH, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_29,

Abstract: The Crosshead Inspection System (CIS) utilizes machine vision technology for on-line inspection of a crosshead which is a diesel engine component. The system includes three functional modules:

1. Part handling subsystem presents parts for inspection and accepts or rejects them based on signals from the image analysis software.
2. Image acquisition hardware consisting of optics, light sources and two video cameras to collect images of inspected parts.
3. Image analysis software to analyze images and send pass/fail decision signals to the handling subsystem.

The CIS acquires and inspects two images of each part. The upper camera generates an image of the part's top surface, while the lower camera generates an image of the so-called "pockets" on the part bottom. Both images are acquired when a part-in-place signal is received from the handling system. The surface inspection camera and light source are positioned at opposed low angles relative to the surface. Irregularities manifest themselves as dark regions on the surface. These regions are detected, measured and compared to user specifications. The pocket inspection function detects the presence of tumbler stones in which the contrast of these stones is enhanced with circularly polarized lighting and imaging. The graphical user interface of the CIS provides easy setup and debugging of the image processing algorithms. A database module stores and archives part inspection statistics for later evaluation. The inspection rate is sixty parts per minute.

29.1 Introduction

Metal engine components often must be fabricated within very specific highly constrained specifications. In the case of crossheads, which are used in diesel engines, the pockets and flat region on the top of the part are especially critical. The top surface, especially, needs to be very flat and free of defects. The pockets, into which rods fit precisely, also need to be defect free with no residual debris from earlier manufacturing processes. Traditionally, human inspectors sort parts but defects are sometimes missed and sorting is inconsistent since the reject criteria are highly subjective.

A system for automatic inspection of crossheads overcomes these limitations to enhance product quality while lowering labor costs. While this is conceptually a relatively straightforward application, a number of factors increase the difficulty of developing a suitable system. For example, part size varies considerably which affects the handling system, illumination, and algorithms. Certain manufacturing processes change over time that alters the basic appearance of parts and affects the contrast of key features.

The following paper describes such a system that is currently used in production including strategies for handling different parts and dealing with cosmetic variations in appearance. In the next section, the basic approach is described which is followed by a discussion of the algorithms that were developed. Finally, results are presented.

29.2 Basic Approach

Illumination is a critical aspect for the inspection tasks of interest here since parts have highly variable reflectance properties. This is especially true for the pockets since the various types of polishing operations can dramatically change their appearance. The flat surface, however, is the most critical aspect of the inspection and requires a very carefully designed and optimized optical setup.

29.2.1 Optics and Lighting

The top-surface defect types to be detected generally fall into three physical categories. The first appears as a simple, reasonably sharp edged dig or divot, often with a dark diffuse bottom. The second defect type appears as fine, high aspect-ratio scratches. The third, so-called "soap marks," are very shallow, smooth-sided dents that result from excess die compound being in place during the forming process. These defects retain the same surface reflectance as adjacent good areas. It is this last defect category, in conjunction with the varying nominal surface texture and reflectance variations that presented the greatest challenge in terms of detection.

The final form of the optical design for the top surface inspection was very low angle (as measured from the part surface) illumination with exactly opposed imaging (grazing illumination). This technique provides high contrast to detect the shallow, smooth soap-mark defects while still providing good contrast for the other defect types. Such a technique also reduces the effects of surface texture and reflectance variations. At very low angles, most reasonably smooth surfaces become highly specular according to the Fresnel equations. As an example, paper, which is normally considered fairly diffuse, appears as a good specular reflector when viewed at a very low angle with an opposed low angle light source. In addition, the angular extent of the illumination needs to be adjusted to balance good defect detection with low sensitivity to small angle variations due to part handling. These tradeoffs were achieved with a variable sized horizontal slit in front of a diffuse light source as shown in *Fig. 29.1*. Finally, in order to provide the most efficient light collection, a lens was placed in front of the diffuse light-source. Assuming that the part top surface is a plane mirror, this lens is configured so that the image of the slit is formed at the small entrance pupil of the camera lens. In this way all of the light that reflects from such a hypothetical mirror enters the lens. For the real part surface, any defect feature that causes a redirection or absorption of the light will generate a dark spot in the image including the shallow and otherwise nearly undetectable soap marks.

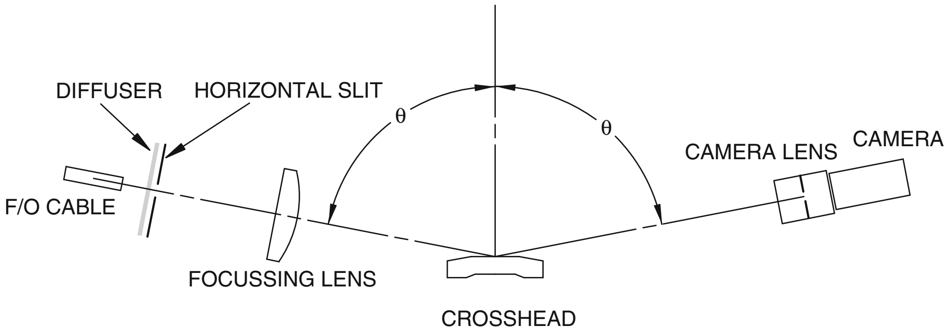


Fig. 29.1
Surface inspection optical setup

The pocket inspection, while nominally more straightforward, provided its own challenges. The main defect to be detected is the presence of tumbling stones in the pockets. These stones are generally light to dark brown in color and have a reasonably diffuse surface. After repeated processing applications, however, they can get quite dirty, and appear very dark. A secondary but still important *potential* defect type is the presence of gross cracks at the bottom of the pocket. By eye these appear as dark canyons. The metal pockets of the crosshead themselves are generally reasonably specular, but they have an even greater variation in surface reflection than the top surface.

Under conventional direct illumination and monochrome imaging, the stones can appear to have the same brightness as the bottom of the pocket. Even using color imaging, the dirty stones may be almost indistinguishable. An initial approach to enhance the stone contrast was to use a linear polarizer in front of the light source and a crossed linear polarizer (analyzer) in front of the camera lens. By this method, specular reflections can be almost completely eliminated. Linearly polarized light that reflects from a diffuse surface is, to a certain extent, re-randomized, and some component of this reflected light can pass through the crossed analyzer. In this way the diffuse stones, almost regardless of color, can be made to appear as bright objects against a dark background.

Unfortunately, such a technique leaves some gross cracks almost undetectable. The problem is that the specular pocket bottom appears as dark as the non-reflecting crack. One method of correcting this problem is to rotate the polarizer or analyzer slightly to allow some specular light to be transmitted to the camera. Mechanically and from a maintenance point of view, however, this is not optimum. Disassembly or replacement of optical components would require a difficult realignment of the linear polarizers.

The final solution was to use a single circular polarizer in front of the light source and camera as shown in ➲ *Fig. 29.2*. In certain respects a circular polarizer (CP) behaves in the same

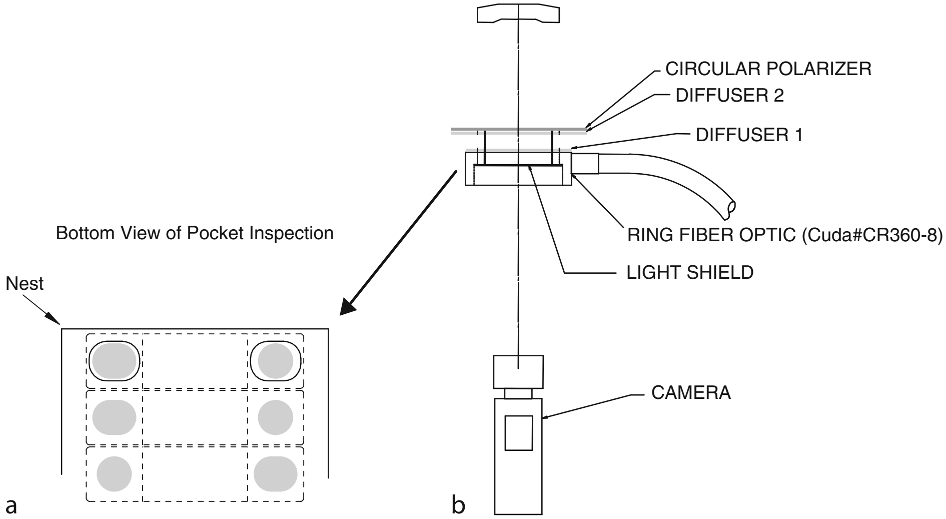

CIRCULAR POLARIZER
DIFFUSER 2
DIFFUSER 1
RING FIBER OPTIC (Cuda#CR360-8)
LIGHT SHIELD
Bottom View of Pocket Inspection
Nest
CAMERA
a
b

Fig. 29.2
Pocket inspection setup: (a) Pockets are visible through holes in the nest. (b) Optical setup, including ring lighting, polarizer, camera, and lens

way as a linear polarizer with a crossed analyzer. Light that passes through the circular polarizer and is specularly reflected from an object is blocked from passing back through the CP. The CP, however, is more dependent on light wavelength for proper performance. When used in broad band white light, as is the case for this application, some portions of the light spectrum pass through the CP and back when specularly reflected. The stones, whether clean or dirty, remain as the brightest objects to be observed. The pocket bottom remains darker, but not black. The gross cracks, by virtue of not reflecting any light, are seen as black features. Furthermore, the CP is not sensitive to rotational orientation and may be disassembled for cleaning or replaced with no need for critical realignment.

29.2.2 Image Acquisition and Analysis Hardware

Two standard CCD cameras were used to acquire images of the flat surface and pockets. The camera video was digitized using a frame grabber that could accept up to four cameras. A standard personal computer analyzed the data.

29.2.3 Handling System

The custom handling system was designed to accommodate four different crosshead designs with as little changeover time as possible. The motor-and-cam operated system has an in-feed stage consisting of a simple conveyor. Currently, parts must be arranged manually on this long conveyor into a single line. The system was designed, however, to accommodate hopper storage and vibratory bowl feeding in the future.

Once the parts are in line, the conveyor feeds them to a nest for positioning. Here the direction is changed and parts are oriented side by side as shown in ➲ *Fig. 29.3*. A pusher rod connected to the main cam slides the parts down one at a time until they reach the inspection station where an inductive sensor provides a *part-in-place* signal to the inspection system. A second pusher arm, connected to the same cam, slides the part out, once more moving in a direction parallel to its long axis. The crosshead then travels down a chute that has an electro-pneumatic diverter at the end. Before the part reaches the diverter, the inspection system provides an accept or reject signal to the diverter.

In order to change part types, the in-feed conveyor position must be adjusted and a nest for the new part mounted. This changeover only requires about 10 minutes.

It should be noted that the original handling concept included independent computer control of the in-feed, pusher arms, and out-feed mechanisms. The customer, however, preferred purely mechanical cam operation. Their opinion was that this was more reliable and more easily debugged. Such a concept also limits the flexibility of the system and part reject timing became a critical challenge.

29.3 Analysis Algorithms

As in many machine-vision applications, conflicting requirements include data processing, reliable inspections and keeping development and equipment costs under control. A general-purpose image processing library developed at the Edison Industrial Systems Center was

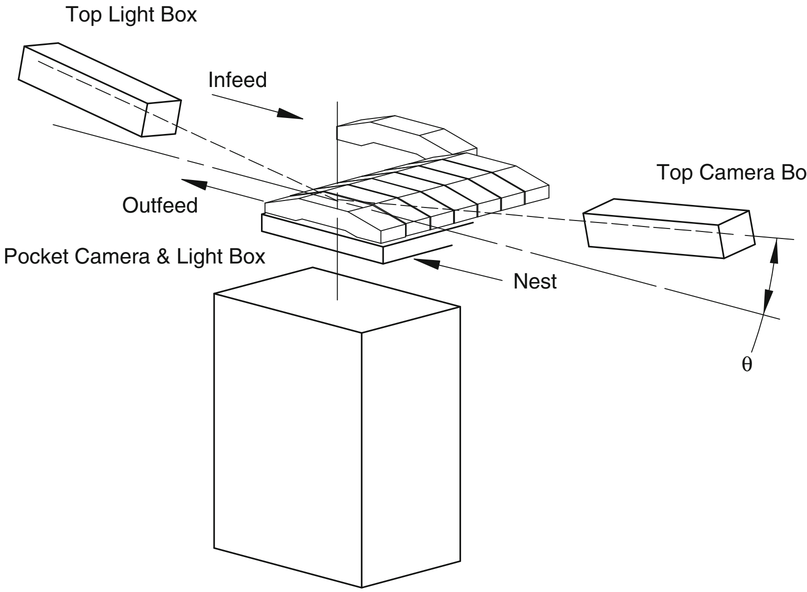


Fig. 29.3
Crosshead inspection system inspection setup

employed here [1] to speed development and allowed the use of a personal computer for image processing and data analysis.

29.3.1 Surface Inspection

The surface inspection algorithm consists of two main parts: finding the location of the surface for inspection (for smaller parts, several of them may be visible in the image) and then analyzing the surface image. The algorithm consists of the following steps:

1. Find a suitable threshold for binarizing the image
   (a) Create a histogram of the image
   (b) Fit a mixture of two Gaussian distributions to the histogram
   (c) Set threshold midway between peaks of Gaussians
2. Identify the surface
   (a) Threshold the input image using value found in the previous step
   (b) Find the largest blob – valid blob area limits are determined during calibration
   (c) Smooth the blob's boundary
   (d) Shrink blob boundary inwards per user-specified setting to create a region of interest (ROI)
3. Inspect surface area
   (a) Threshold image within the ROI – use the same threshold as in the previous step
   (b) Detect blobs and compute their size and depth attributes and compare each blob's attributes against user defined-rejection criteria

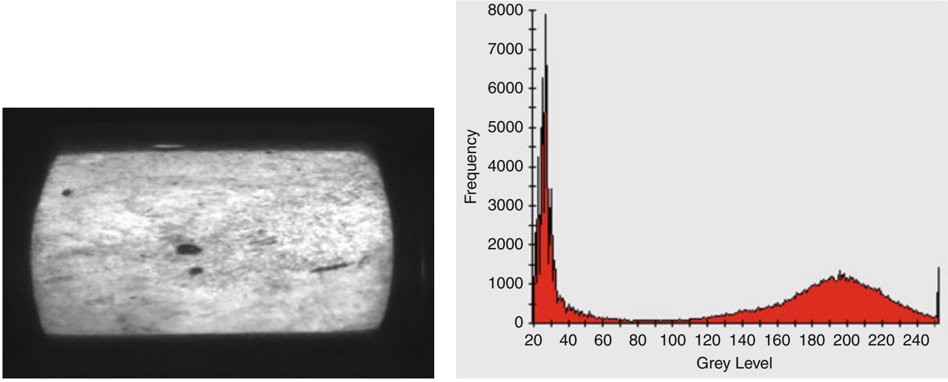


Fig. 29.4
Image of the crosshead surface (*left*) and the corresponding histogram

That thresholding algorithm is based on the *optimal thresholding* method [2]. The method assumes that the image contains two values combined with a additive Gaussian noise. The mixture probability function is

$$\begin{aligned} p(z) &= P_1 p_1(z) + P_2 p_2(z) \\ P_1 + P_2 &= 1 \end{aligned} \tag{29.1}$$

Where $p_1(z)$and $p_2(z)$ are Gaussian distributions. An example of an actual physical distribution of gray level values is shown in *Fig. 29.4* which is close to a mixture of two Gaussian distributions. The user can specify a balance parameter that shifts the threshold towards the brighter or darker peak.

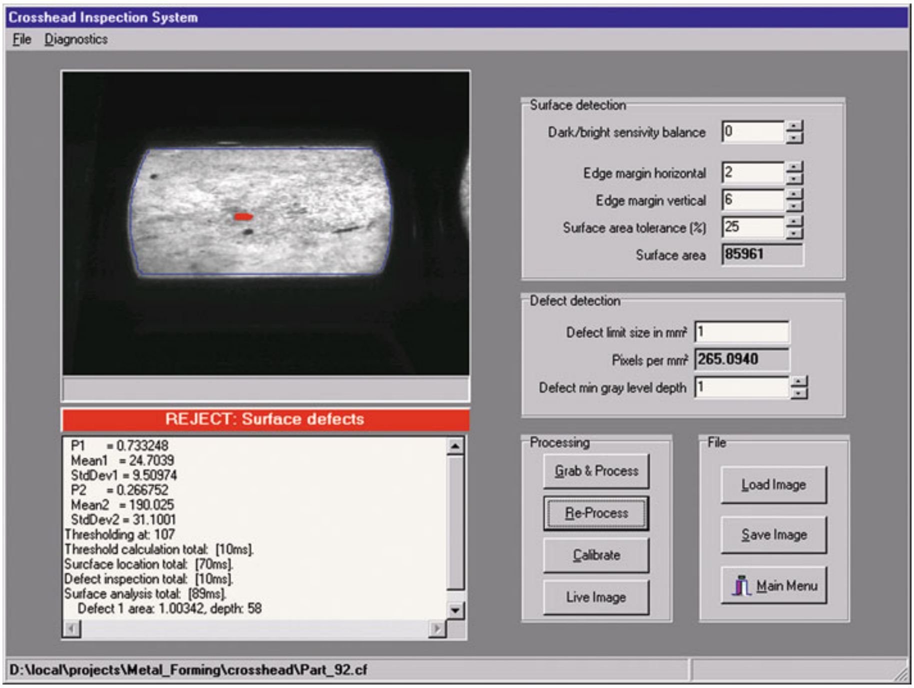
The surface area is found by thresholding the image. The edges of the surface blob are smoothed using one-dimensional morphology [3] and eroded to remove noise at the edges of the surface. This blob defines the region of interest for inspection of defects on the crosshead surface. The dark regions, represent potential surface defects which are also detected by thresholding.

29.3.1.1 Setup and Calibration

Figure 29.5 illustrates the setup and calibration screen for surface inspection. The surface calibration algorithm is used to estimate the surface area of a good part and determine actual dimensions from pixel values. The setup screen is also used to alter inspection parameters. Values calculated by the inspection algorithm are displayed in the log window and can be used to adjust sensitivity parameters.

29.3.2 Pocket Inspection

The original pocket inspection algorithm segmented the image of each pocket to detect the pocket rim, sidewall and bottom. However due to significant variations in the pocket surface finish and pocket size, the detection algorithm and optical setup were not sufficiently robust. After discussions with the client, it was decided to concentrate on the detection of stones that represented the main quality consideration. To address this objective, a cross polarization optical setup was implemented. The image processing requirements were simplified to calculating the mean gray level and standard deviation within each pocket's ROI. Example images of the pockets with no polarizer are shown in ➋ *Fig. 29.6* while images using a cross-polarizer are shown in ➋ *Fig. 29.7*. As may be seen, the polarizer significantly enhances the contrast of the stone in ➋ *Fig. 29.7*.



Dark/bright sensivity balance: 0
Edge margin horizontal: 2
Edge margin vertical: 6
Surface area tolerance (%): 25
Surface area: 85961

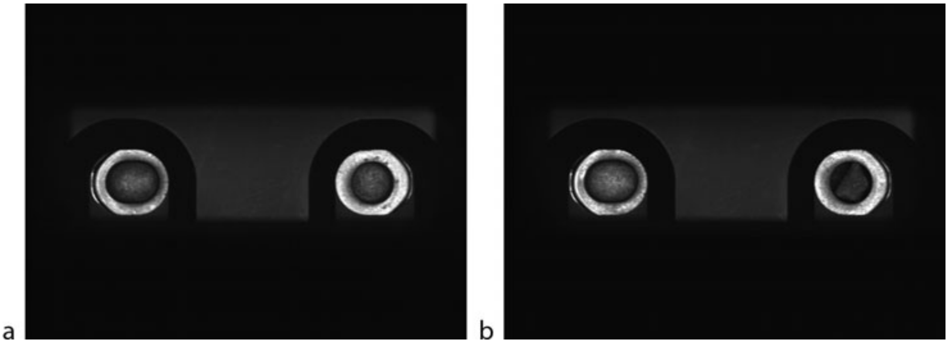
Defect limit size in mm²: 1
Pixels per mm²: 265.0940
Defect min gray level depth: 1

REJECT: Surface defects

```
 P1      = 0.733248
 Mean1   = 24.7039
 StdDev1 = 9.50974
 P2      = 0.266752
 Mean2   = 190.025
 StdDev2 = 31.1001
Thresholding at: 107
Threshold calculation total:  [10ms].
Surcface location total:  [70ms].
Defect inspection total:  [10ms].
Surface analysis total:  [89ms].
   Defect 1 area: 1.00342, depth: 58
```

D:\local\projects\Metal_Forming\crosshead\Part_92.cf

◘ Fig. 29.5
Surface Inspection diagnostics screen



◘ Fig. 29.6
Original setup without cross polarization: (a) good part, (b) part with stone in the right pocket

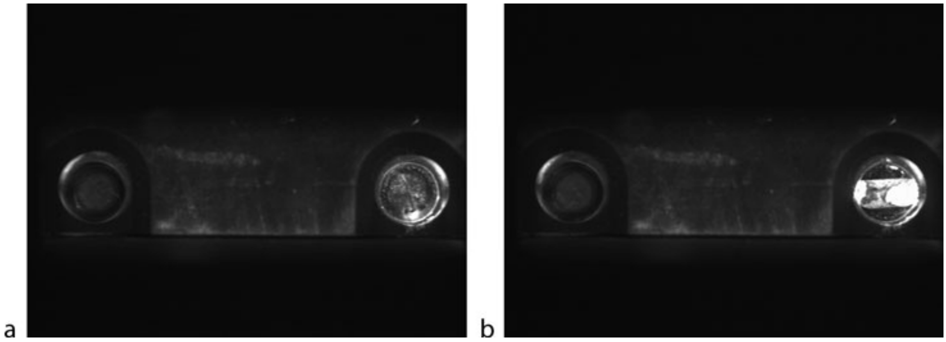
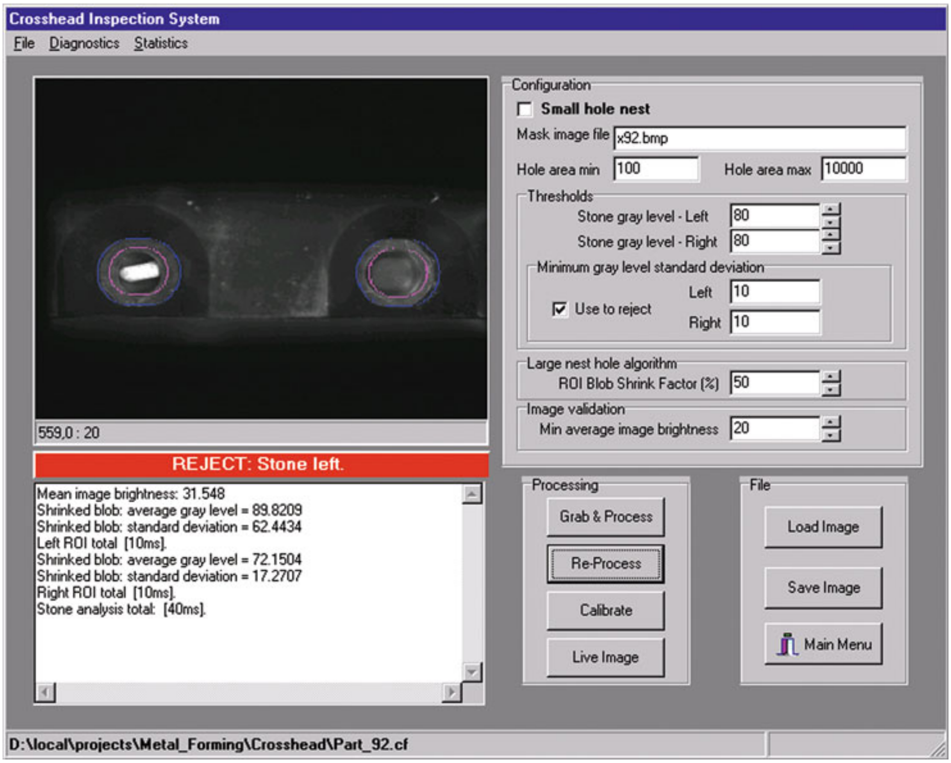


Fig. 29.7
Final setup with cross polarization filter: (a) good part, (b) part with stone in the right pocket



Configuration

□ **Small hole nest**

Mask image file: x92.bmp

Hole area min: 100 Hole area max: 10000

Thresholds

Stone gray level - Left: 80

Stone gray level - Right: 80

Minimum gray level standard deviation

☑ Use to reject Left: 10 Right: 10

Large nest hole algorithm

ROI Blob Shrink Factor (%): 50

Image validation

Min average image brightness: 20

559,0 : 20

REJECT: Stone left.

```
Mean image brightness: 31.548
Shrinked blob: average gray level = 89.8209
Shrinked blob: standard deviation = 62.4434
Left ROI total [10ms].
Shrinked blob: average gray level = 72.1504
Shrinked blob: standard deviation = 17.2707
Right ROI total [10ms].
Stone analysis total: [40ms].
```

D:\local\projects\Metal_Forming\Crosshead\Part_92.cf

Fig. 29.8
Setup and calibration screen for pocket inspection. There is a stone in the left pocket

29.3.2.1 Setup and Calibration Screen

The setup and calibration screen for pocket inspection in shown in ➲ *Fig. 29.8*. Pocket calibration is used to determine the nest hole locations through which pockets are visible. The user is asked to place a diffuser in the place of a part. The resulting image is processed to create two ROIs corresponding to the hole locations. The setup screen is also used to alter inspection parameters. Values calculated by the inspection algorithm are printed in the log window and can be used to adjust the algorithm's setup parameters. The average gray level is used to detect stones in pockets while the gray-level standard deviation detects missing pockets (pockets that are filled with metal).

29.3.3 Database

The Crosshead Inspection System program saves inspection statistics to a local Paradox database which contains the number of good and failed parts processed each day for each part type. The data can be exported in text or Excel format to a file or copied to other database through the ODBC interface.

29.4 Results

The Crosshead Inspection System described in this paper has proven to be robust in detecting crosshead surface defects and tumbling stones in crosshead pockets. An automotive part manufacturer in Northwestern Ohio currently uses the system for on-line inspection of four types of crossheads at a rate of sixty parts per minute.

Acknowledgment

This project has been founded by the Metal Forming and Coining Corporation, Maumee, Ohio, USA.

References

1. Sacha JP, Shabestari BN, Kohler T (1997) On the design of a flexible image processing library in C++. In: Machine vision applications, architectures, and Systems Integration VI, SPIE Vol. 3205, pp 82–89, Pittsburgh, 14–17 October 1997
2. Gonzalez RC, Woods RE (1992) Digital image processing. Addison-Wesley, New York
3. Miller JW, Shabestari BN (1994) Detection of secondary reflections using morphology. In: Machine vision applications, architectures, and systems integration III, SPIE vol 2347, pp 301–306, October 1994

30 A Machine Vision System for Quality Grading of Painted Slates

Ovidiu Ghita · Tim Carew · Paul F. Whelan
Dublin City University, Glasnevin, Dublin, Ireland

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_30,

Abstract: The major aim of this chapter is to detail the technology associated with a novel industrial inspection system that is able to robustly identify the visual defects present on the surface of painted slates. The development of a real-time automated slate inspection system proved to be a challenging task since the surface of the slate is painted with glossy dark colours, the slate is characterised by depth profile non-uniformities and it is transported at the inspection line via high-speed conveyors. In order to implement an industrial compliant system, in our design we had to devise a large number of novel solutions including the development of a fully customised illumination set-up and the development of flexible image-processing procedures that can accommodate the large spectrum of visual defects that can be present on the slate surface and the vibrations generated by the slate transport system. The developed machine vision system has been subjected to a thorough robustness evaluation and the reported experimental results indicate that the proposed solution can be used to replace the manual procedure that is currently employed to grade the painted slates in manufacturing environments.

30.1 Introduction

Over the past three decades, the deployment of vision-based solutions in the development of automatic inspection systems has witnessed a substantial growth. This growth was motivated in part by the increased flexibility attained by the vision-based industrial inspection solutions when applied to the classification of the manufactured products with respect to their aesthetical and functional characteristics and equally important these systems opened the opportunity to control the production process by evaluating the statistics that describe the efficiency of the overall manufacturing process. In this context, it is useful to mention that slate manufacturing is a highly automated process where product inspection being the solely task that is still performed by human operators. In this process, the human operators analyse the slates visually when they emerge via a conveyor from the paint process line and they make a judgement whether the produced units have a surface finish of acceptable quality or not. If defective slates are detected during visual inspection, they are removed from the production line and put aside for rework or rejection. Although the in-line manual inspection generates accurate results when applied to product quality assessment, it is highly reliant on the pictorial information perceived by the human visual system (HVS) [1–3] and consequently it is critically influenced by the experience and the acuity of the human operator. In addition to this, it is important to note that the performance of the human operator is not constant and moreover the manual inspection represents a monotonous work that cannot be used in conjunction with high-speed product manufacturing lines. In this context, Tobias et al. [3] list some of the key factors that justify the adoption of machine vision systems in industry. These are increased productivity, improved product quality, absence of human inspectors performing dull and monotonous tasks, high-speed inspection (matched by high-speed production) and reduced human labour costs. In particular we would like to stress that the incorporation of automation in manufacturing industry is not necessarily about the displacement of labour, but more as an answer to the expectation of an increasingly educated labour force and the compliance to the recent realities of the world economy [4].

However, the design of industry-compliant machine vision systems is far from a trivial task as several issues including the mechanical design, development of an appropriate illumination set-up, optimal interfacing between the sensing and optical equipment with the computer vision component have to be properly addressed in order to accommodate all challenges that are encountered in a typical industrial environment [5–8]. In this regard, the aim of this chapter is to detail the development of a fully integrated machine vision system that has been specifically designed to assess the surface quality of the painted slates. In this chapter we will discuss in detail the design choices that were made in the development phase of the proposed vision sensor and we will also describe the algorithmic solutions that were implemented to accommodate the large variation of the structural and paint defects that are present on the surface of the painted slates. This chapter is organised as follows. ➤ Section 30.2 discusses the related systems that have been documented in the literature. ➤ Section 30.3 provides a description of the typical slate defects and the discussion is continued in ➤ Sect. 30.4 with the brief presentation of the developed slate inspection system. In ➤ Sect. 30.5 the adopted illumination set-up is detailed. ➤ Section 30.6 details the image processing inspection algorithm, while in ➤ Sect. 30.7 the challenges introduced by factory conditions are analysed. ➤ Section 30.8 presents the experimental results, while ➤ Sect. 30.9 concludes this chapter.

30.2 Related Vision-Based Inspection Systems

Although the application of machine vision solutions in the development of industrial systems is a well-documented topic of research, we were not able to identify any relevant work on the inspection of painted slates. However, following a detailed search on the specialised literature, we were able to find a large number of approaches that have been proposed to address the automatic inspection of ceramic tiles [3, 9–15], which are products that have several common characteristics with painted slates. Thus, although the manufacturing processes for slates and ceramic tiles are different, both products have rectangular shapes, have textured surfaces and they are transported at the inspection point via conveying means [2]. Also, the defects associated with ceramic products resemble characteristics that are common with the visual defects that are specific for painted slates, but the range of defects (types and sizes) for ceramic products is substantially reduced when compared to the large variety of slate defects.

The works in the field of ceramic tile inspection reviewed in this chapter employed a large spectrum of different imaging sub-systems and processing techniques to detect the visual defects that may be present on the surface of the product. One issue that has to be accommodated during the development phase of the developed inspection systems was the relative large range of structural and paint defects that can be encountered during the manufacturing process. In order to address this challenging scenario, the advantages associated with the multi-component inspection process start to become apparent. In this regard, the vast majority of reviewed systems approached the inspection process in a modular fashion where each component has been customised to identify a specific category of defects. To further improve the performance of the defect identification module, a critical issue was the adoption of appropriate opto-mechanical and sensorial equipment and the optimisation of the illumination set-up. Thus, most ceramic inspection systems employed line-scan cameras to minimise the spatial non-uniformities during the image acquisition process, and this option was also motivated by the fact that conveying means were invariably used to transport the products to the inspection line. (The use of standard array CCD/CMOS cameras was rarely noticed and this imaging solution has been primarily employed in the development of prototype systems.) The use of line-scan cameras has alleviated to some extent the requirement to devise complex illumination arrangements, as the light intensity has to be maintained uniform only along a narrow stripe area (that is imaged by the line-scan sensors). To this end, the ceramic inspection systems examined in our review employed either diffuse [9–12, 16] or collimated lighting set-ups [1, 2, 12, 16].

In line with the optimisation of the opto-mechanical components, the development of robust image-processing procedures to detect the visual defects proved to be a key element to the success of the overall inspection system. Our survey on ceramic tile inspection revealed that two approaches were dominant, namely, approaches that perform the identification of the visual defects using either morphological techniques [1, 13, 16–20] or inspection solutions based on texture analysis [2, 14, 21–25]. From these approaches, the morphological-based implementations proved prevalent as they offer efficient implementations that are capable of real-time operation. In this regard, Boukouvalas et al. [10] employed 1D convolvers in the horizontal and vertical directions to identify the line and spot defects on the surface of the coloured ceramic tiles. In their implementation, the filter coefficients are tuned to capture a narrow range of line widths and this fact substantially complicates the generalisation of this approach if it will be re-engineered to encompass the wide variety of visual defects present on the surface of the slates. This work was further developed by the authors in [11, 12] where they developed more involved techniques that have been applied to inspect the coloured and patterned ceramic tiles. In [11], the authors introduced a clustering approach that has been applied to segment the image data captured from a patterned tile into background and foreground information. In this study, the authors were particularly concerned with the development of an intensity correction procedure that aimed to alleviate the spatial and temporal non-constancies that were inserted by the imperfections in the illumination set-up and non-linearity response of the imaging sensor. Although the authors did not provide numerical results to quantify the accuracy of the proposed approach, this paper is a good example that illustrates the range of problems that need to be addressed in the implementation of machine vision systems. A different approach was proposed in [10] where the authors discussed the application of Wigner distributions to the inspection of patterned tiles. Although this approach is interesting as it can be applied to the inspection of a large palette of ceramic products, it has less relevance to slate inspection as the surface of the painted slates is characterised by a mild non-regularized texture. Peñaranda et al. [9] proposed an alternative approach that has potential to be applied to slate inspection. In their paper, the authors proposed an inspection algorithm that evaluates the local grey-scale intensity histograms as the main discriminative feature in the process of identifying the defective image regions. Unfortunately, when we applied this solution to slate inspection, this approach proved unreliable since we found out that the greyscale distributions vary considerable from slate to slate when they emerge from the paint line. In addition to the inter-slate greyscale intensity variation, the low statistical impact of the small defects on the profile of the intensity distributions precludes the use of a priori patterns that are able to encompass the variations within image regions of acceptable finish quality. A related approach was proposed by Boukouvalas et al. [12] where colour histograms have been employed to sample chromato-structural properties of patterned ceramic tiles. In this work, the authors attempted to address a large range of issues including the evaluation of robust similarity metrics, spatial and temporal colour normalisation and the development of a computationally efficient hierarchical colour representation scheme. A few years later, this approach was further developed by Kukkonen et al. [14] where they extracted ten local features from the local distributions calculated in the hyper-spectral domain. The main motivation to use hyper-spectral data resides in the fact that it samples in a more elaborate manner the spectral properties of the ceramic tiles than the standard RGB data [26], but the experimental results reveal that the best performance was achieved when normalised *rgb* chromaticity values were used for classification purposes.

As mentioned earlier, a distinct area of research was concerned with the application of texture-based approaches in the development of product inspection systems. Approaches that are included in this category include the work of Tobias et al. [3], Ojala et al. [27], Mäenpää et al. [21] and Mandriota et al. [25]. These methods addressed a large range of industrial inspection applications and the most promising technique when evaluated in conjunction to the inspection of painted slates was the *local binary pattern* (LBP) approach that was initially developed by Ojala et al. [27]. To this end, in a recent publication [2], we have developed an automatic slate inspection system where the texture and tonal (greyscale) information were adaptively combined in a split-and-merge strategy. The experimental results proved the effectiveness of this approach (99.14%), but the main limitation was the large computational time required to process the slate image data.

From this short literature survey, we can conclude that the development of a robust algorithm for slate inspection is a challenging task since a large number of issues relating to the opto-mechanical design and the selection of appropriate image processing procedures have to be addressed. During the development phase of the proposed prototype, we were forced to adopt a modular approach as the slate's greyscale information shows a heterogeneous distribution with substantial inter-slate intensity variations. Moreover, in our design, we had to confront other challenging issues including the depth profile variations (slate bowing) and vibrations that were introduced by the slate transport system.

30.3 Description of the Visual Defects Present on the Surface of Painted Slates

The slates are manufactured using a mix of raw materials including cement, mineral, and synthetic fibres and after they are structurally formed and dried, they are automatically painted on a high-speed line. While the primary function of the slates is to prevent water ingress to the building, they also have certain aesthetical functions. Consequently, the identification of visual defects is an important aspect of the slate manufacturing process and to answer this requirement inspection procedures are in place to ensure that no defective products are delivered to customers. Currently, the slates are manually inspected by human operators and they make a decision as to whether each individual product is defect-free and remove the slates that are not compliant to stringent quality criteria. To this end, the products meeting the quality criteria need to be correctly formed (having an undistorted rectangular shape with minor deviations from a planar surface) and their top surface should be uniformly painted in dark grey with a high gloss finish (the characteristics of the manufactured slates are listed in ➲ *Table 30.1*).

An interesting aspect associated with the manual slate inspection is to evaluate the frequency of the minor and repetitive faults that may occur when the slates are painted. This information can be used to correct the improper settings of the high-speed paint line and to allow the efficient sorting of the finite products into batches with uniform characteristics.

As the slate manufacturing process entails two major production stages, namely, the slate formation and slate painting, the visual defects can be divided into two broad categories: substrate and paint defects. Thus, substrate defects include incomplete slate formation, lumps, depressions and template marks, whereas paint defects include no paint, insufficient paint,

Table 30.1
The main characteristics of the manufactured slates

Dimension	Nominal (mm)	Tolerance (mm)
Length	600	±3
Width	300	±3
Thickness	4.0	±0.1
Local flatness profile	±0.1	±0.1
Concavity	±0.05	±5
Convexity	±0.05	±5

paint droplets, orange peel, efflorescence and paint debris. The substrate and paint defects may have arbitrary shapes and their sizes range from 1 mm^2 to hundreds of square millimetres. ➲ *Table 30.2* details a comprehensive list of most common visual defects present on the surface of defective slates, where for each defect type a brief description is provided.

To further aid the reader in the process of interpreting the variability of the visual defects that may occur during the slate manufacturing process, a number of representative paint and substrate defects are depicted in ➲ *Figs. 30.1* and ➲ *30.2*, where ➲ *Fig. 30.1a* illustrates a defect-free slate image section.

30.4 Overview of the Developed Slate Inspection System

In order to devise a flexible machine vision solution, we have adopted a modular approach in our design. The prototype slate inspection system was built to closely replicate the manufacturing environment and consists of three main modules. The flowchart detailing the logical connections between constituent modules is illustrated in ➲ *Fig. 30.3*. All modules of the developed inspection systems are controlled by an industrial PC, which is also the host of the image processing software that has been designed to perform the classification of the slates into acceptable and defective products.

The first module consists of the product feeding and the mechanical interface (slate transport system). In our implementation, we have employed a 2 m long belt-driven industrial conveyor manufactured by Noreside Conveyors and Elevators (Ireland) that has been constructed to minimise the vibrations during the slate transport. An important issue was the selection of the belt material that minimises the amount of reflective light arriving back at the camera. After a careful selection, we have chosen a belt that is coated with a matte black rubber material that has low specular properties. To minimise the lateral drifts and slate rotations, a guide was placed on one side of the conveyor.

The second module of the slate inspection system is represented by the optical and sensing equipment. In the development of our slate inspection sensor, we have used a Basler L102 2K-pixel line-scan camera which is fitted with a 28 mm standard machine vision lens (aperture set to 2.8). The line-scan camera operates at a scan frequency of 2.5 kHz. To allow a facile calibration of the sensing equipment with the illumination unit, the line-scan camera was attached to a micropositioner that allows fine adjustments to the camera view line with respect to the *xyz* coordinates. The camera set-up is illustrated in ➲ *Fig. 30.4*. The interface between the

Table 30.2
A brief description of the visual defects

No.	Defect type	Defect size	Description
1	Lumps	$2.0 < \{L,W\} < 50$ $\pm 0.1 < H < \pm 3.5$ mm	Excess material on surface. Often conical shape
2	Depressions	$2.0 < \{L,W\} < 50$ $\pm 0.1 < D < \pm 3.5$ mm	Insufficient material on surface. Often inverse conical shape
3	Holes	$\phi = 4.5$ mm	Holes are present to assist insertion of nails. Absence indicates fault
4	Incomplete slate	Any $W \neq W_{\text{expected}} \pm 3.0$ mm Any $L \neq L_{\text{expected}} \pm 3.0$ mm	W, L dimensions not conforming to production specification
5	Poor quality edge	$W = 0.2$ mm * $L > 10$ mm	Edge not straight or having a feathery feel
6	Template mark	$0.5 < \{L,W\} < 600$ mm $0.1 < \{D,H\} < 3.0$ mm	Excess material on the slate surface caused by damage to forming template
7	Template mark II	1.0 mm $< d <$ all slate	Texture variation on the slate surface, usually roughness
8	Insufficient paint	20 mm $< \{W, L\} <$ all slate	Shade variation due to insufficient paint usage
9	Missing paint	2 mm $< \{W, L\} <$ all slate	No paint or incomplete painting
10	Droplet	$2 < \{W, L\} < 15$ mm $0.05 < D < 0.5$ mm	Excess paint on surface, dried and cracked
11	Efflorescence	Area > 25 mm^2	Contaminant on surface preventing the correct adherence of paint
12	Paint debris	$2 < \{W, L\} < 50$ mm	Dried paint debris encrusted on slate surface giving a rough texture
13	Orange peel	$20 < \{W, L\} <$ all slate	Shade variation caused by overheating the slate
14	Barring	$W = 10 \pm 5$ mm $20 < L < 600$ mm	Shade variation caused by uneven heating of slate
15	Spots	$1 < \{W, L$ ø$\} < 5$ mm	Small areas presenting shade variation
16	Nozzle drip	$W = 10 \pm 5$ mm $20 < L < 600$ mm	Shade variation caused by uneven paint delivery from nozzle
17	Wax mark	$5 < \{W, L\} < 50$ mm	Splash of wax on top surface

L length, W width, ϕ diameter, D depth, H height

camera and the industrial PC computer was facilitated by a Euresys frame grabber. Another important component of the optical and sensing module is represented by the illumination set-up. In our implementation, we have designed a collimated light arrangement that relies on the strong reflective properties of the slate's surface (full details about the implementation of the illumination set-up used in our design will be provided in the next section of this chapter). The last component of this module is represented by a proximity sensor, which has the role to trigger the image acquisition process prior to the arrival of the slate at the inspection line.

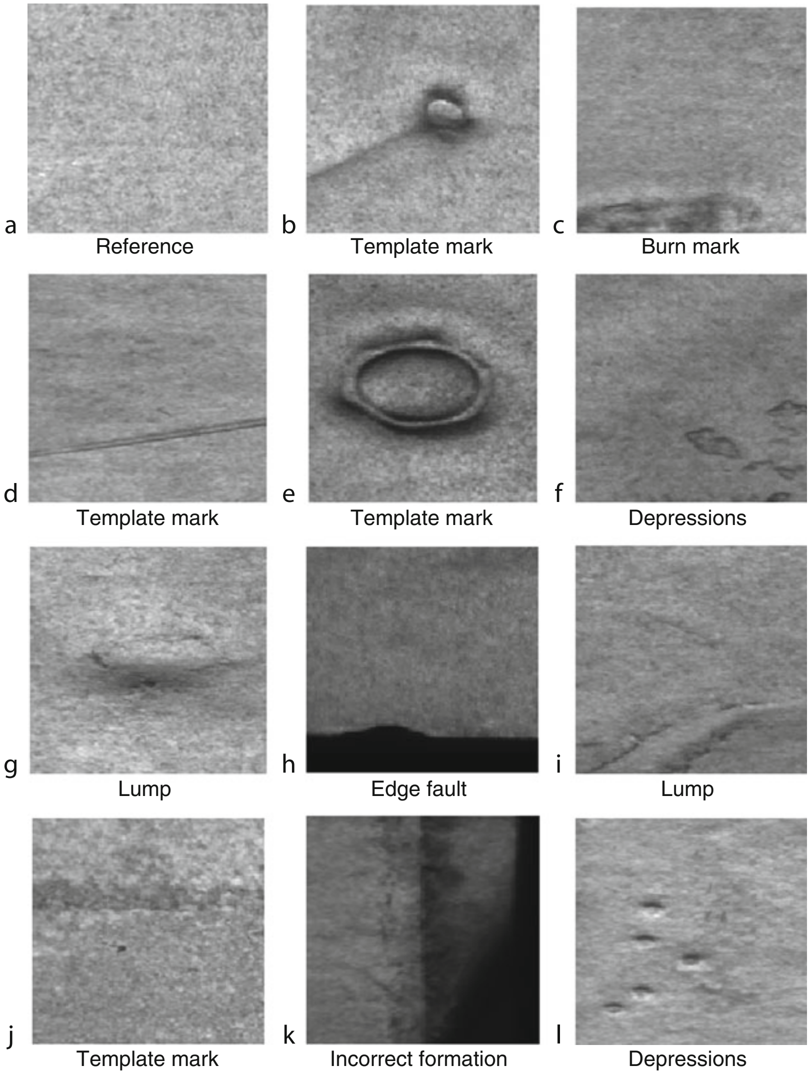


Fig. 30.1
A selection of representative substrate defects including template mark, lump, depressions, burn mark and incorrect slate formation. (a) Defect-free slate section. (b–l) Defective slate sections

The final module of the system performs the identification of the visual defects in the image data captured by the line-scan camera. While the development of a single procedure to identify all types of defects detailed in *Table 30.2* was not feasible, the proposed inspection algorithm has been also designed in a modular fashion where each component has been tailored to identify one specific category of defects (the proposed inspection algorithm will be detailed in Sect. 30.6). The developed prototype slate inspection system is illustrated in *Fig. 30.5*.

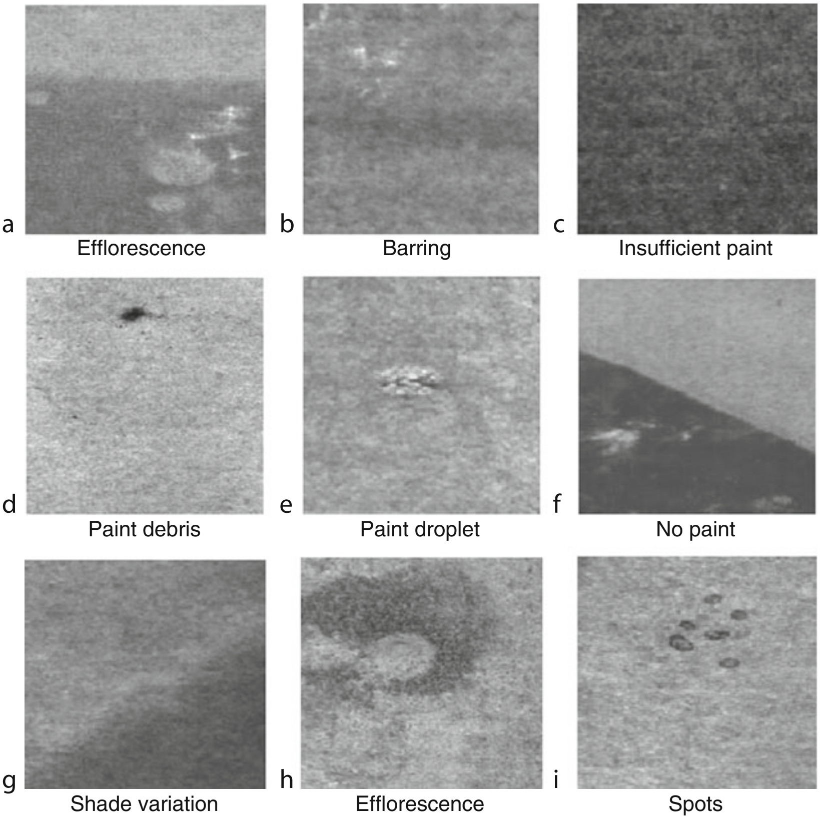


Fig. 30.2
A selection of representative paint defects including efflorescence, no paint, insufficient paint, paint droplet, shade variation, paint debris, barring and spots

30.5 Illumination Set-Up

The strategy used to image the slate relies on the strong reflective properties of the slate surface. In our experiments, we have tested several illumination set-ups involving diffuse and collimated lighting arrangements. Initially, diffuse lighting techniques were explored as this illumination technique generates a uniform wide light band that offers a facile camera calibration. Thus, the first experiments were carried out using an aperture fluorescent light system with a cylindrical focusing lens (TSI Model AFL9000), where light is dispersed equally in a 60° arc. One important aspect when designing our illumination set-up was the identification of the lamp angle of incidence that generates the strongest optical signal that is reflected back to the line-scan camera. In our experimental trials, we have discovered that the optimal optical response was achieved when the lamp angle of incidence was set at 40° and the camera was symmetrically located at 40° relative to the slate position. With this arrangement, the camera exposure time was set to 2 ms and at this setting the moving direction pixel resolution was approximately 2 mm when the conveyor was operated at

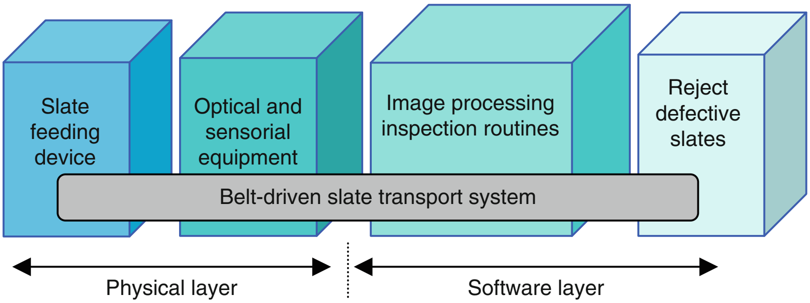

Slate feeding device
Optical and sensorial equipment
Image processing inspection routines
Reject defective slates
Belt-driven slate transport system
Physical layer
Software layer

Fig. 30.3
Overview of the designed slate inspection system

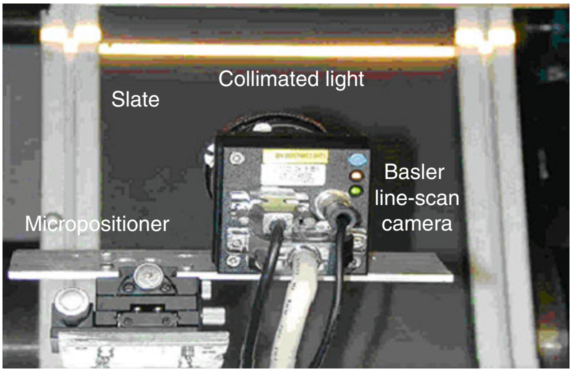


Fig. 30.4
The Basler-2K line-scan camera mounted on a *xyz* micropositioner that facilitates fine adjustments of the camera line view

production line speeds. This resolution is not sufficient to identify the small and medium-sized visual defects and our additional tests confirmed that the light intensity produced by diffuse lighting techniques is substantially lower than that required to properly image the slates at the inspection line speed. Consequently, our efforts were concentrated on the development of an alternative illumination unit that involves a customized collimated lighting approach.

The collimated illumination solution adopted in our implementation is building on the fact that the paint defects have reduced gloss levels (except paint droplets) than the slate areas of acceptable quality. Substrate defects are associated with errors in slate formation and the appearance of these visual defects also results in a reduction of light arriving at the line-scan sensing element. The illumination arrangement comprises two Fostec DCR III 150 W lamp controllers, a Fostec 30″ fibre optic light line and a cylindrical lens. A schematic description of the devised illumination set-up is illustrated in ➋ *Fig. 30.6*. The lamp angle of incidence has been set to 45° and the camera view angle was located symmetrically to 45°. The lamp intensity level was set to approximately 80% with respect to the maximum level and the camera exposure time was set to 400 μs at a scan frequency of 2.5 kHz. This setting allows us to image the slate

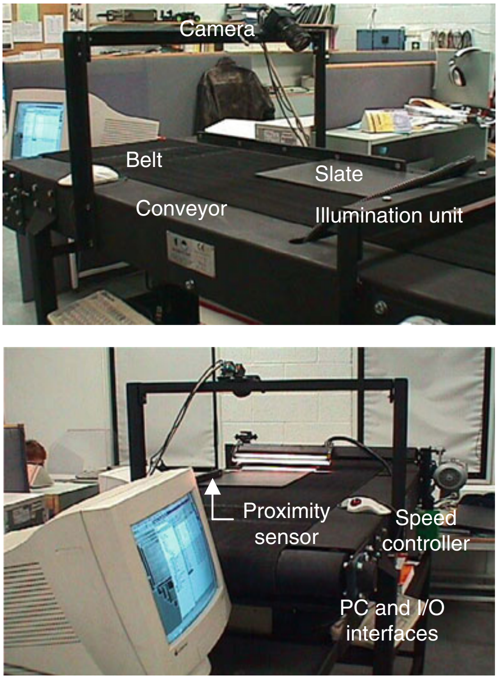


Fig. 30.5
The prototype slate inspection system [1]

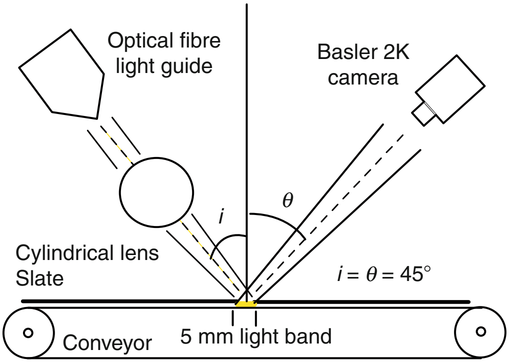


Fig. 30.6
Schematic detailing the developed illumination set-up

with a cross pixel resolution of 0.221 mm and a moving pixel resolution of 0.244 mm. This pixel resolution is sufficient to properly image the entire range of defects that are present on the surface of the painted slates.

Although the illumination set-up detailed in ➜ *Fig. 30.6* generates sufficient illumination levels required for image acquisition at production line speeds, one issue that we encountered was the narrow band of the focused collimated light. Indeed, when we positioned the lens to attain perfectly focused collimated light, the resulting light band was only 5 mm wide (see ➲ *Fig. 30.6*) and the process required to align the camera line view for a distance of 30″ requires adjustments with a precision of less than 1° ($\delta\theta < 1°$). To facilitate such adjustments, we mounted the camera on a micropositioner as indicated in ➲ Sect. 30.4. However, the alignment of the camera line view to the longitudinal axis of the light band was not the only issue we had to confront during the development phase of the illumination set-up. One of the most challenging problems was the variation in depth profiles across the slate due to slate bowing that is sometimes associated with slates that have acceptable quality with respect to inspection criteria. The small slate bowing was found to range from negligible to 5 mm over the slate length and up to 2 mm along the slate width. Although the variation in depth profile does not affect the functionality of the product, it has introduced a substantial challenge that has to be accommodated by the proposed illumination solution. While the depth profile non-uniformities raise and lower the absolute position of the projected light band with respect to the camera line view, this practically compromises the image acquisition process. To give some insight into the problem generated by slate bowing, if we consider that the incidence of the light band and the camera view angle are set symmetrically to 45° as illustrated in ➲ *Fig. 30.6*, if the slate raises the light band with 1 mm, then the reflected light is shifted to 1.5 mm in opposite direction with respect to the camera view line. Since the slate possesses some elasticity, a pneumatic solution that forces the slate into a uniformly planar shape during inspection would be feasible. However, there are several disadvantages associated with such mechanical solutions that can be used to force the slate into the nominally flat shape. The first is related to the fact that the painted surface can be easily damaged when the pneumatic device is operated at high production line speeds, and secondly the development of such mechanical system would be problematic taken into consideration the non-smooth characteristics of the underside of the slate. As a mechanical solution to force the slate into a uniform flat position proved unfeasible, we elected to defocus the lens that is used to collimate the light generated by the Fostec fibre optic light line. This produces a wider light band but this is achieved at the expense of reduced collimation. To compensate for the reduced intensity levels caused by de-collimation, we used the spare capacity in the lamp controller. Following a simple trigonometric calculation, the lens has been defocused to generate a 25 mm wide light band, an increase that is sufficient to accommodate the slate bowing effects in the range [0, 5 mm] and the vibrations caused by the slate transport system.

The last issue that we confronted during the development of the illumination set-up was the implementation of a software routine that was applied to compensate for the non-linearities in the horizontal profile (longitudinal axis) of the light band. As the fibre optic light line is 30″ long, the profile of the light band shows stronger illumination levels towards the central position when compared to the intensity level supplied at the left and right extremities of the slate. This is illustrated in ➲ *Fig. 30.7a* where the unprocessed horizontal light profiles for three reference slates are depicted. To compensate for this uneven response of the illumination unit, we have subtracted the mean intensity level (calculated from all reference slates that are used to train the system) from the unprocessed data and the obtained results are averaged with respect to each pixel position to compute the light compensation map. This light compensation map encodes the deviations from the expected mean intensity level, and it is directly used to linearise the unprocessed horizontal profile (see ➲ *Fig. 30.7b*). The application of the horizontal profile linearization reduced the light variation in the horizontal direction from 20% to 2.5%. The linearisation of the response of the illumination unit in the moving direction was deemed unnecessary, as the light intensity variation in the vertical axis shows negligible variation due to de-collimation.

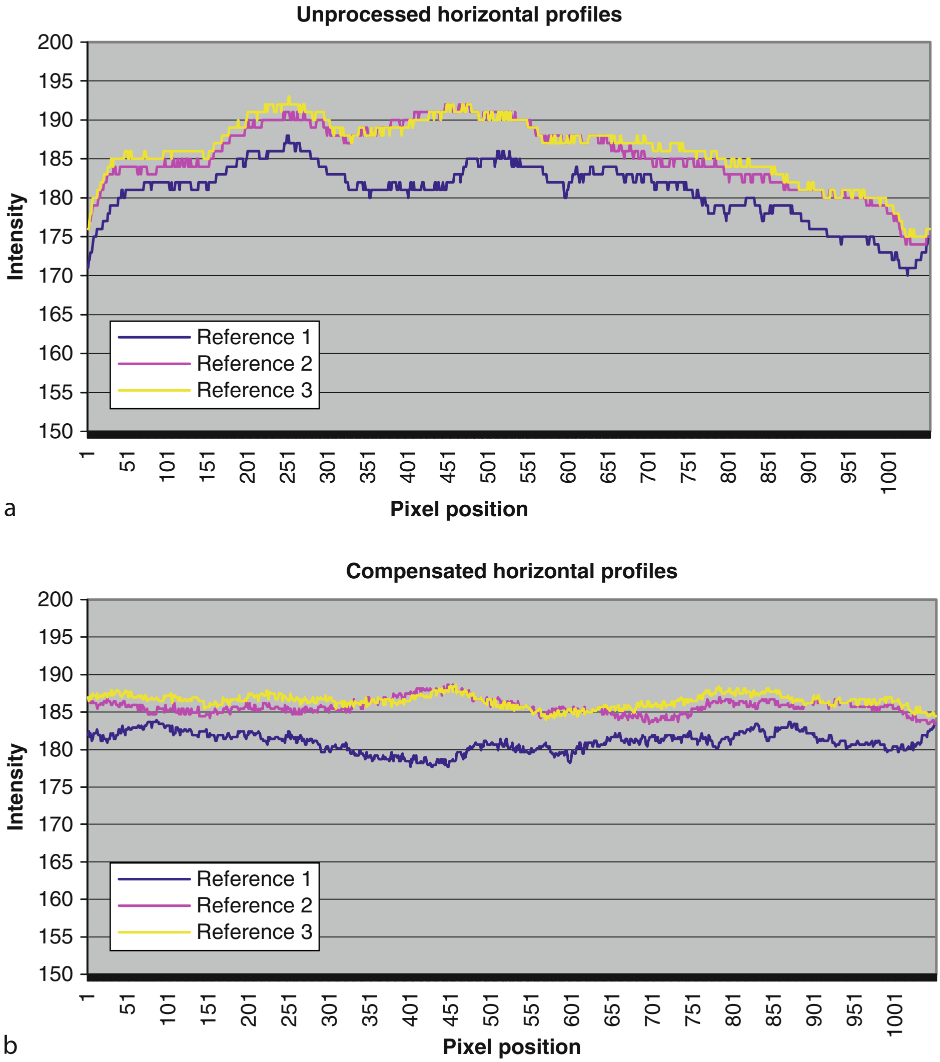


Fig. 30.7

Compensation for the non-linearities in the response of the light unit along the horizontal direction. (a) Unprocessed horizontal light profiles. (b) Compensated horizontal light profiles

30.6 Slate Inspection Algorithm

As indicated in Sect. 30.4, the slate inspection algorithm has been designed in a modular fashion where each component has been optimised to identify one category of visual defects. This approach was appropriate as the modular architecture allows facile system training and more importantly the parameters that need to be optimised for each component can be easily adjusted if changes are made on the specification of the finite product. To this end, the proposed algorithm involves five distinct computational strands that were tailored to address specific inspection tasks. Full details about each image processing component of the developed inspection algorithm will be provided in the remainder of this section.

30.6.1 Segmentation of Slate from Image Data and the Nail Hole Verification Procedure

The first step of the slate inspection algorithm involves the segmentation of the slate from the image data. The identification of the slate boundaries was facilitated by cutting slots in the conveyor base and ensuring that the width of the belt is less than that of the slate. This design choice was implemented to ensure that no light arrives back at the line-scan camera when the slate is not imaged and to provide a sharp rise in the optical signal when the slate arrives at the inspection line. Thus, a simple threshold operation is sufficient to robustly segment the slate data and the segmentation process is followed by the identification of the edges and the four corners of the slate. To avoid computationally expensive procedures required to optimise the image acquisition process, a proximity sensor was employed to activate the line-scan sensor prior to the arrival of the slate at the inspection point.

Once the corners of the slate are identified, the next operation verifies if the nail holes are correctly positioned with respect to slate boundaries. The nail hole detection involves a threshold-based segmentation technique since the intensity of the optical signal associated with the nail hole is substantially lower that that given by the slate surface. The developed nail hole checking procedure starts with the identification of the hole that is located on the left-hand side of the slate surface. The detected *xy* co-ordinates of the nail hole are compared to pre-recorded values and if the procedure passes inspection conditions then the algorithm proceeds with the verification of the remaining two nail holes that are located close to the right-hand side top and bottom slate corners. If the nail checking procedure triggers the absence of a nail hole or an incorrect nail hole placement with respect to slate boundaries, the slate is classed as defective and put aside for rejection or re-work. If the nail holes are correctly positioned, then the algorithm fills them with adjacent slate surface pixel information to prevent the identification of visual faults when the slate data is subjected to the automatic inspection process. ➲ *Figure 30.8* describes graphically the nail verification procedure.

30.6.2 Identification of the Visual Defects

Prior to the development of the computational modules for visual defect identification, we have analysed the grey level signals associated with reference and defective slates. The experimental trials indicated that the mean grey levels associated with acceptable slates from similar production batches can vary up to 20 grey levels where the overall mean value is 167. To clarify the source of this greyscale variation among reference (non-defective) slates, we evaluated the intensity data using different image acquisition approaches. From these experiments, we concluded that these variations were not caused by the imperfections in the sensing and optical equipment but rather due to an acceptable variation in the slate surface paint colour. Thus, the image processing procedures that will be investigated have to be sufficiently robust to accommodate this level of inter-slate greyscale variation. In order to identify the computational components of the inspection algorithm, we have analysed the grey level signals associated with paint and substrate defects and a summary of the test results is shown in ➲ *Table 30.3*.

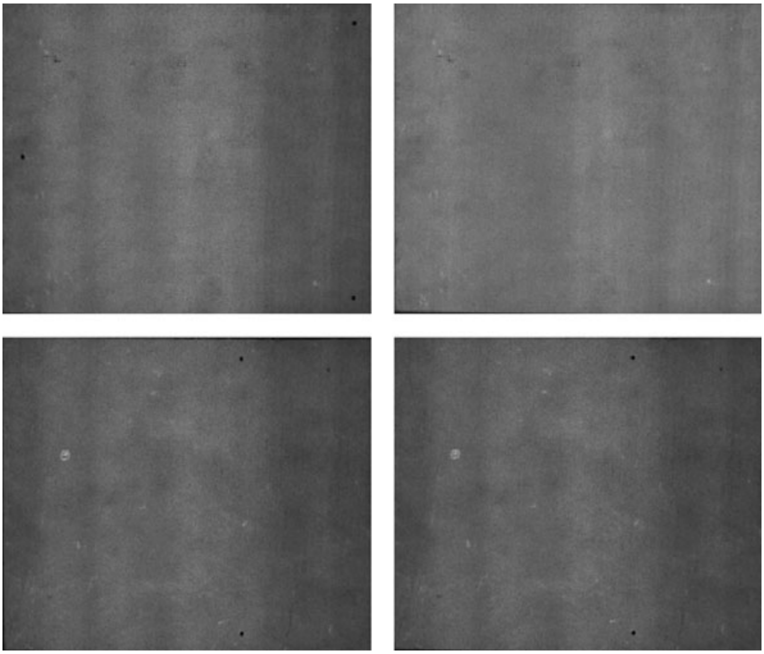


Fig. 30.8
Nail hole verification procedure. (*Top row*) Slate that passes the nail hole verification test. (*Bottom row*) Slate that fails the nail hole inspection procedure. In the last column, the images resulting after the nail hole inspection procedure are displayed. Note that the image that passes the nail hole verification test has the nail hole areas filled with pixels information from adjacent slate surface data to be ready for defect inspection

This analysis allowed us to evaluate the impact of the visual defects on the overall intensity data distribution. Our study revealed that the vast majority of defects have a negligible influence on the overall slate intensity data distribution, and this finding clearly indicates that it would be desirable to inspect the slate image in small subsections, as the relative impact of the defect on the grey level statistics is substantially increased. To this end, the slate image was divided into segments of 128 × 128 pixels and each segment needs to be individually processed.

Based on the analysis of the greyscale values associated with reference and defective image sections, we have devised an algorithm which consists of four distinct computational strands – in this approach each component has been designed to strongly respond to a narrow category of visual defects. The block diagram of the developed slate inspection algorithm is depicted in *Fig. 30.9* and the main components can be summarised as follows:

- Global mean threshold method
- Adaptive signal threshold method
- Labelling method
- Edge detection and labelling method

Each component depicted in *Fig. 30.9* is applied to each 128 × 128 segment of the slate image and the results are compared to experimentally determined thresholds. Each component of the slate inspection will be detailed in the remainder of this section.

Table 30.3
The grey level values associated with a representative sample of visual defects

No.	Defect type	Overall slate greyscale mean	Defect greyscale value
1	Lump 1	186	150
2	Lump 2	180	140
3	Lump 3	184	125
4	Lump 4	182	135
5	Lump 5	191	160
6	Insufficient paint	181	136
7	Paint debris	172	80
8	Paint droplet	179	150
9	Spots	184	150
10	Barring	181	166
11	Orange peel	181	127
12	Efflorescence 1	181	135
13	Efflorescence 2	175	150
14	No paint	180	100
15	Shade variation	180	150
16	Template mark 1	177	145
17	Template mark 2	189	140
18	Template mark 3	179	120
19	Template mark 4	180	150
20	Template mark 5	181	150

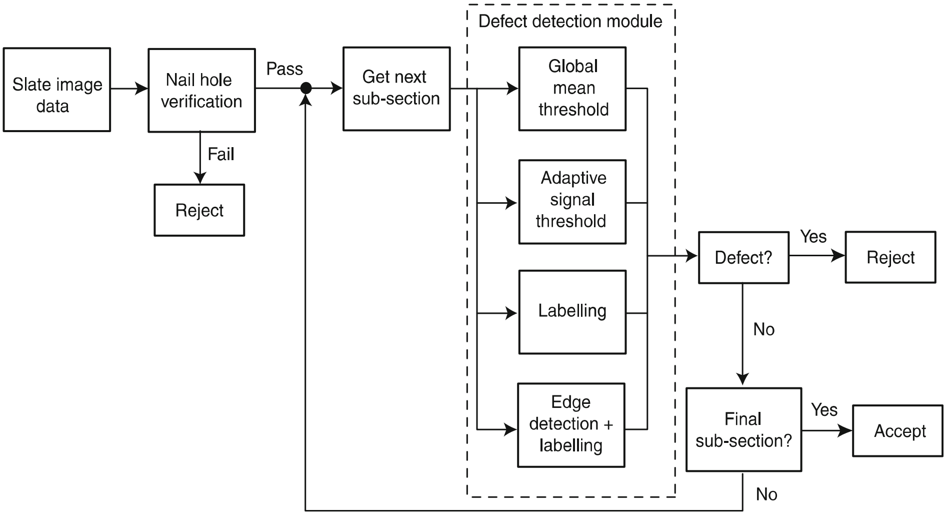


Fig. 30.9
Overview of the slate inspection algorithm

30.6.3 Global Mean Threshold Method

This component has been developed to evaluate the mean grey level of the image section with the aim of identifying the gross defects that may be present on the surface of the slate. In this regard, the mean value for each subsection is compared to an experimentally determined mean value and this approach proved to be effective in detecting a range of defects including missing paint, orange peel, efflorescence, insufficient paint and severe shade variation.

In spite of its simplicity, this method proved to be the most reliable in detecting defects that cover large areas of the slate surface, defects that have a dramatic impact on the greyscale distribution when evaluated at local level (see ➔ *Fig. 30.10*). To increase the robustness of the proposed method, a global mean threshold has been employed to identify defects such as missing paint and orange peel, and grade-specific thresholds were experimentally determined for the identification of low contrast defects such as efflorescence, insufficient paint and shade variation.

30.6.4 Adaptive Signal Threshold Method

Our experimental measurements indicated that the slate texture induces a greyscale variation for non-defective image sections in the range [mean − 30, mean + 30]. When analysing the 128 × 128 image sections for visual defects, to avoid the occurrence of false positives we need to locate the paint and substrate defects outside the range [mean − 40, mean + 60]. By thresholding the image data with these values, a relative large category of visual defects can be robustly identified [1]. The defects detected by this method include missing paint (where less than 50% of the image section is defective), localised paint faults such as paint debris, droplets, spots and structural defects generated by an improper slate formation.

This method has not been found useful in detecting low-contrast visual defects such as barring and elongated depressions whose grey levels are within the greyscale variation generated by the slate texture. Representative paint and substrate defects that are detected by the adaptive signal threshold method are depicted in ➔ *Fig. 30.11*.

30.6.5 Labelling Method

The aim of this method is to identify the visual defects that cover a relative small area with respect to the overall size of the analysed 128 × 128 image section. This method consists of several steps that are illustrated in ➔ *Fig. 30.12*.

The first step of the labelling method converts the image corresponding to the analysed slate section into the binary format by applying a local adaptive threshold operation [28]. The aim of this step is to identify the relative bright and dark regions with respect to the local grey mean value. The second step involves the application of a binary opening operation with the aim of removing the small unconnected regions that are caused by the non-regularized slate texture and the image noise. The third step involves the application of the labelling by area operation [28, 29] to provide a unique label for each disjoint white blob in the image resulting from the morphological opening step. Since the small blobs are often associated with the slate texture, the largest blobs are of interest as they are generally caused by visual defects. Thus, to reduce the impact of the randomly distributed blobs that are generated by the slate texture, we compute the area of the ten largest blobs and this feature is used to discriminate between

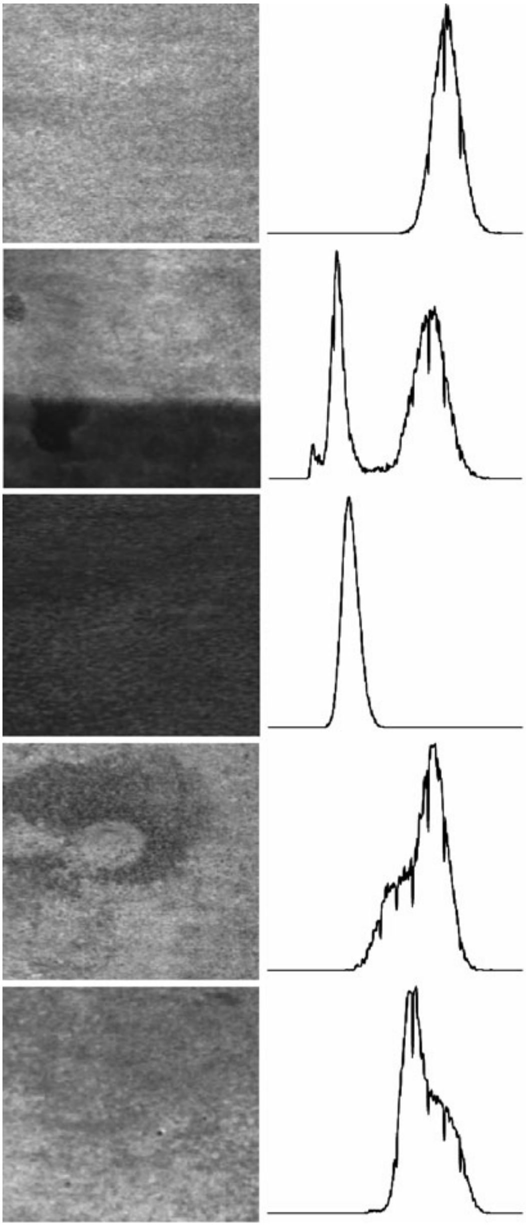


Fig. 30.10

Gross defects identified by the global mean threshold method. (*Left*) Images showing a reference slate section and slate sections exhibiting gross visual defects. (*Right*) The corresponding greyscale distributions. From top to bottom, reference (non-defective) slate section, slate section detailing an insufficient paint defect, slate section detailing an orange peel defect, slate section detailing an efflorescence defect and slate section detailing a shade variation defect

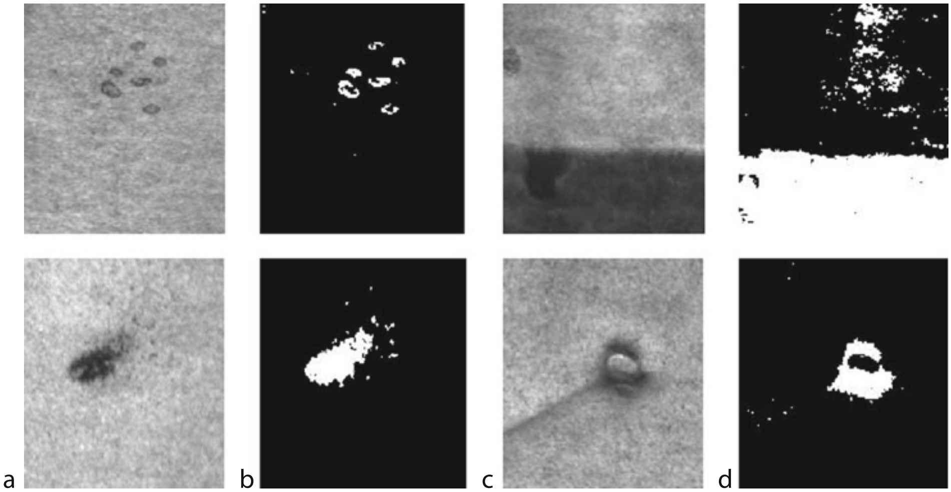


Fig. 30.11
Visual defects identified by the adaptive signal threshold method. (a, c) Defective slate sections. (b, d) Resulting images after the application of the adaptive signal threshold method. *Top row* – paint defects such as paint droplets and insufficient paint. *Bottom row* – substrate defects such as burn mark and template mark

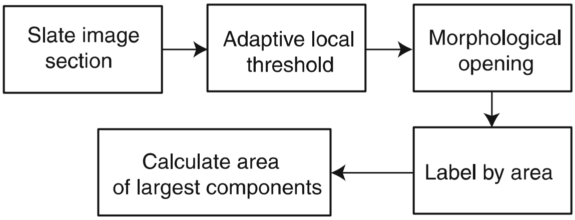


Fig. 30.12
Overview of the labelling method

reference and defective image sections. This method proved effective in identifying high-contrast defects such as droplets, paint debris, spots, efflorescence and template marks. A small collection of visual defects that are identified by the labelling method is depicted in *Fig. 30.13*.

30.6.6 Edge Detection Method

The last component of the inspection algorithm has been designed to specifically detect the narrow elongated defects, as the previously described methods were not sufficiently accurate in identifying this type of slate faults. As this defect identification method is based on the application of first-order derivative operators (Sobel edge detector), prior to edge extraction the image data is subjected to median filtering. The experimental measurements carried out

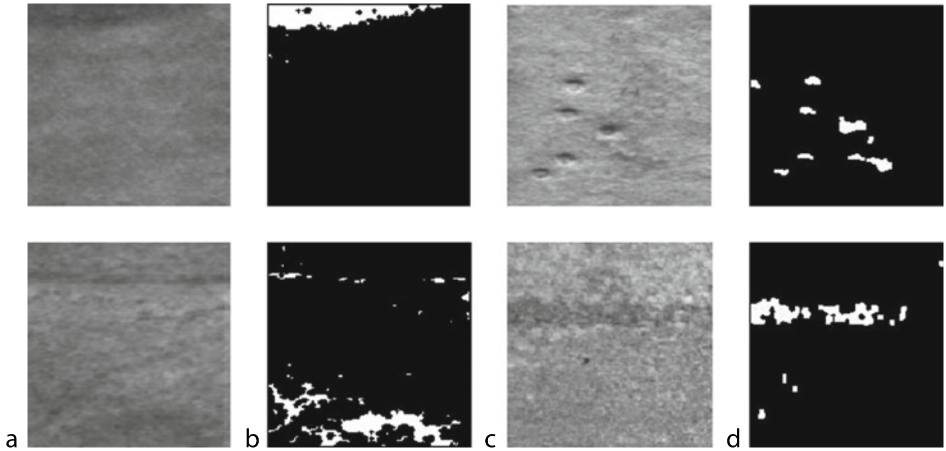
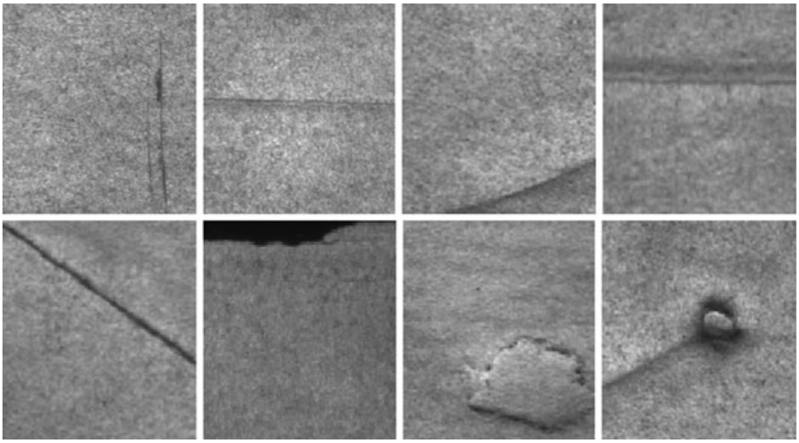


Fig. 30.13
Visual defects identified by the labelling method. (a, c) Defective slate sections. (b, d) Resulting images after the application of the labelling method. ***Top row*** **– paint defects such as shade variation and paint droplets.** ***Bottom row*** **– substrate defects (template marks)**

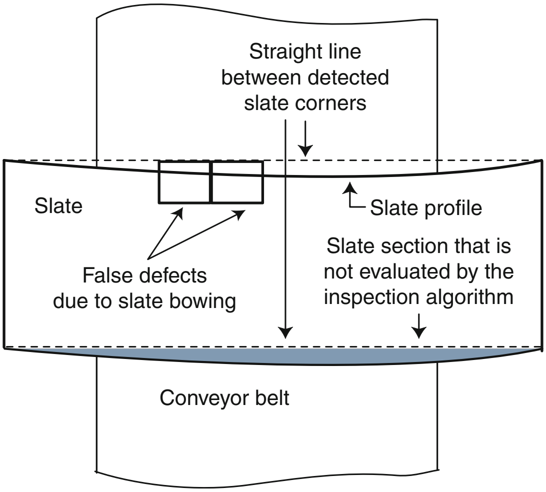
during the development phase revealed that narrow defects such as depressions and elongated substrate marks have a shallow intensity profile and as a result gaps are present in the resulting edge structure. To compensate for this issue, a morphological closing operator is applied and the resulting data is labelled by area for further analysis. Similar to the approach used for labelling method, the largest ten components were retained and their collective area is used as a discriminative feature to grade the slate section under analysis as acceptable or defective. *Figure 30.14* depicts some representative defects that are identified by the edge detection method.

30.7 Effect of Slate Bowing on Slate Inspection

The slate inspection algorithm detailed in Sect. 30.6 performs the identification of the visual defects on each 128 × 128 segment of the slate image. To facilitate the tiling of the slate image into sub-sections, the slate was approximated with a rectangle that is obtained by connecting the corners of the slate that have been identified during the slate segmentation process (see Sect. 30.6.1). This procedure worked very well when applied to flat slates, but generated false triggers when applied to slates that are affected by depth profile variations (slate bowing). These false positives are caused by the displacements between the straight edge obtained by connecting the slate corners and the curved edge of the imaged slate. This slate edge positioning errors lead to an incorrect image tiling and this generates situations when the 128 × 128 image segments include non-slate image data (background information), while on the opposite side of the slate there are sections of the slate that are not investigated by the inspection algorithm (see the shaded area in *Fig. 30.15*). This is illustrated in *Fig. 30.15* where the problems caused by slate bowing are illustrated. An example that shows the occurrence of false positives due to slate bowing is depicted in *Fig. 30.16*.



◘ Fig. 30.14
Visual defects identified by the edge detection method. (*Top images*) Paint defects. (*Bottom images*) Structural defects



◘ Fig. 30.15
False defects generated by slate bowing

To address this issue, the procedure applied to identify the slate edges has been modified in order to accommodate the deviations that occur from the straight edge when imaging slates that are affected by depth profile variations. In this regard, the edge that is generated by connecting the slate corners is divided into shorter lines with 30 mm length and the location of these short segments is identified with respect to the segmented slate data. Using this approach, the image tiling process ensures that the image sub-sections are placed adjacent to these segments and as a result no slate data will be missed by the inspection algorithm or false positives generated by the incorrect positioning of the image sub-segments with respect to the real edge of the slate.

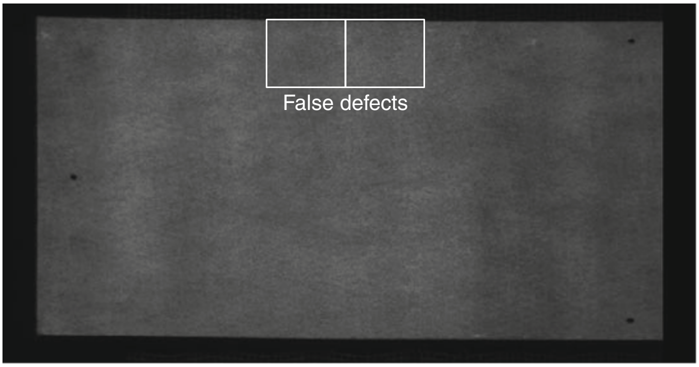


Fig. 30.16
Example illustrating the false defects caused by slate bowing effects

30.8 Experimental Results

The conveyor speed was set at 38 m/min and the camera exposure was set to 400 μs giving a scan frequency of 2.5 kHz. The variability of the conveyor speed has been measured by imaging a reference slate in succession and the experimental results indicate that the variation profile of the conveyor speed induced a 0.8% variation in the pixel resolution in the moving direction [1]. The variation in the pixel resolution due to inconstancies in the conveyor speed has negligible effects on the image acquisition process and it was found that it has no negative effects on the inspection results.

Since the inspection algorithm detailed in Sect. 30.6 entails the optimisation of a large number of threshold parameters, the system has been trained using 400 slates. The developed slate inspection system has been tested on more than 300 unseen slates that are sampled from different production batches. The slates used in the experimental activity were graded into acceptable and defective units by an experienced operator based on a comprehensive visual examination. The proposed system was able to correctly grade the acceptable slates with an accuracy of 99.32% (148 slates) and the defective slates with an accuracy of 96.91% (162 slates). A detailed performance characterisation is provided in *Table 30.4*. To qualitatively and quantitatively illustrate the performance attained by the proposed system, in *Table 30.4* the detection rate for each category of defects listed in *Table 30.2* is provided.

As indicated in *Table 30.4*, the lowest performance of the slate inspection system was achieved in the identification of depressions, which is a subclass of substrate defects. The system failed to identify a defective slate containing a shallow depression that was positioned parallel to the moving direction axis. To fully clarify the impact of these defects on the inspection results, we imaged the slate with the short edge facing forward (slate rotated with 90°). Using this slate orientation, the system was able to identify the defect as it produced sufficient contrast when compared to adjacent non-defective areas of the slate. We envision that the detection rate for this type of substrate defect can be substantially increased if the slate is imaged in two orthogonal directions. The inspection system was also not able to identify two small template marks with an area of approximately 1 mm^2. These defects were missed as their statistical impact on the greyscale distribution of the 128 × 128 slate segment was very limited and the relaxation of threshold conditions for labelling method was not an option since it will

Table 30.4
Summary of detection rate for each category of visual defects

Type	Undetected	Detected	Accuracy (%)
Missing paint	0	4	100
Insufficient paint	0	9	100
Efflorescence	0	15	100
Shade variation	0	12	100
Nozzle drip	0	14	100
Droplets	0	8	100
Dust	0	4	100
Wax	1	7	87.5
Template marks	2	33	94.28
Template marks II	0	5	100
Lumps	0	13	100
Depressions	1	3	75
Bad edge	0	18	100
Misc. types	1	12	92.30
Total	5	157	96.91

lead to a substantial increase in the number of false positives. Typical inspection results are depicted in *Fig. 30.17* where white boxes are used to mark the defective image sections. Additional results are illustrated in *Fig. 30.18* where reference and defective image sections are depicted and the images resulting after the application of the four computational strands of the inspection algorithm are also shown (for the sake of completeness, the pixel count for each method of the inspection algorithm is provided).

30.9 Conclusions

The major objective of this chapter was the introduction of a machine vision system that has been developed for the automated inspection of painted slates. Since the development of an industry-compliant inspection system requires the identification of optimal solutions for a large spectrum of problems relating to opto-mechanical and software design, in this chapter we provided a comprehensive discussion about each component of the system and we emphasized the difficulties that we had to confront during the development phase of the inspection sensor. The illumination set-up proved in particular challenging as we had to devise a solution that is able to provide sufficient illumination intensity that allows imaging the slate at production line speeds, but at the same time it is able to accommodate non-idealities such as slate depth profile variations and vibrations that are introduced by the slate transport system.

The experimental data indicates that automatic slate inspection can be achieved and the installation of the proposed inspection system in a manufacturing environment is a realistic target. The evaluation of the inspection prototype in a factory-style manner was an important part of the development process, as it allowed us to sample, address and optimise a large

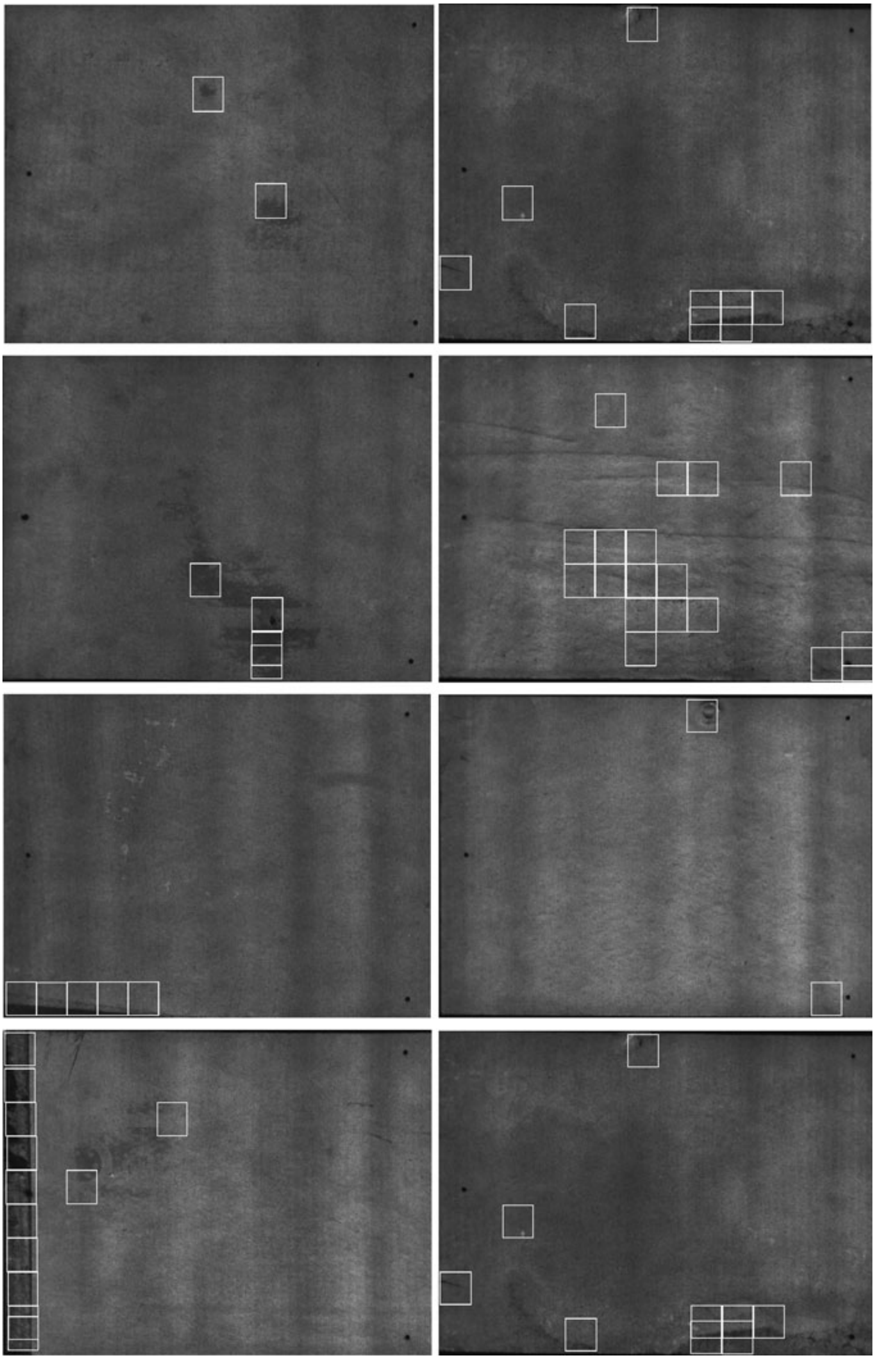


Fig. 30.17

Identification of the visual defects on a selection of defective slates that exhibit a large variety of paint and substrate defects

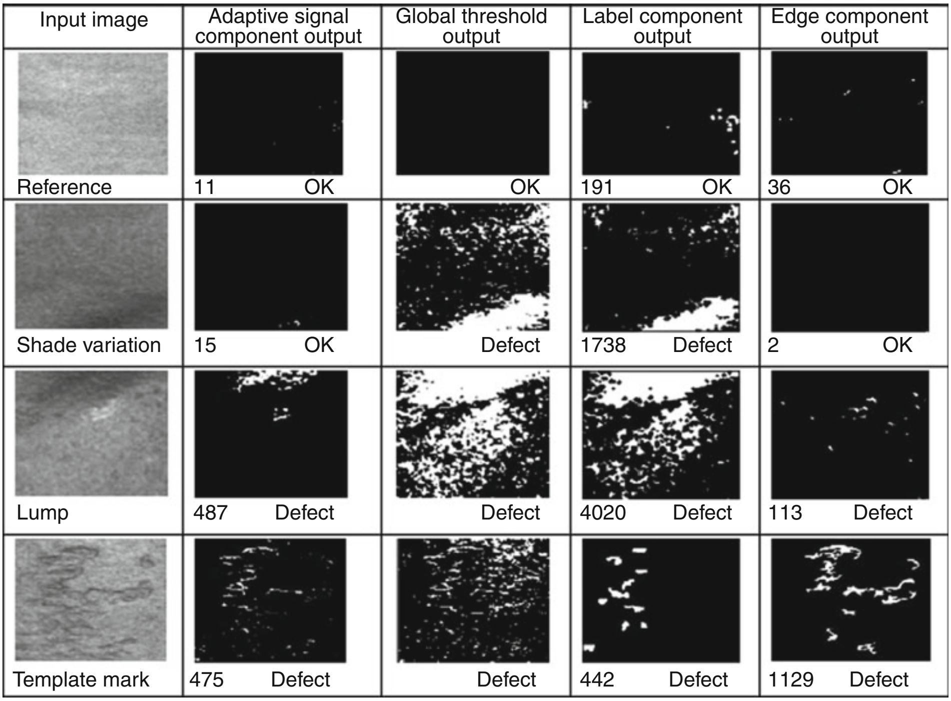


Fig. 30.18
Output of the inspection algorithm for a small set of reference and defective slate sections

number of hardware and software design choices that are required for robust slate inspection. The proposed inspection system attained 96.91% overall accuracy in detecting the visual defects present on the slate surface and this level of performance indicates that the devised machine vision solution can be considered as a viable option in replacing the existing manual inspection procedure.

Acknowledgements

We would like to acknowledge the valuable contribution of our industrial partner during the development phase of our slate inspection system and Enterprise Ireland for its generous financial support.

References

1. Ghita O, Whelan PF, Carew T (2006) A vision-based system for inspecting painted slates. Sensor Rev 26(2):108–115
2. Ghita O, Whelan PF, Carew T, Nammalwar P (2005) Quality grading of painted slates using texture analysis. Comput Ind 56(8–9):802–815
3. Tobias O, Seara R, Soares F, Bermudez J (1995) Automated visual inspection using the co-occurrence approach. In: Proceedings of the IEEE midwest symposium on circuits and systems, Rio de Janeiro, pp 154–157

4. Porter AL, Rossini FA, Eshelman BJ, Jenkins DD, Cancelleri DJ (1985) Industrial robots, a strategic forecast using the technological delivery system approach. IEEE Trans Syst Man Cybern 15(4):521–527
5. Batchelor BG, Waltz FM (2001) Intelligent machine vision: techniques, implementations and applications. Springer, New York. ISBN: 978–3540762249
6. Ghita O, Whelan PF (2003) A bin picking system based on depth from defocus. Mach Vis Appl 13(4):234–244
7. Batchelor BG, Whelan PF (1997) Intelligent vision systems for industry. Springer, London. ISBN: 978–3540199694
8. Newman T, Jain A (1995) A survey of automated visual inspection. Comput Vis Image Underst 61(2): 231–262
9. Peñaranda J, Briones L, Florez J (1997) Colour machine vision system for process control in ceramics industry. Proc SPIE 3101:182–192
10. Boukouvalas C, De Natale F, De Toni G, Kittler J, Marik R, Mirmehdi M, Petrou M, Le Roy P, Salgari R, Vernazza F (1997) An integrated system for quality inspection of tiles. In: Proceedings of the international conference on quality control by artificial vision, France, pp 49–54
11. Boukouvalas C, Kittler J, Marik R, Petrou M (1997) Automatic color grading of ceramic tiles using machine vision. IEEE Trans Ind Electron 44(1): 132–135
12. Boukouvalas C, Kittler J, Marik R, Petrou M (1999) Colour grading of randomly textured ceramic tiles using colour histograms. IEEE Trans Ind Electron 46(1):219–226
13. Fernandez C, Fernandez J, Aracil R (1998) Integral on-line machine vision inspection in the manufacturing process of ceramic tiles. Proc SPIE 3306
14. Kukkonen S, Kälviäinen H, Parkkinen J (2001) Color features for quality control in ceramic tile industry. Opt Eng 40(2):170–177
15. Smith ML, Stamp RJ (2000) Automated inspection of textured ceramic tiles. Comput Ind 43(1): 73–82
16. Carew T, Ghita O, Whelan PF (2003) Exploring the effects of a factory-type test-bed on a painted slate detection system. In: International conference on mechatronics, Loughborough, pp 365–370
17. Müller S, Nickolay B (1994) Morphological image processing for the recognition of surface defects. Proc SPIE 2249:298–307
18. Sternberg SR (1985) A morphological approach to finished surface inspection. In: Proceedings of the IEEE international conference on acoustics, speech, signal processing, Tampa, pp 462–465
19. Kuleschow A, Münzenmayer M, Spinnler K (2008) A fast logical-morphological method to segment scratch-type objects. In: Proceedings of the international conference on computer vision and graphics, Warsaw, pp 14–23
20. Costa CE, Petrou M (2000) Automatic registration of ceramic tiles for the purpose of fault detection. Mach Vis Appl 11(5):225–230
21. Mäenpää T, Viertola J, Pietikäinen M (2003) Optimising colour and texture features for real-time visual inspection. Pattern Anal Appl 6(3):169–175
22. Kyllönen J, Pietikäinen M (2000) Visual inspection of parquet slabs by combining color and texture. In: Proceedings of the IAPR workshop on machine vision applications, Tokyo, pp 187–192
23. Latif-Amet A, Ertüzün A, Erçil A (2000) Efficient method for texture defect detection: sub-band domain co-occurrence matrices. Image Vis Comput 18(6–7):543–553
24. Xie X (2008) A review of recent advances in surface defect detection using texture analysis techniques. Electron Lett Comput Vis Image Anal 7(3):1–22
25. Mandriota C, Nitti M, Ancona N, Stella E, Distante A (2004) Filter-based feature selection for rail defect detection. Mach Vis Appl 15(4):179–185
26. Picon A, Ghita O, Whelan PF, Iriondo PM (2009) Fuzzy spectral and spatial feature integration for classification of non-ferrous materials in hyperspectral data. IEEE Trans Ind Inf 5(4):483–494
27. Ojala T, Pietikäinen M, Mäenpää T (2002) Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans Pattern Anal Mach Intell 24(7):971–987
28. Whelan PF, Molloy D (2000) Machine vision algorithms in Java, techniques and implementation. Springer, London. ISBN: 1-85233-218-2
29. Davies ER (1997) Machine vision: theory, algorithms, practicalities, 2nd edn. Academic, San Diego. ISBN: 0-12-206092-X

31 Inspecting Glass Bottles and Jars

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_31,

31.1 Introduction

Despite the popularity of plastic containers, glass bottles and jars remain important for sealing and preserving food products, toiletries and pharmaceuticals. The raw materials are abundant and inexpensive. Although it is brittle, glass is strong and chemically inert, resistant to microbial attack and impervious to water, oil, alcohol, etc. It can be moulded to form a wide variety of useful shapes. It also has the attractive feature of being transparent, which allows the contents to be presented in a visually appealing way. It can also be coloured for cosmetic effect.

Glass is ubiquitous, yet it is potentially dangerous, particularly if it breaks and the resulting shards are swallowed. There is a wide range of defects in moulded glassware used for food and drink (bottles, jars, cheap tumblers, dishes, bowls, etc.) that should be considered:

- Malformations (These are often due to moulding faults as when molten glass is injected into the mould when it is too hot, or sticks to the mould as it opens.)
- Cracks, checks and crizzles (The latter do not penetrate right through the glass wall.)
- Inclusions (These might typically be stone-like fragments of refractory materials that have broken off the lining of the crucible holding the liquid glass.)
- Air bubbles
- Loose glass fragments (For example, a vial breaks during manufacture and scatters flakes and slivers, contaminating nearby containers.)
- Spikes (These are "stalagmites" of glass, found typically in jars and other items that are made by pressing or a *press-blow* manufacturing process. They occur on the base or side walls of a jar.)
- Bird-swings (These are horizontal filaments of glass that span across the inside of a bottle. Physically, they resemble stalagmites and stalactites that have joined together. They result from a faulty *blow-blow* manufacturing process. They occur only on the side walls of a bottle.)

This is a much simplified catalogue of defects that are known to arise in everyday bottles, jars, tumblers. Loose glass fragments, in the form of thin strands, flakes and slivers can occur virtually anywhere. In this short chapter we can only consider a few of these defects; our intention is to highlight the potential for applying Machine Vision, even in an area, such as this where the product is often difficult to illuminate and view. Some defects are euphemistically called *critical* in the glass industry: in lay terms *dangerous*!

Many of the faults that occur in glassware can only be detected properly by optical means. They cannot, for example, be detected by touch. However, a modern manufacturing line produces bottles at a speed that exceeds a human being's ability to inspect them reliably. Recall the cartoon in ➋ Chap.1, showing a (male) human inspector distracted by a fair damsel. (Female inspectors are also susceptible to distraction!) Thus, there is a need for automated visual inspection equipment that can meet this need. In this chapter we briefly explore the subject and will consider some of the topics that a vision engineer designing an inspection system for glass-ware must address.

31.2 Lighting and Viewing

There are several possible methods for inspecting glassware:

- Back-illumination. The light source, bottle and camera are in line.
- Dark-field illumination (➋ *Fig. 31.1*).

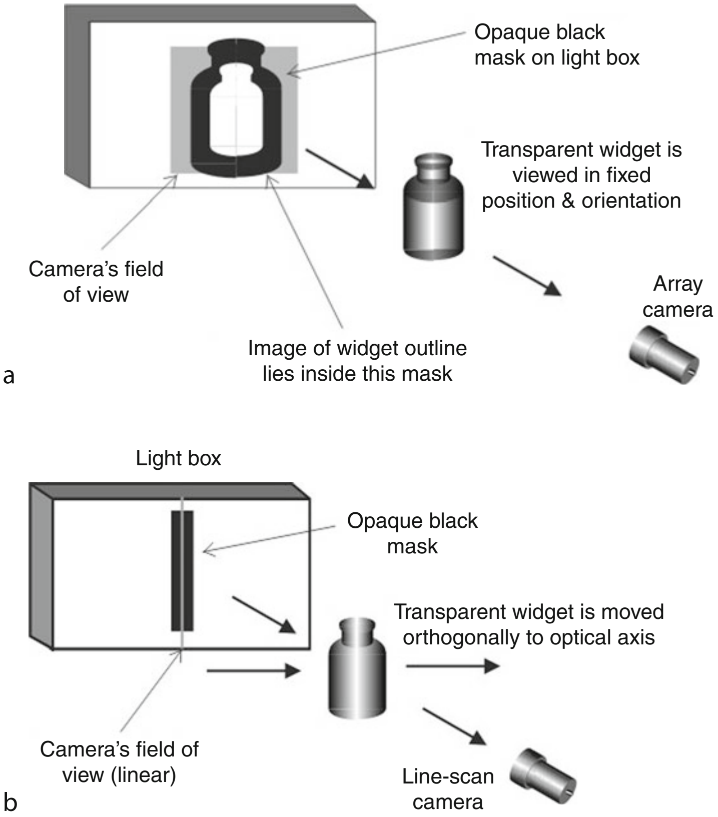

Opaque black mask on light box
Transparent widget is viewed in fixed position & orientation
Array camera
Camera's field of view
Image of widget outline lies inside this mask
a
Light box
Opaque black mask
Transparent widget is moved orthogonally to optical axis
Camera's field of view (linear)
Line-scan camera
b

Fig. 31.1
Dark-field illumination. (a) Area-scan camera. (b) Line-scan camera

- Injecting light into the fully formed product from a small number of carefully chosen "point" sources.
- Observing self-emission in the Near-Infra Red wave band, from hot glasss (i.e. after moulding but before annealing.

These are all described in detail and illustrated in Chap. 40 and will not be discussed further. From a vision engineer's perspective, one of the most important features of glass is its clarity and refractive properties (refractive index is about 1.5–1.7). Care must always be taken to shield a glass object from ambient light during inspection. This rule should be followed whether it is being examined automatically or by eye. Failure to do so will result in poor quality images with highlights occurring in a host of unexpected places. This is caused by direct surface reflection and "ducting" of light by total internal reflection through the body of the product.

Both line-scan and standard (array) cameras have been used for glass inspection. A line-scan camera allows a distinctive form of dark-field illumination to be used. As the camera's field of view is a very thin rectangle, light can be injected into the glass wall very close to it. There is no direct counterpart using an area-scan camera. Moreover, many bottles and jars have

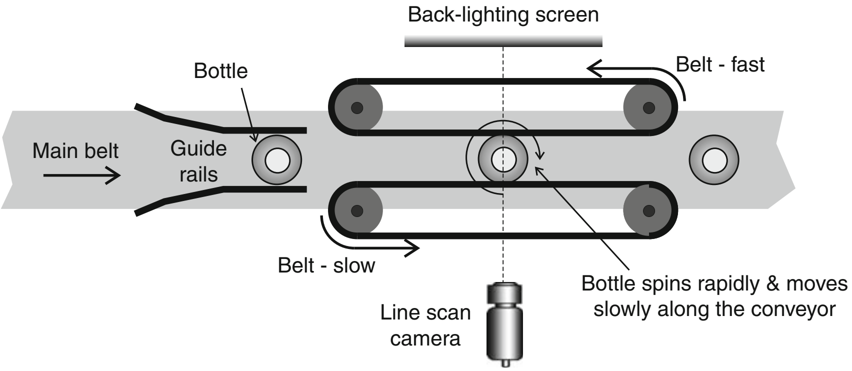


Fig. 31.2
During inspection a round bottle is held between two contra-rotating belts. If the speed of these belts is carefully chosen, the bottle rotates rapidly as it advances slowly along the belt. (It is transported by the side belts and slips on the conveyor.) Hence, a line-scan camera is able to sense local features, such as bird-swings, stones, cracks, etc., wherever they occur around the cylindrical surface of a jar or bottle.

a circular cross section. Again, a line-scan camera offers a unique opportunity in providing an all-round view, which is useful for detecting local defects (➲ *Fig. 31.2*).

Some of the images in the chapter show vignetting and lens distortion, which a telecentric lens would avoid (➲ Chap. 6). This is why the sides of the bottle in ➲ *Fig. 31.3* do not appear parallel. The close-up view of the mouth of a bottle (➲ *Fig. 31.4*) shows another problem: circular features are seen as ellipses of varying eccentricity, depending on their vertical position. In some cases, the effect is that of looking up and sometimes looking down. Again, a telecentric lens would avoid this, viewing all such features precisely side-ways on.

31.3 Bottle Shape

Shape and size measurement is an obvious requirement for both bottles and jars. We shall illustrate various ideas with reference to a back-illuminated bottle. First, however, we must consider how we generate a binary image on which appropriate dimensional measurements can be taken. Consider the bottle shown in ➲ *Fig. 31.3a*. This a typical circularly symmetrical bottle and will serve to illustrate how we derived a suitable algorithm, using the QT interactive image processing system (➲ Chap. 21). As we shall show later, the same procedure has subsequently been applied quite successfully to other glass containers and to images obtained using dark-filed illumination.

31.3.1 Pre-processing

First, of all, we must admit that ➲ *Fig. 31.3a* is not a particularly good of a bottle but it suits our purposes. The background intensity is highly variable as shown in ➲ *Fig. 31.3b* and *c*.

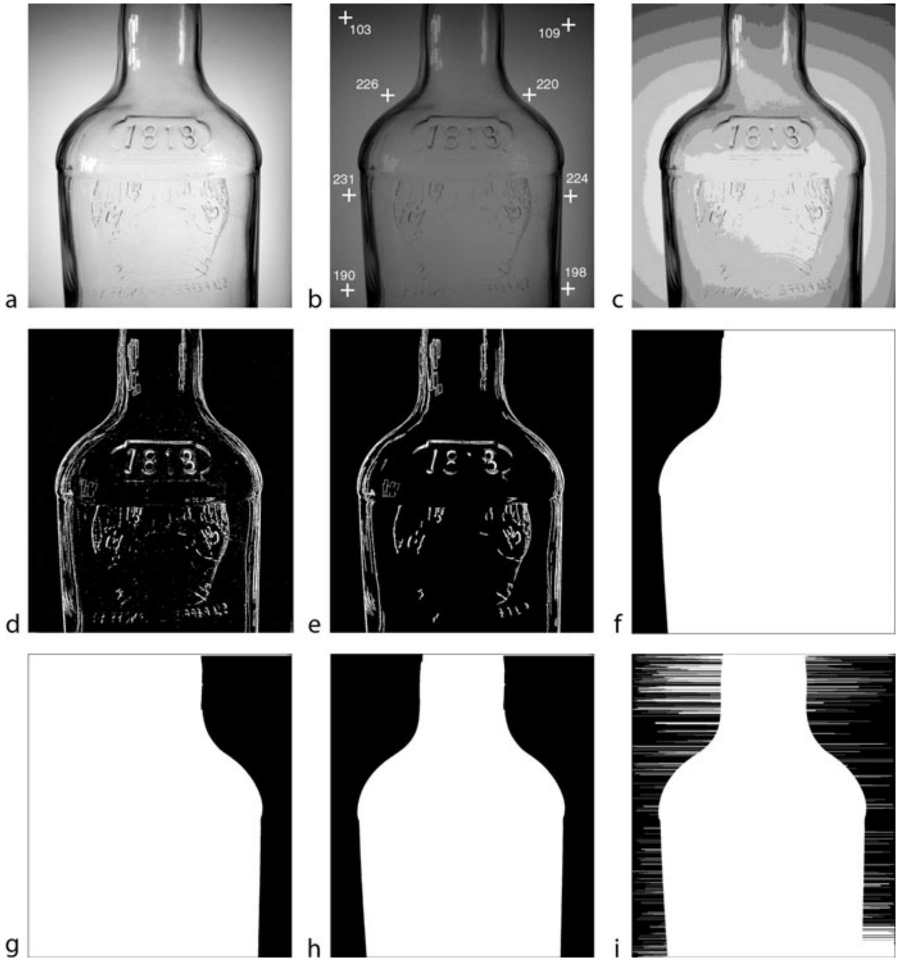


Fig. 31.3

Preliminary processing steps, tracing the outer edge of a bottle. (a) Original image. Back lighting. (b) Background intensity changes over a wide range. Numerals indicate intensity values in the original image. (c) Posterised image, emphasises the variable background intensity. (d) Edge detector. (e) Noise removal. (f) ''Shadow'' projecting to the right [QT: *rox*]. (g) Shadow projecting to the left. (h) ANDing the two ''shadow'' images. (i) Effects of noise pixels outside the bottle profile. This shows why kgr is needed.

This is due to the use of a multi-purpose lens from a digital photographic camera. There is too much ambient light, which causes glinting on the neck and left shoulder. While our casual approach to image formation seems to ignore the lessons so earnestly expressed elsewhere in this book, there is a serious point to be made: the algorithm we have devised is robust and is able to cope easily with these avoidable artifacts. This is in contrast to simple thresholding

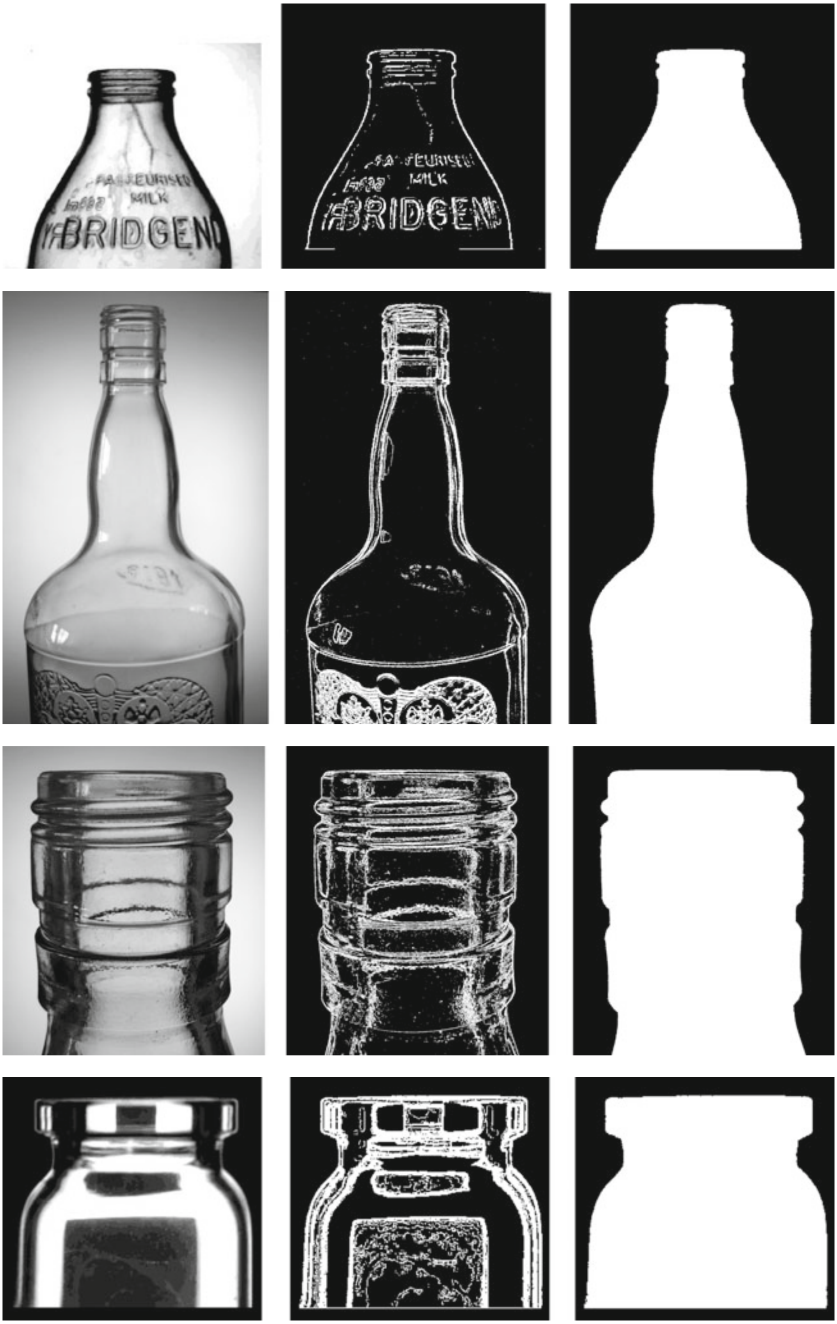


Fig. 31.4

Testing algorithm robustness. The bottom row relates to a brown vial with dark-field illumination. Notice the parallax effect, which is particularly pronounced in the third row. Highlights due to glinting (*second row*) have no effect on the profile but would if they appeared at the edge of the bottle

which is too unreliable. After a short period of experimentation, the following procedure was devised using QT:

```
lnb,       % Largest neighbour
sub,       % Subtract. This & previous command forms edge detector. Other
             options available
neg,       % Negate
enc,       % Enhance contrast
thr(100),  % Threshold (➔ Fig. 31.3d)
kgr(100),  % Keep only big blobs (➔ Fig. 31.3e). ➔ Figure 31.3i shows
             benefits of this operation)
rox,       % Row maximum (➔ Fig. 31.3f )
wri,       % Save image
swi,       % Switch Current & Alternate images
lrt,       % Flip image horizontally
rox,       % Row maximum
lrt,       % Flip image horizontally (➔ Fig. 31.3g)
rei,       % Read image saved earlier
mni,       % AND images together (➔ Fig. 31.3h)
```

Having established an effective algorithm for generating a "solid" profile image, we can now begin to measure the shape of the bottle.

31.3.2 Vertical Sides

There are four important measurements:

(a) Are the sides straight?
(b) Are the sides parallel?
(c) Are the sides vertical?
(d) Is the seperation between the sides correct?

An important point to note is that, at any given moment, we know what type of bottles the factory is making. Hence, we know the ideal dimensions of the bottles we are inspecting. Hence we can use absolute position measurements in the vertical direction.

A simple test can be based on two horizontal scan lines, one positioned just below the shoulder and the other just above the base (➔ *Fig. 31.5*). For simplicity, we shall consider only one side (left). Let us suppose that these scan-lines are at heights Y1 (lower) and Y2 (upper). We need only scan along these two lines to find the corresponding X-coordinates. Call them X1 and X2. The angle of the side wall is then given by

$$\theta_{\mathrm{left}} = \tan^{-1}((\mathrm{Y2} - \mathrm{Y1})/(\mathrm{X2} - \mathrm{X1})) \tag{31.1}$$

A similar calculation yields θ_{right}, the angle of the other side wall. Clearly, θ_{left} and (180 −) should be very nearly equal. The tilt of the bottle can be estimated by finding

$$(\theta_{\mathrm{left}} + \theta_{\mathrm{right}})/2 \tag{31.2}$$

Ideally, this should be 90°.

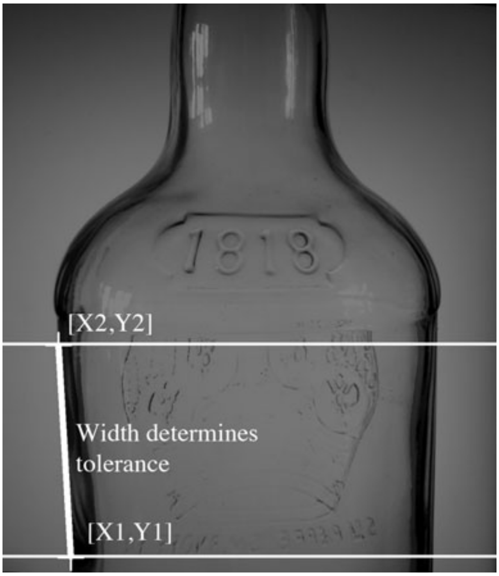


Fig. 31.5
Measuring the angle and straightness of the side walls. Y1 and Y2 are fixed in the knowledge of the product dimensions. To check the straightness of the side wall, draw a thick line [QT: *vpl(X1,Y1,X2,Y2,255), dil(tolerance)*]. The edge of the bottle should be confined to this thick line

The straightness of the side walls can be checked very easily in the following way (➋ *Fig. 31.5*).

- Draw a white line between [X1,Y1] and [X2,Y2].
- Expand this line using binary dilation.
- The amount of dilation determines the tolerance band.
- The edge of the bottle should lie entirely within this thickened line.

A more robust procedure uses regression to fit the straight line (➋ *Fig. 31.6*). This can be used to measure the angle of the side wall and check its straightness in the same way as already described.

31.3.3 Neck and Shoulder

The physical strength of a bottle is determined in large part by the radius/radii of curvature of its neck-shoulder region. This parameter is particularly important if a downwards thrust is applied during capping. Using similar edge-sampling techniques to those already described, it is possible to fit circles and polynomial curves. In ➋ *Fig. 31.7a* and *b*, just three edge points were used to calculate the circle parameters. (QT: *fcd*. Notice that the QT command *pft(y1,y2, no_of_points,2)* fits a general 2nd-order curve; it is not constrained to be a circle.) A polynomial curve can be fitted in a similar manner (➋ *Figs. 31.7c* and ➋ *31.8*). The following M-function shows how the parameters of 3-point circles and polynomials can be calculated. (The QT function *pft* works in a similar way on the second half of *bottle*.)

```
function bottle
% Demonstration: Fit 3rd order polynomial & circle to neck/shoulder
  region of a bottle. [1/0]
global current              % Standard QT declaration
global alternate            % Standard QT declaration
rni(31)                     % Read standard image
enc                         % Enhance contrast
thr                         % Threshold at mid-grey
neg                         % Negate
binary = current;           % Keep result of all this hard work for use
                              again later
[x,y,r] = fce(49,30,30);    % Fit circle; 'hardwired' parameters since
                              we know the bottle
s = ['Circle parameters: X = ',int2str(x),', Y = ',int2str(y),',
  R = ',int2str(r)];
rni(31)                     % Read standard image again
swi,                        % Interchange Current & Alternate images
adi,                        % Add Current & Alternate images
mxi                         % Maximum intensity in Current & Alternate
                              images
alt = current;              % Keep result
current = binary;           % Restore image saved earlier
[x,y] = hzs(80,275, 25);    % Find X values from 25 scan lines in
                              Y-interval [80,275]
p = polyfit(y,x,3);         % Polynomial fitting program (MATLAB
                              function)
q = polyval(p,y);           % Evaluate polynomial (MATLAB function)
zer                         % Black image - same size as Current image was
[u,v] = size(y);            % How big is the image?
for i = 2:v                 % Draw curve as liece-wise linear function
  vpl(round(q(i)),round(y(i)), round(q(i-1)),round(y(i-1)),255)
end
% Next line: Formatting message for the user
t = ['; Polynomial: (',num2str(p(1)),'*y^3) + (', num2str(p(2)),'*y^2)
  + ... (',num2str(p(3)),'*y) + (',num2str(p(1)),')'];
rni(31)                     % Read standard image yet again
swi,                        % Interchange Current & Alternate images
adi,                        % Add Current & Alternate images
mxi                         % Maximum intensity in Current & Alternate
                              images
alternate = alt;            % Recover intermediate result, ready to
                              display it
```

```
subplot(1,2,1) ,            % Display image
  imshow(current)
subplot(1,2,2) ,            % Display image
  imshow(alternate)
```

31.3.4 Symmetry, Tilt and Gross Misshapes

In order to verify that a bottle is upright and symmetrical, we use a simplified (i.e. one-dimensional) version of the medial axis transform (Chap. 13). The QT function *mch* finds the mid-point of each *vertical* white chord in a binary image. Hence, to generate the white pixels in *Fig. 31.9* we applied the following QT command sequence to the binary image in *Fig. 31.3h*

```
yxt     % Interchange X and Y axes because mch works on vertical chords
mch     % Mid point of vertical chord
mch     % Repetition is necessary because mch does not finish the job on
          large images
yxt     % Interchange X and Y axes - restore image to original orientation
```

For a well-made bottle, the points generated in this way should all lie on, or very near, a vertical white line. A straight line (*L*) can then be fitted to these points, by regression analysis. *L* is, of course, an estimate of the axis of symmetry. Its orientation is a measure of the tilt. Flipping the image about *L* allows the symmetry to be checked (*Fig. 31.10*).

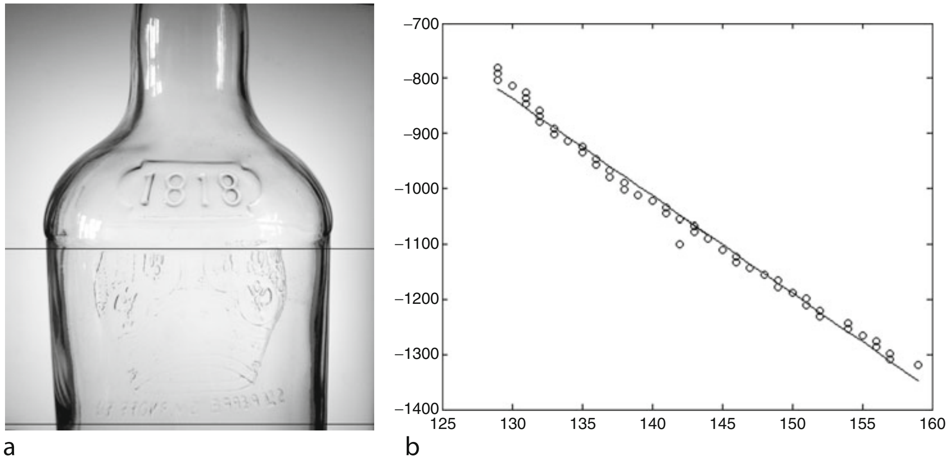


Fig. 31.6

Using regression analysis to fit a straight line to (*the left-hand edge of*) a bottle. (a) Edge pixel coordinates within the limits indicated by the two black lines formed the input for the regression analysis program. (b) Straight line fitted to the edge pixels. Numbers indicate coordinate values, relative the top-left corner of the image [QT: *pft(780,1320,50,1)*]

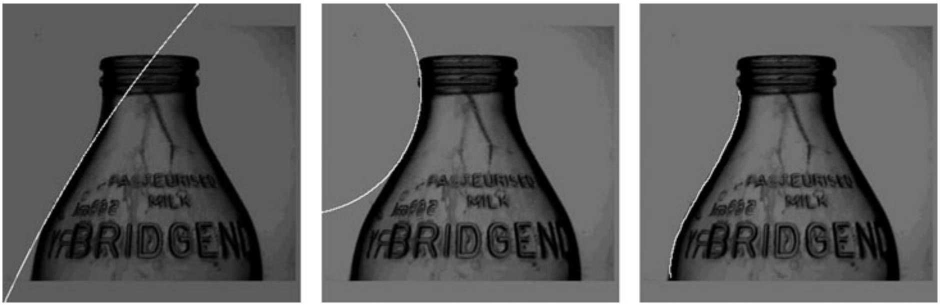


Fig. 31.7
Fitting circles to the neck-shoulder region using just three pixels. (a) Shoulder. This circle was fitted using just three edge pixels. [QT: *fcd*] (b) Neck. (c) 3rd-order polynomial function. This and (b) were generated using *bottle*, listed in the text

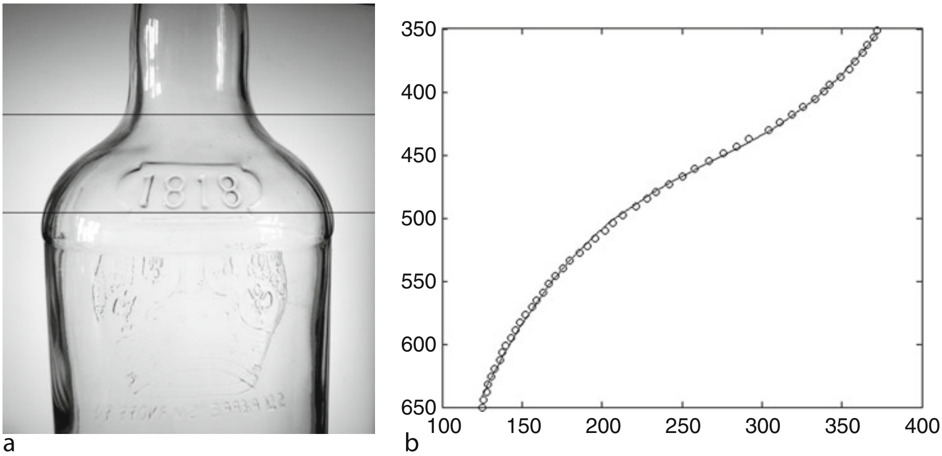


Fig. 31.8
Fitting curves. (a) Original image. The black lines indicate sampling limits. (b) 3rd-order polynomial function [QT: *pft(390,650,50,3)*]

If the neck of the bottle were bent to one side, *L* would not fit all of the white points accurately.

Another simple but effective check can be made at the same time, by measuring the length of each horizontal chord across the bottle. This yields a graph like that shown in *Fig. 31.11*. This can easily be compared to the ideal width profile and will immediately warn us, for example, if the bottle has collapsed under its own weight when it was in a semi-molten state. *Figure 31.12* shows the kind of defect that can be detected in this way.

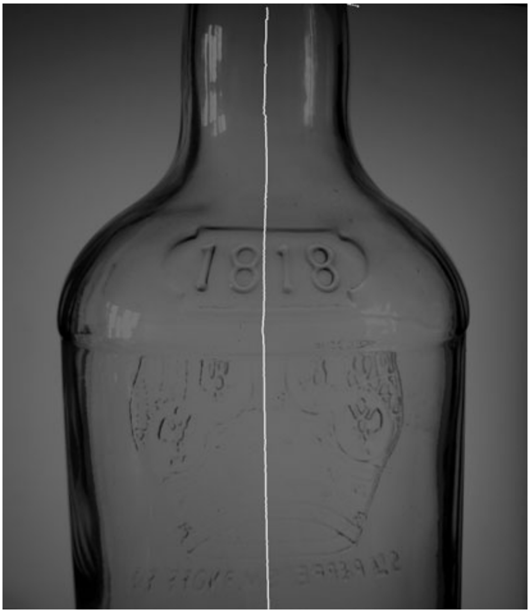


Fig. 31.9
Central axis, used for verifying that the bottle is tilted and symmetrical

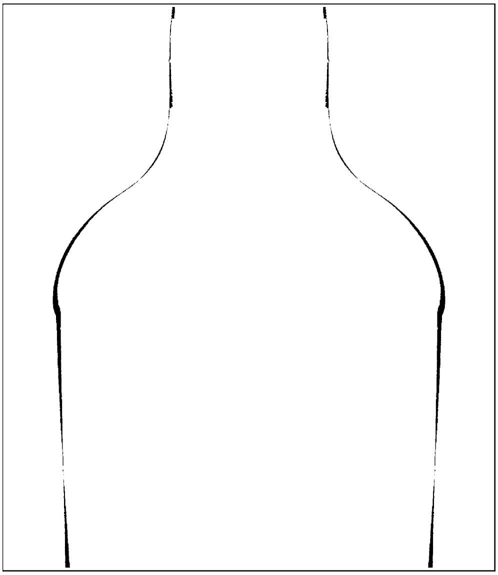


Fig. 31.10
Checking whether a bottle is symmetrical. Flipping the binary image about the axis of symmetry shows the differences between the two sides of the bottle. Simply counting the black pixels gives a crude measure of symmetry

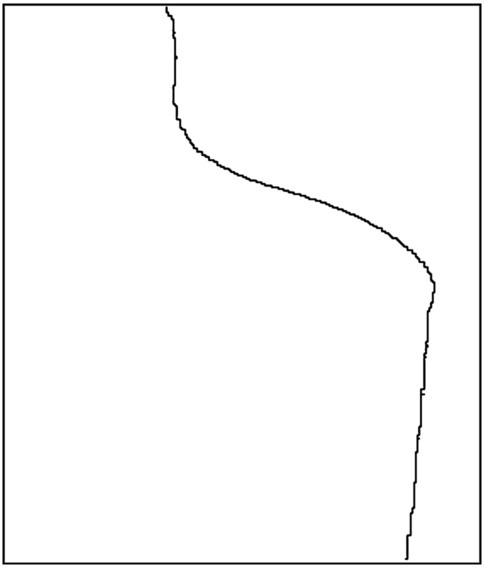


Fig. 31.11
Bottle width measurements

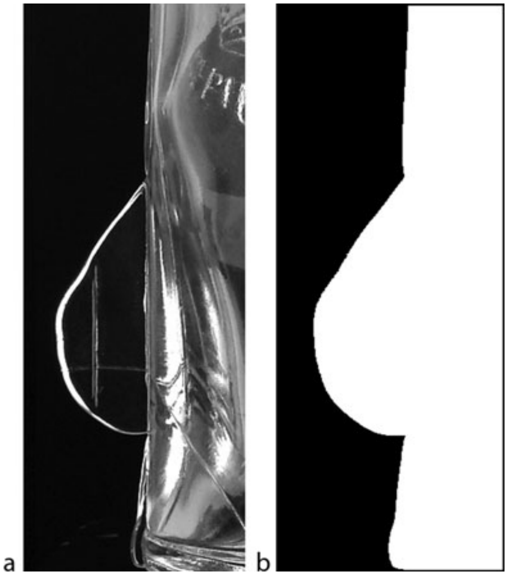


Fig. 31.12
Lower portion of the edge of a bottle with a serious fault: a "fin" caused by the mould not closing properly. (a) Original image. Dark-field illumination. (b) Binary image generated by the algorithm as described in Sect. 31.3.1

31.4 Other Features

In addition to shape, there is a variety of other important features on bottles, jars, tumblers and other glass containers. It is impossible to cover these in full here. Hence, we shall simply demonstrate that Machine Vision can be used effectively in other ways.

31.4.1 Measuring Filling Level

Checking that a glass vessel is fully charged is clearly of interest, particularly in industries such as food, toiletries, cosmetics and pharmaceuticals. Two distinct approaches seem promising. In the first, the meniscus is detected by exploiting the fact that it produces a dark horizontal streak with back-lighting (➲ *Fig. 31.13*). Integrating the intensity along the rows of the image produces a one-dimensional array of numbers from which the height of the meniscus can be calculated. The meniscus position corresponds to a local minimum in this array. When the meniscus lies within the neck–shoulder region, this method is less reliable, although it is effective in ➲ *Fig. 31.16*.

In the second method, we make use of the fact that a filled round bottle forms a crude cylindrical lens, which may be used to distort a pattern, such as a series of vertical parallel lines, placed behind it. In ➲ *Fig. 31.14*, the tumbler is a simple cylinder, with no embossing. It is a straightforward matter to detect the sudden change in the width and spacing of the dark stripes. A slightly different approach is taken in ➲ *Fig. 31.15*, where a single magnified stripe "fills" the jar, up to the meniscus but not above it. The vertical position of the transition can be detected by the function *jar_fill_level*, defined below. This is based on run-length coding (*rlc*) and thresholding (*thr(X)*) to find horizontal cords that have more than *X* white pixels. (The value of *X* is chosen to be a little greater that the width of the stripe as it appears where there is no water present.) Here is the listing of *jar_fill_level*.

```
function y = jar_fill_level
% Find the the filling level in a jar. [1/0]
global current          % Standard QT declaration
global alternate        % Standard QT declaration
original = current;     % Save original image
enc                     % Enhance contrast
thr(0,127)              % Threshold at mid-grey
kgr(1000)               % Eliminate blobs with area < 1000 pixels
yxt                     % Interchange X and Y axes
rlc                     % Run-length code
thr(250)                % Threshold
big                     % Select biggest blob
yxt                     % Interchange X and Y axes
[x,y] = tlp;            % Top-left-most point
zer                     % Black image
[w,h] = dgw;            % Picture dimensions
vpl(1,y,w,y,255)        % Draw horizontal white line at height
alternate = original;   % Recover original image
```

```
subplot(1,2,1) ,          % Display Current image
  imshow(current)
subplot(1,2,2) ,          % Display Alternate image
  imshow(alternate)
```

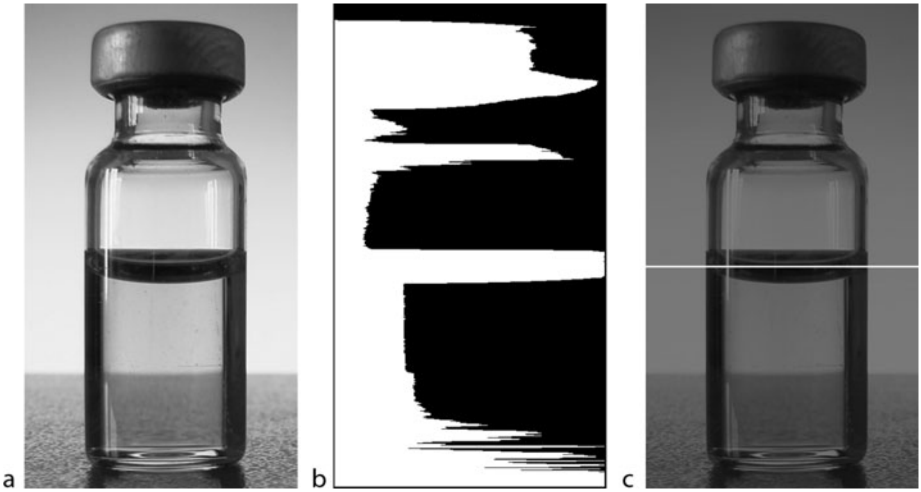


Fig. 31.13
Filling level in small clear glass vial. (a) Original image. Back lighting. (b) Plot of total intensity in each row [QT: *rin, csh, wgx,sub, thr*]. (c) Centre line of the dark horizontal streak due to the meniscus [QT: *rin, thr(X), thn finds the centre line*]

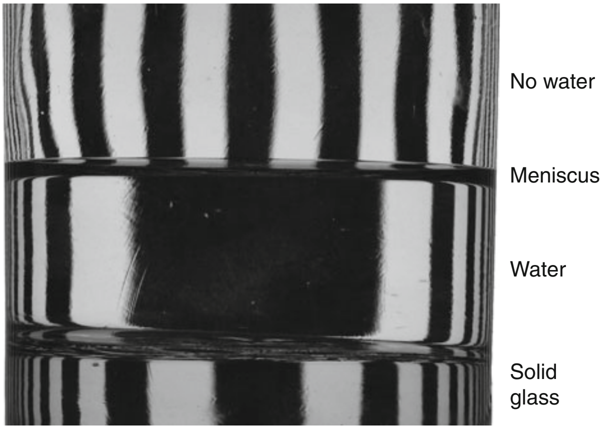


Fig. 31.14
Distortion of a striped background pattern by a partially filled cylindrical tumbler with vertical straight sides, a thick bottom of solid glass but no embossing. Notice that the width and spacing of the dark stripes is changed by the water, which forms a cylindrical lens

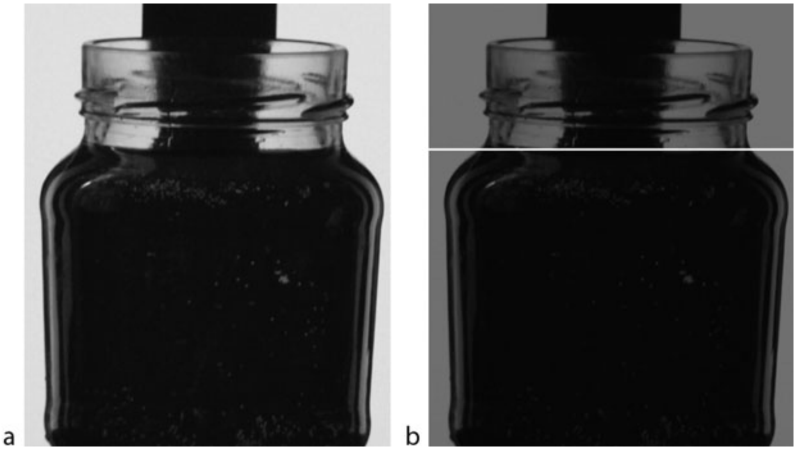


Fig. 31.15
Second method for finding the filling level. (a) Original image. A single black stripe is magnified by the glass-air ''lens'' so that it fills the whole width of the jar. The background stripe is visible at the very top of the image. (b) The white line is the output of *jar_fill_level*

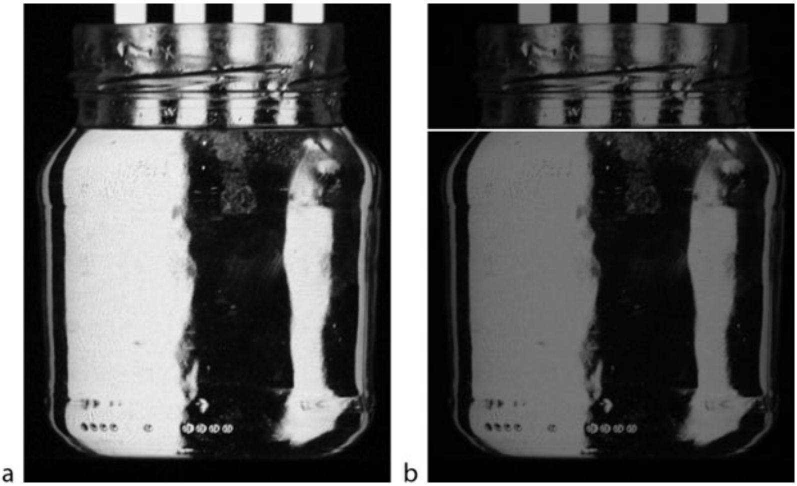
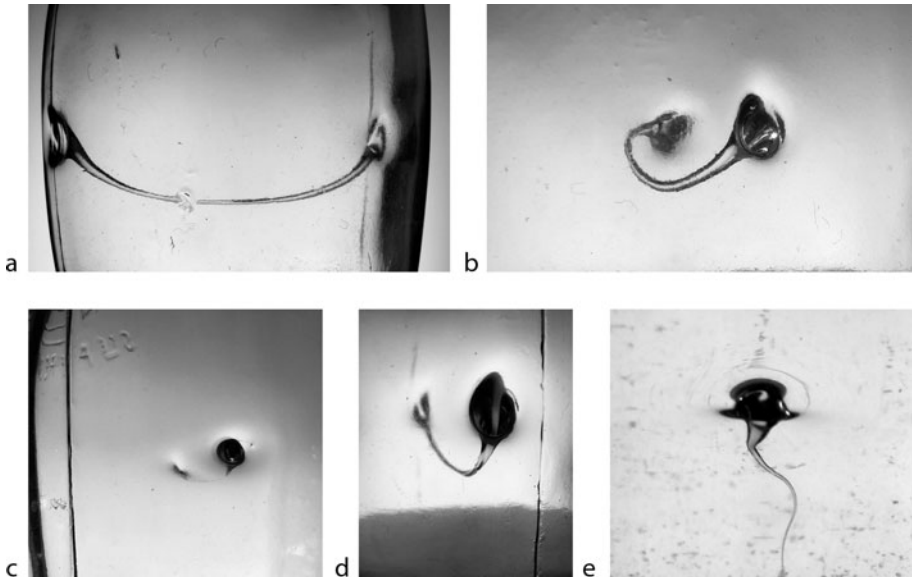


Fig. 31.16
Detecting changes in stripe width/spacing using morphology. (a) Original image. The poor definition of the edges of the broad black stripe does not adversely affect this method. (b) Filling level detected [QT: *wri, enc, thr, eri(200,1), [x,y] = tlp; zer, [w,h] = dgw; vpl(1,y,w,y,255), rei, swi,adi,mxi*]

An alternative algorithm detects changes in the width and spacing of the stripes using morphology. The structuring element should be in the form of a horizontal line, one pixel high. ➲ *Figure 31.16* illustrates this.

31.4.2 Bird-swings, Spikes, Embossing

A bird-swing is a critical defect that occurs in bottles. It consists of a filament of glass that spans the interior of the bottle and is attached at two diametrically opposed points on the side wall. The roots of the filament are conical. This is important as the roots deflect back-lighting and cause a dark shadow. A spike is like a stalagmite of glass attached at only one end and may occur on the base or side wall of a jar. Once again, the root has a conical form. Bottles are made by a blow-blow process and jars by a press-blow process. Hence, they are susceptible to different types of defect. A bird-swing/spike, in a stationary bottle/jar is easily detectable by human eye. However in a factory, when the product is moving at a high rate (perhaps exceeding ten bottles/jars per second), this is not so easy. People get bored and tired on repetitive inspection tasks and are easily distracted. For these reasons, a human inspector will not detect all critical defects with resulting product safety and liability concerns.



◘ Fig. 31.17
Critical defects due to excess glass. Back illumination. (a) Bird-swing viewed from the side. Notice the conical roots of the filament, which can be several millimetres thick or as fine as a human hair. The latter is more dangerous. (b) Bird-swing, viewed near end-on. This is the preferred viewing angle. (c) Bird-swing. The dark vertical streak is the shadow of the moulding seam. (d) Bird-swing. The dark patch at the bottom is the image of the table-top. (e) Spike (The author's daughter narrowly missed injury from this defect, which occured in a milk-bottle. The tail of the spike is finer than a human hair.)

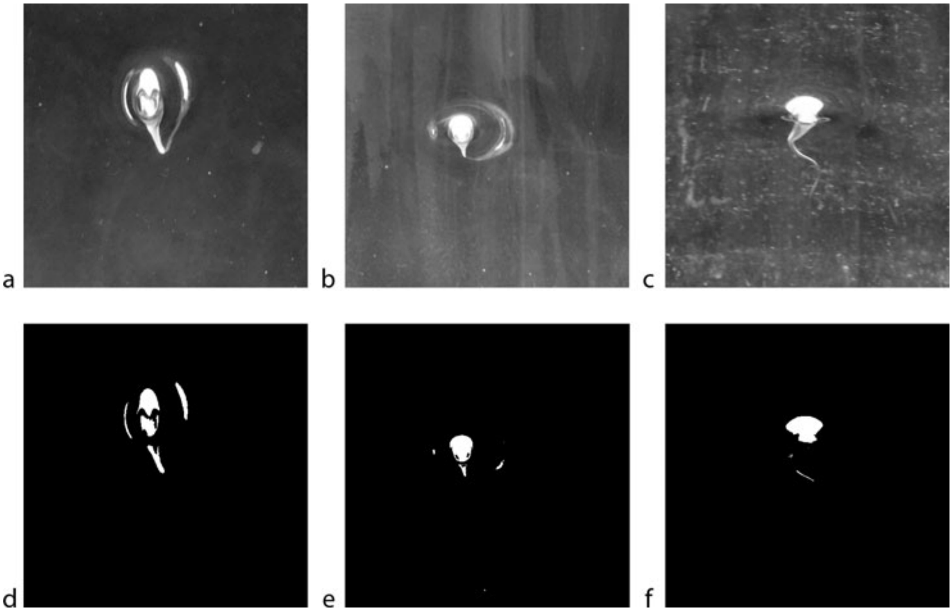


Fig. 31.18

Dark-field illumination produces total internal reflection on bird-swings and spikes. The high contrast produced in this way makes simple fixed-parameter thresholding straightforward. (a) Original image: bird-swing. (b) Original image: bird-swing. (c) Original image: spike. (d) Bird-swing, thresholded. (e) Bird-swing, thresholded. (f) Spike, thresholded

Two series of bird-swing/spike images are shown in *Fig. 31.17* (back-lighting) and *Fig. 31.18* (dark-field illumination). In view of the high contrast, simple thresholding is able to detect both types of defect quite successfully, with either type of illumination. To be sure that thresholding is independent of lamp brightness variations, it is always good idea to normalise the intensity scale first. (QT: *enc*. As is emphasised throughout this book, this is good practice, since light levels can vary as lamps age or supply voltages vary. People are insensitive to such changes.) However, *Fig. 31.17* and *Fig. 31.18* are unrepresentative in showing bird-swings and spikes in clear areas, where there is no embossing. It is evident in *Fig. 31.4* (two top rows), that embossing can also produce high contrasts. This means that, in order to detect bridswings/spikes, we have to examine the embossing for deviations from the norm. This is not always easy or reliable. This is one situation where the product might usefully be designed with inspection in mind. Since bird-swings and spikes are both potentially dangerous types of defect, anything that aids their detection is welcome.

31.4.3 Cracks

Cracks need no introduction. Crizzles or checks are like cracks but do not penetrate right through the glass. Both can be seen, either as dark streaks, if they block light from a back-illumination source, or very bright if they deflect light by total internal reflection at the fracture

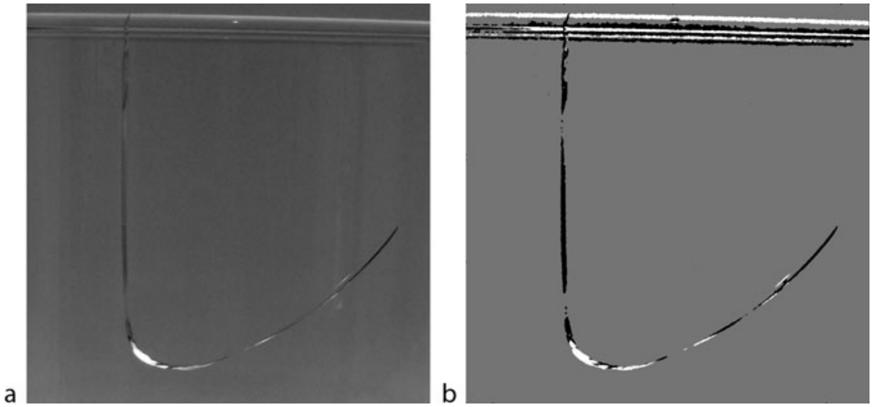


Fig. 31.19

Crack, plain-sided tumbler. (a) Original image. Notice that the crack, which is continuous, sometimes appears dark, bright or totally disappears. (b) Dark and light streaks highlighted separately [QT: *crack2(11,135,50,4)*]

surface. If the crack is viewed edge on, it may not be visible at all see (Fig. 31.19a). It is always worth experimenting with the angle of the lighting viewing arrangement, to optimise the contrast created by cracks. Sometimes, cracks form preferentially in one direction, rather than another. In this case directional lighting from one side (or top/bottom) may produce the best results.

The morphological closing operator (QT: *crk*) is an effective way of detecting dark (or bright) streaks produced by cracks. Here is a more complete algorithm that includes noise-reduction filtering.

```
function crack(a,b,c,d)
% Demonstration: Detect cracks manifest as dark streaks. Bright streaks
  are ignored [0/0]
global current
global alternate
% Default parameters were set to detect the crack in a ferrous component:
  QT image no. 41
if nargin == 0
          a = 5;
          b = 145;
          c = 50;
          d = 4;
end
original = current;
crk(a)    % Adjust parameter to taste: bigger for wider cracks
thr(b)    % Fiddle with this parameter
kgr(c)    % Discard blobs with area < c pixels
dil(d)    % Dilate, noise reduction filter - part 1
ero(d)    % Erosion, noise reduction filter - part 2
```

```
rbe             % Remove blobs touching image border
alternate = original;
subplot(1,2,1) , imshow(current)
subplot(1,2,2) , imshow(alternate)
```

In order to detect cracks in glass that can be either bright or dark compared to the background, it is necessary to apply *crack* twice, before merging the results (➜ *Fig. 31.19b*).

```
function crack2(a,b,c,d)
global current
global alternate
original = current;
mdf(5)            % Median filter, noise reduction
wri(1)            % Save original image
crack(a,b,c,d)    % Defined above - detecting dark streaks
wri(2)            % Save intermediate result for later
rei(1)            % Read original image
neg               % Negate
crack(a,b,c,d)    % Defined above - detecting bright streaks
neg               % Negate
rei(2)            % Read intermediate result saved a little while ago
adi               % Add images
neg               % Negate (Purely cosmetic function: keep original
                    dark-bright polarity)
alternate = original;
subplot(1,2,1) , imshow(current)
subplot(1,2,2) , imshow(alternate)
```

As the author was searching for a suitable algorithm, several others were investigated using QT. (This is exactly what QT was designed to do.) Convolution (high-pass) and DOLP filtering are obvious alternatives producing results that are roughly comparable to those created by *crack* and *crack2*. However, they are certainly no better and are often inferior.

31.5 Final Remarks

In this short chapter, we have only touched on the subject of inspecting glass-ware. We have not considered embossing, stress, loose glass fragments, etching, decorative cutting, engraving, glass–metal seals, glass–glass fusion, coloured glass, air bubbles (accidental and intended), crizzles, staining, surface dirt and inclusions. Inspecting optical components is a highly specialised subject and relies heavily on high-purity lighting from lasers. Inspection *through* glass is also important. In this, we include products such as glass instrument covers, computer display screens, automobile wind-screens, head-lamps, etc. Clearly, glass inspection is a big and important subject. However much work we do, there will always be new challenges where the quality and safety of glass products needs to be improved.

32 Stemware Inspection System

J. P. Sacha[1] · *Spencer D. Luster*[2] · *Behrouz N. Shabestari*[1] ·
John W. V. Miller[3] · *Murat Sena*[4]
[1]Edison Industrial Systems Center, Toledo, OH, USA
[2]Light Works, LLC., Toledo, OH, USA
[3]The University of Michigan-Dearborn, Dearborn, MI, USA
[4]Visionex, Inc., Green Bay, WI, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_32,

Abstract: A system for the inspection of stemware for geometrical defects and glass imperfections is described in this paper in which cameras at three stations are used for enhanced defect detection. Internal glass quality, contaminants and optical nonuniformity are evaluated at the first station. The second station inspects the glass rim and stemware shape and detects internal defects such as cracks. The third station inspects the stemware base for geometrical and internal defects. Glass irregularities are optically enhanced using a striped illumination pattern and detected using morphological processing. Geometric inspection is enhanced through the use of converging illumination at the second station, while the third station utilizes telecentric optics. Progressive scan cameras and frame grabbers capable of simultaneous image capture are used at each station. The system software comprises six modules consisting of the system manager, I/O manager, inspection module for each station, and stemware sorting and logging module. Each module runs as a separate process. Applications communicate with each other through TCP/IP sockets, and can be run in one or more computers. Two personal computers are used for the system described here.

32.1 Introduction

The manufacturing of glassware requires good process control to maintain high quality which in turn requires good product monitoring. Traditionally, human inspectors have performed this task, unfortunately not very effectively or consistently. Glassware defects are often subtle and can be caused by a variety of factors. Defects include misshapen ware, surface contamination, cracks, and inclusions which may simply be cosmetic or affect the structural integrity of the ware. In either instance, quality considerations dictate that defective ware with such defects should be rejected.

The objective of the inspection system described here is to provide 100% inspection of stemware for these defect types. Three different stations perform the inspection operations, a body inspection station, a base station and the sidewall/rim inspection station. In each of these, special illumination and optics are used to enhance the associated defect types. For the body and sidewall/rim stations, cameras acquire images of stemware from three angles to provide complete coverage while the Base Station uses two cameras. An overview of the system, including the Reject Station is shown in ➔ *Fig. 32.1*.

The Body Station inspects internal glass quality such as contamination and waviness using a vertical-line pattern placed over the light source which will be distorted by the optical characteristics of the sidewall of a given piece of ware. Excessive distortion is generally indicative of a defect, which is detected by analyzing the resulting captured image with appropriate algorithms. Contamination caused by chemical residue is detected at this station by the presence of numerous small dark or bright nonuniformities contrasted against the brighter or darker regions of the striped background.

The Sidewall and Rim Station inspects the rim, geometry of the stemware body and stem, and internal defects such as cracks and certain refractive defects that are difficult to detect in the body station. A uniform illumination source is used here to back-light the ware but special optics are used so that the light is highly collimated. This provides very high contrast between the edge of the ware and the background illumination so that ware shape can be measured very accurately. Refractive defects are also easily detected with this type of illumination.

The Base Station evaluates the base geometry and detects defects in this region. The ware is inverted so that an unobstructed view of the base is obtained. The light sources are similar

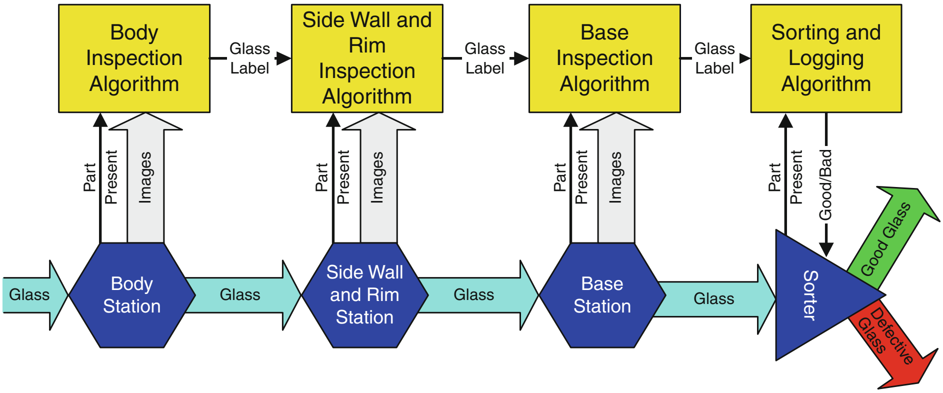


Fig. 32.1
Inspection and sorting components

to those used in the Sidewall and Rim station so that the rim edge has high contrast and refractive defects can be detected.

Shuttered progressive-scan cameras and frame grabbers capable of simultaneous image capture are used at each station since the ware moves at a relatively rapid rate past the cameras. Shuttering the cameras minimizes motion blur and progressive scanning eliminates interlace distortion.

The system software is comprised of six modules: system manager, I/O manager, inspection module for each station, and stemware sorting and logging module. Each module runs as a separate application, which communicate with each other through TCP/IP sockets. This allows the software to be run in a multiple- or single-processor configuration efficiently. Currently two Windows NT workstations are used.

32.2 Optical and Illumination Considerations

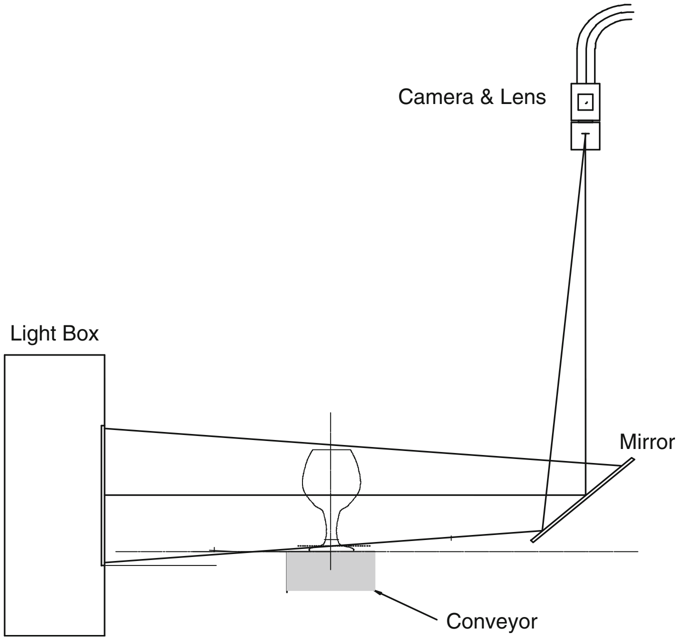
The optical layout was designed to

- Maximize sensitivity to defects
- Minimize problems associated with magnification
- Keep illumination sources reasonably small as well as the overall volume needed for the inspection system

Mirrors are used extensively to fold optical paths. An example of this can be seen in *Fig. 32.2*.

32.2.1 Body and Sidewall/Rim Stations

These stations each include three identical imaging and lighting systems whose views of the stemware are separated by 120°. This angular separation is intended to allow viewing of all inspection areas of the ware. Each system includes three progressive scan area camera each with a c-mount lens and a diffuse light source. Each source consists of a halogen lamp, fiber optic



■ **Fig. 32.2**
Arrangement of camera and light source for sidewall inspection

cable, and diffuser. A large Fresnel lens is used to image the diffuser onto the camera lens plane and a mirror folds the optical path to minimize overall system size.

32.2.2 Base Station

The station includes two identical imaging and lighting systems whose views of the stemware are separated by 180°. Each camera views the upside-down stemware at an angle of about 30° from horizontal. With this configuration, the base appears as an ellipse but only the top half of the ellipse is used for measurement and defect detection. Each system includes a single array area scan camera with a large aperture telecentric lens and a large area diffuse light source [1]. The view angle for the base station was chosen to provide the best base image possible without interference from the body of the stemware, particularly for short, wide ware such as brandy balloons (snifters). ➲ *Figure 32.3* shows the basic layout.

32.3 Image Analysis Algorithms

A variety of image analysis techniques is needed for the range of defects that occur with glassware. Mathematical morphology is used extensively since glass defects typically have structural features that distort light in unusual ways due to their refractive properties. After potential defects have been isolated by morphology, connectivity analysis is used to identify defects.

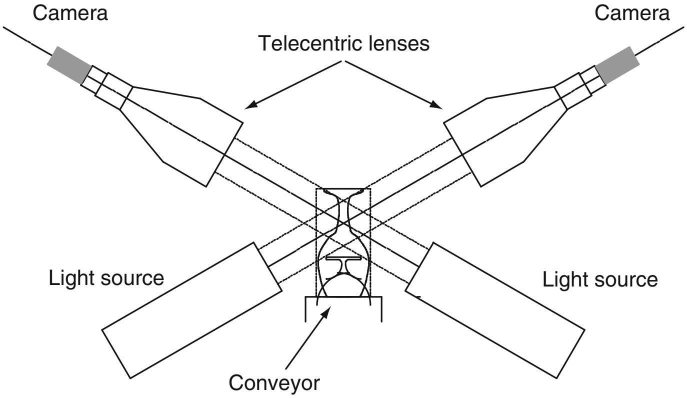


Fig. 32.3
Arrangement of camera and light source for base inspection

32.3.1 Detection of Refractive Defects

Refractive defects present a significant challenge because acceptable glassware exhibits a wide range of refractive properties. Also, what is considered acceptable for one type of glassware may be totally unacceptable for another. It was important to provide adjustable parameters and setup aids to provide adequate flexibility.

Several different types of illumination were employed to improve the detection of this class of defect. A vertical striped pattern, used in the body inspection station, is sensitive to many of these defects, especially when they cover an extensive portion of the sidewall but exhibit relatively mild refractive properties. For example, one type of defect, designated as *wavy*, causes excessive bending of the striped pattern over large areas of a glass as illustrated in ➲ *Fig. 32.4*. A long structuring element, which is parallel to the striped pattern is used to detect such distortions. The length of the structuring element can be used as a sensitivity adjustment. Virtually all glassware exhibits significant waviness in the lower part of the sidewall so this region is masked off.

For smaller, more refractive defects, the stripe pattern was distorted over a relatively small area as illustrated in ➲ *Fig. 32.5*. Here a small dark region and even smaller bright region are visible near the top of the glass centered horizontally. Morphological opening and closing operations can isolate both defects by removing them and subtracting the processed images from the original images.

Not all refractive defects can be detected with this pattern and light source, however. In the sidewall/rim inspection and base stations, highly-collimated illumination is used in part to isolate the glass edge and is discussed in the following section. This light source is very effective for finding refractive defects since they divert light away from the camera and appear as dark regions.

➲ *Figure 32.6* illustrates how a small defect can be detected with this light source. A morphological closing operation is used to detect the defect [2]. A masking algorithm is used to identify the very dark edges and other regions of the ware so that a large portion of the sidewall can be inspected. The rim is also inspected using this view for bumps and other irregularities using morphology.

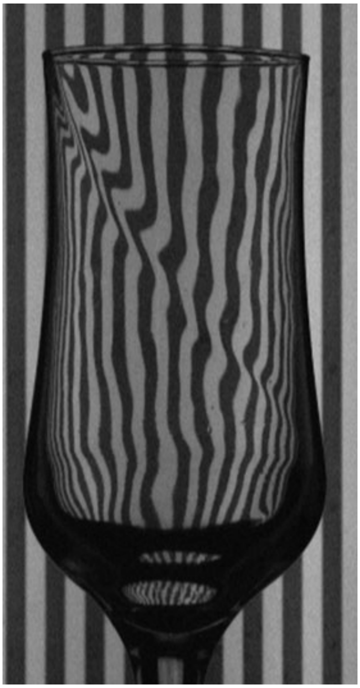


Fig. 32.4
Wavy glass

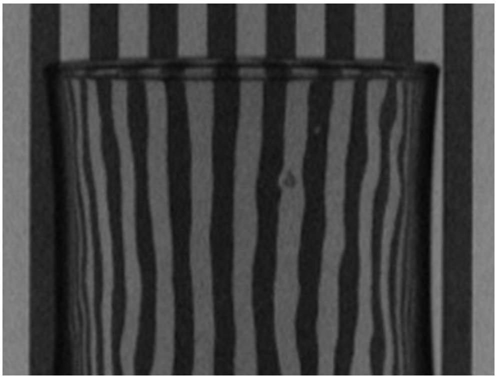


Fig. 32.5
Small defects

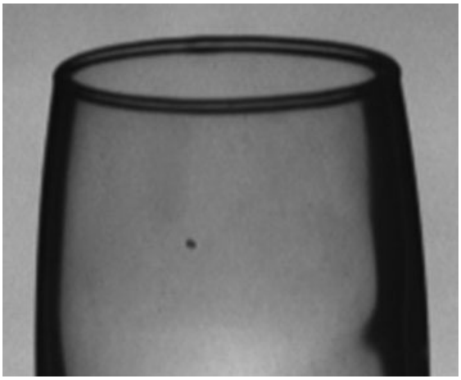


Fig. 32.6
Small defect

32.3.2 Geometrical Inspection

Glassware can be misshapen if there is a problem with certain manufacturing parameters such as temperature. While this type of product does not require high precision, glassware must not appear to be irregular. With the collimated illumination provided in the sidewall/rim and base inspection systems, edges are very distinct and can be detected with very good accuracy using simple thresholding.

Once the edge points of the sidewall have been obtained, they are processed to obtain an estimate of the centerline of the ware by finding their average. Regression is used to determine if the centerline deviates significantly from vertical. Curvature of the centerline indicates that the glass is leaning.

The overall shape of the ware is typically not very critical but some means to characterize the acceptability is needed. To address this issue, software was developed so that the system could be trained using ware designated as *acceptable*. This was accomplished by generating a signature created by subtracting the left edge from the right edge of the sidewall. Using a number of glassware samples, an average signature was generated that could be used to identify glasses that were significantly different. A similar approach was used for the base station.

32.4 User Interface

The user interface is a critical aspect of the inspection system since the sophisticated image analysis algorithms used need to be set up properly for good results given that operators cannot be expected to understand the intricacies. Borland's C++ Builder was used to create the various user screens since it provides tools that allowed them to be created rapidly. In addition, since the C language was used, interfacing to the various image analysis routines was very easy. Each inspection station has its own setup window with a number of tabs so that parameters can be adjusted as needed. In addition, camera images are displayed and graphics can be superimposed over the images to aid in setup. The inspection algorithm of each of the stations can be configured and debugged using the main setup screen.

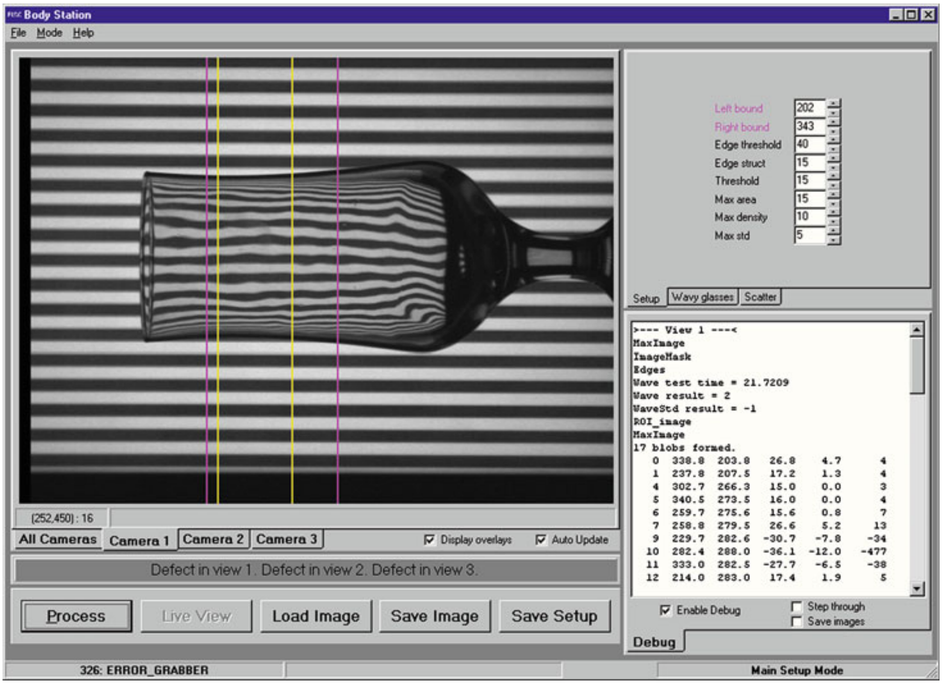
32.4.1 Setup Screens

An example of a setup screen is given in ➲ *Fig. 32.7* which is used for the Body Station. A station setup screen is divided into four sections, image view panel, command panel, parameter setup panel, and debug log panel. The command panel allows the operator to perform a number of actions that include

- Processing the displayed image
- Enabling a live view from the cameras
- Loading a previously stored image
- Saving a captured image
- Saving the current setup information

Images are captured by enabling and then disabling the live view option which may then be saved if desired. These features provide operator flexibility and the ability to track how production is varying over time. Setup time is also greatly reduced by recalling previous setup information.

The parameter setup panel is used to configure an inspection algorithm. It typically contains several pages that can be selected by pressing the appropriate tab. The parameter setup panel is synchronized with the image view panel. Parameters are adjusted for each camera view separately with only a few parameters common to all camera views. To access parameters for a desired camera view, the operator selects the corresponding tab in the image view panel.



Left bound 202
Right bound 343
Edge threshold 40
Edge struct 15
Threshold 15
Max area 15
Max density 10
Max std 5

(252,450) : 16

☑ Display overlays ☑ Auto Update

Defect in view 1. Defect in view 2. Defect in view 3.

```
>---- View 1 ---<
MaxImage
ImageMask
Edges
Wave test time = 21.7209
Wave result = 2
WaveStd result = -1
ROI_image
MaxImage
17 blobs formed.
   0  338.8  203.8   26.8    4.7      4
   1  237.8  207.5   17.2    1.3      4
   4  302.7  266.3   15.0    0.0      3
   5  340.5  273.5   16.0    0.0      4
   6  259.7  275.6   15.6    0.8      7
   7  258.8  279.5   26.6    5.2     13
   9  229.7  282.6  -30.7   -7.8    -34
  10  282.4  288.0  -36.1  -12.0   -477
  11  333.0  282.5  -27.7   -6.5    -38
  12  214.0  283.0   17.4    1.9      5
```

☑ Enable Debug ☐ Step through ☐ Save images

326: ERROR_GRABBER

Main Setup Mode

Fig. 32.7
Body Station – main setup screen

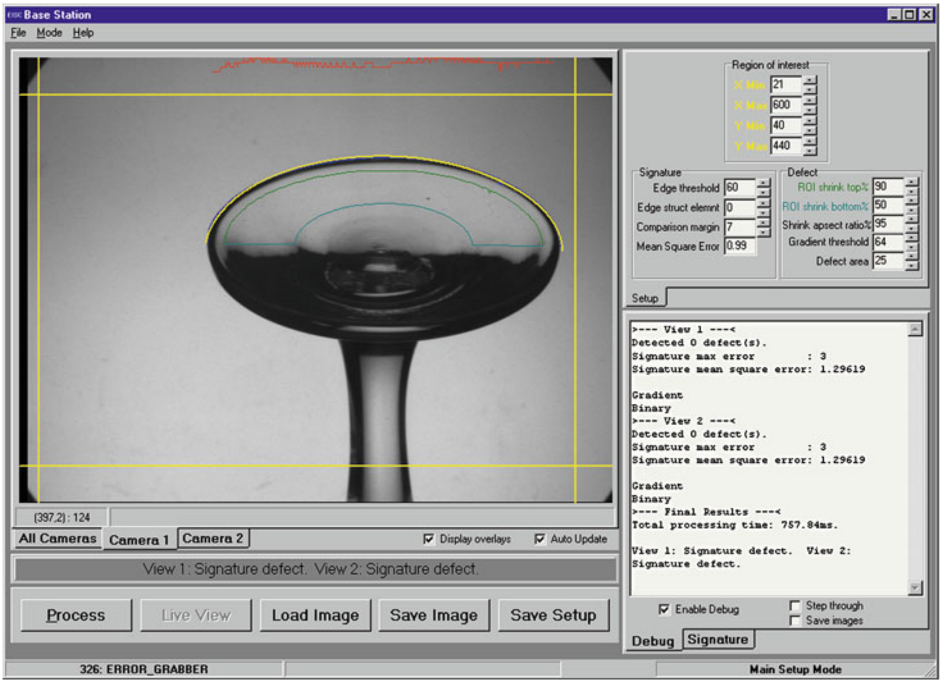
Parameter names which are displayed in color can be graphically illustrated in the image view panel. The color corresponds to that of the line in that panel. Modification of a parameter can be automatically updated by checking the appropriate box to allow updates to be displayed simultaneously.

The debug log panel displays debug information from the inspection algorithms. Display of debug information can be disabled using *Enable Debug* check box. When debugging is disabled, only the total execution time of the inspection algorithm is displayed. Note that debugging increases execution time and is not indicative of execution time in production mode. Check boxes *Step through* and *Save images* control display and automatic saving of debug images.

32.4.2 Body Station

As noted previously, the body station inspects for the presence of refractive defects, chemical residue and waviness in the glass sidewall. A number of important parameters need to be set here including top and bottom boundaries for the type of glassware being inspected, the defect threshold, and defect area and density. The density of a defect is the average gray level of the corresponding blob after processing.

Parameters are also provided for detecting contamination which appears as small, locally scattered spots and are detected by the "Scatter" tab. They may be brighter or darker than the



(397,2) : 124

☑ Display overlays ☑ Auto Update

View 1: Signature defect. View 2: Signature defect.

326: ERROR_GRABBER

Region of interest

X Min 21
X Max 600
Y Min 40
Y Max 440

Signature

Edge threshold 60
Edge struct elemnt 0
Comparison margin 7
Mean Square Error 0.99

Defect

ROI shrink top% 90
ROI shrink bottom% 50
Shrink apsect ratio% 95
Gradient threshold 64
Defect area 25

```
>--- View 1 ---<
Detected 0 defect(s).
Signature max error        : 3
Signature mean square error: 1.29619

Gradient
Binary
>--- View 2 ---<
Detected 0 defect(s).
Signature max error        : 3
Signature mean square error: 1.29619

Gradient
Binary
>--- Final Results ---<
Total processing time: 757.84ms.

View 1: Signature defect.  View 2:
Signature defect.
```

☑ Enable Debug ☐ Step through ☐ Save images

Fig. 32.8
Base Station – main setup

local background. Parameters include total particle count and a value for combining blobs in close spatial proximity. Similarly, the "Wavy Glass" tab allows parameters related to measures of waviness to be adjusted.

32.4.3 Sidewall and Rim Station

Parameters for the sidewall and rim station are adjusted in the same manner as the body station. Parameters differ since other measurements are taken, especially with respect to ware shape and rim measures. Nominal values and allowable deviation from these values can all be adjusted here. These include geometrical measures such as rim thickness, regularity and differences from a reference shape. The reference shape can be established by running a series of acceptable ware through the system in training mode.

32.4.4 Base Station

Similarly as in the side wall and rim station, the shape of the stemware base can be trained by presenting examples of acceptable ware. Allowable deviations of the inspected shape compared to the learned one can be adjusted. The difference between learned and inspected shape is displayed in the upper part of the diagnostic image, see ❯ *Fig. 32.8*. The region of interest for defect inspection inside the base is defined in the relation to the outside silhouette to the base: the silhouette is shrunk inwards to define the ROI.

32.5 Results and Conclusions

The system has been rigorously tested in the laboratory using a wide variety of defects and appears to provide highly reliable defect detection. It is currently being installed in a plant so no production results are as yet available. Inspection time for any given inspection (with three cameras) is under 100 ms with up to an additional 60 ms required for image acquisition. Total inspection time for all three stations is well under 600 ms as required to meet production specifications.

In general, this system provides 100% inspection of stemware for a wide range of defects including sidewall and rim defects, misshapen and improperly sized ware, and base defects. Lighting techniques have been optimized to enhance the contrast of defects and also provide high-quality images for performing geometrical measurements. Shuttered progressive scan cameras provide a means of capturing sharp images of glassware moving down a production line. Software has been optimized for speed and performance so that a wide range of defects can be detected at real-time rates.

References

1. Luster S (1998) "l'elecentric Imaging: a Camera Lens, a doublet, and thou." In: Applied Machine Vision98 Conference, Nashville, May 18–21, 1998
2. Miller JMV, Watlz FM (1997) Software implementation of 2D gray-level dilation using SKIPSM. In: Machine vision applications, architectures, and systems integration V1, SPIE vol 3205, pp 145–152, September 1997, A 215

33 Restoration of Missing Boundary Segments in Shattered Glass

John W. V. Miller · Malayappan Shridhar · T. Peatee
The University of Michigan-Dearborn, Dearborn, Michigan, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_33,

Abstract: Analysis of shattered automotive glass for determining particle size, shape and count is legally required to demonstrate that safety criteria have been met. Manual methods are labour intensive, subjective and prone to errors. This paper presents an approach for automating this analysis. Problems associated with missing boundary segments can cause serious errors unless corrective measures are applied. Techniques for efficient extraction of particle boundaries and restoration of missing segments are presented.

33.1 Introduction

A problem commonly encountered in machine vision applications is the analysis of cellular or interconnected patterns in which the features of interest are either separated by the cell boundaries or are the boundaries. In either case, an important part of the analysis is the isolation and identification of the boundaries using some type of segmentation algorithm or boundary detection scheme. Applications which require this type of processing include analysis of biological cells and crystal grain studies [1]. The analysis of shattered glass, which has a similar appearance, is the topic of this paper.

If the cell boundaries are incomplete or have areas with poor contrast, analysis becomes much more difficult. Missing boundaries will cause adjacent regions to merge during connectivity resulting in fewer 'cells' with larger areas and highly irregular shapes. For this reason, it is necessary to detect the broken boundaries and reconstruct them before meaningful analysis can be performed. A strategy for accomplishing this is presented here with the emphasis on the analysis of images of shattered automotive safety glass.

Due to legal requirements concerning specifications on tempered safety glass, destructive testing on a given number of glass panels taken from production is routinely performed. In a typical test, a glass sample is shattered and certain statistical information is manually extracted from it. Statistics, such as the particle count in regions of the panel with largest and smallest particle sizes and length of longest shards, are commonly gathered.

With the current method, application of an adhesive transparent film is applied to the sample to prevent the pieces from scattering during testing. The sample is then placed over a sheet of photo-sensitive print paper and shattered. The print paper is then exposed with a high-intensity light source and developed. Manual identification of two 5 × 5 cm regions, one with a high particle density and the other with a low particle density, is performed along with manual counting of the particles within these regions. Manual measurements are also done to obtain statistical shape information. This process is labour intensive, error prone and does not provide fast feedback. If a production problem occurs, a large number of finished glass panels may have to be destroyed because the problem was not detected early enough.

A suitable machine vision system, however, would be able to perform the above tasks much faster than manual analysis with more consistent results and at lower costs. The faster feedback would minimise production of unacceptable glass and require much less labour. Such a system is also capable of automating the report generation required by many countries regarding the specifications of the safety glass, as well as maintaining the required permanent records of tested samples using data compression techniques and optical storage technologies.

Algorithms for such a system are described in the remainder of the paper. In the next section, a brief overview of the basic approach is given. In ➲ Sect. 33.3, preprocessing techniques are presented, and the new boundary restoration algorithm is described in

➲ Sect. 33.4. ➲ Section 33.5 examines an alternative approach for boundary reconstruction using watershed transforms, and test results are given in ➲ Sect. 33.6.

33.2 Basic Approach for an Automated Glass Analysis System

The automated system under development utilises a linear camera with 64 grey levels positioned over a conveyer to produce a digital image of the shattered glass. Optionally, images can be acquired from a print, produced as explained above, using a scanner. With either approach, the image is then processed to obtain the desired statistical data.

A major difficulty is the acquisition of an image with distinct boundaries. A typical image is shown in ➲ *Fig. 33.1* in which a number of particle boundaries have poor contrast or missing segments. Due to intensity variations, many of the cracks are lost with simple thresholding, but this can be minimised with more effective image processing. A more severe problem occurs when a crack is perpendicular to the plane of the camera and light source. Such cracks are very small and do not refract sufficient amounts of light to be detected as dark regions by the camera. As a result, there are missing boundary segments that cause the merging of two or more pieces into one larger piece. This decreases the accuracy of the particle counts and distorts statistical information. Such regions must be reconstructed before accurate automated particle analysis can be performed.

Previous work in this area utilised a morphological process in which each piece was subjected to a series of erosion operations until it either was segmented into smaller pieces or was reduced to an insignificant size. If segmentation occurred, the resulting piece or pieces were individually dilated using the same number of iterations that were performed during erosion to approximate their original size.

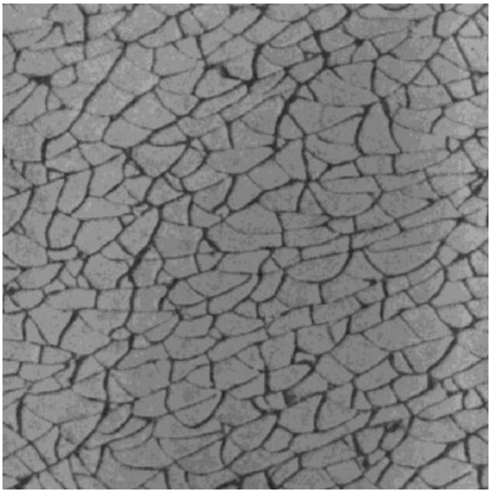


Fig. 33.1
Typical image of shattered glass

One of the underlying assumptions was that pieces were basically convex in shape which was true of some initial shattered panels. However, it was found that other samples contain long thin shards. If the shard had less thickness in the centre than at the ends, repeated erosion operations would incorrectly split the piece at its thinnest point.

Clearly, improvements were needed to obtain sufficient accuracy. Using a technique to restore missing boundary segments rather than a segmentation approach would provide significant improvement. In addition, careful preprocessing would provide a better starting point for boundary restoration. The next section describes a useful processing approach.

33.3 Image Processing

Since the image size of typical windshields scanned at a resolution greater than 100 dpi can be greater than 3 $\times 10^6$ pixels, processing time and hardware costs can become excessive. However, because only 5 × 5 cm regions of interest need to be evaluated fully, just corresponding pixels from the image need to be analized. A given region is extracted from the original image along with a 10 pixel border region which is required for later morphological filtering operations.

The first step in preparing the image for analysis is to perform a logarithmic transformation to remove the multiplicative effects of illumination between areas of high and low intensity. Each pixel in the image is assigned a new grey-scale value by the relationship

$$p[i,j] = \frac{255 log(p[i,j])}{log(255)} \tag{33.1}$$

where $p[i, j]$ denotes the grey-scale value of the pixel at (i, j).

Next, a background estimate is made on the logarithmically transformed image by performing a grey scale morphological closing operation with a structuring element large enough to span the cracks in the image. The resulting background estimate is then subtracted from the log-transformed image to create an inverted image with the particle boundaries bright against a dark background.

To remove any shading still present in the image, a 5 × 5 gradient operation is performed followed by clipping to remove negative pixel values. Since the gradient uses only local information and the image was previously log transformed, the resulting image contains only the particle boundaries and noise. To suppress the noise, a morphological closing operation is performed with a very small (two element) structuring element.

The resulting image is then binarised as illustrated in ➲ *Fig. 33.2*. Two methods have been successfully used. The first approach simply thresholds the entire image about a carefully selected threshold. A minor shortcoming here is that the threshold value was found to change as a function of the glass type which must be manually determined.

A second approach is to segment the image into approximately 25 subregions and threshold each subregion about its local average. Since most of the image is background in which the boundaries are bright, the resulting average will tend to favour the background, producing a suitable threshold. This procedure was found to provide similar results to that of a manually selected global threshold and has the advantage of being completely automated.

At this point the 10 pixel frame is removed from the image, and a single pixel border is placed around the resulting image to connect all boundary segments that have an endpoint at the boarder of the region of interest. Four connectivity is then applied, and remaining isolated

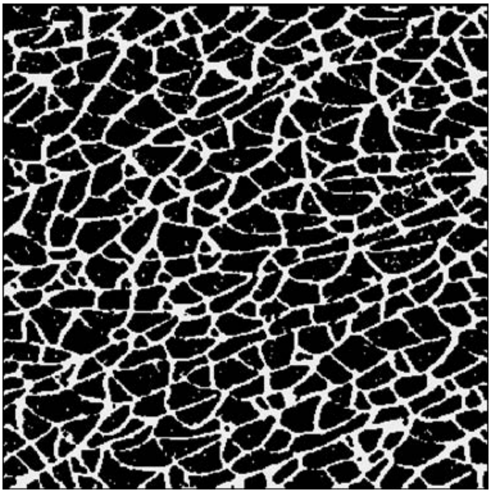


Fig. 33.2
Image after gradient and thresholding operations

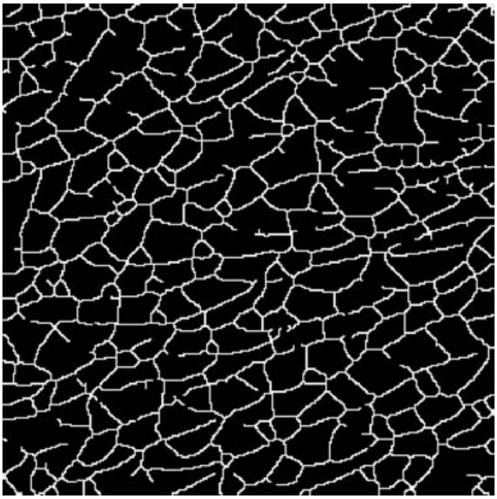


Fig. 33.3
Image after thinning

pixels are discarded. The cleaned, binary image is then thinned by repeated applications of a hit-or-miss transform as illustrated in *Fig. 33.3*.

After thinning, any "branches" which are only a few pixels in length are removed from the image by a simple pruning algorithm. This algorithm uses the same binary classifier as the reconstruction algorithm described in the next section to locate dead-end branches and their corresponding base points. With each branch identified, it is a simple matter to create an image without the small undesired branches caused by the thinning algorithm. The resulting pruned image is then ready for reconstruction.

33.4 Reconstruction Algorithm

Any branches remaining after pruning are assumed to belong to the border set and as such must be reconnected to the remainder of the border. Typically, two cases are encountered. In the first case, there is a section of a border missing from the middle of an edge. In this case, reconstruction involves connecting the resulting two branches to each other. In the second case, a section of border is missing from the 'corner' to somewhere in the middle of a piece. In this case, there is only one branch, and it must be extended in such a way as to complete the boundary pattern.

An algorithm to perform the reconstruction is outlined below and later described in more detail.

1. Apply binary spatial relationship classifier.
2. Identify points with only one neighbour.
3. Locate base point, the first point along a branch that has more than two neighbours.
4. Determine direction of segment.
5. Search for other points with only one neighbour in the direction of the segment, which are designated as candidates for connection.
6. Connect any branches which have only one candidate for connection.
7. Next consider branches with multiple candidates. First, reject any candidates that do not contain the current segment as candidates for their connection. Then, connect the current segment to the remaining candidate closest to the current segment's direction.
8. Connect remaining branches to the vertex (points with more than two neighbours) closest to their direction, or if no vertex was found, extend the branch in its direction until the extended branch connects with the crack pattern.

Since the image is four connected and thinned, every point has at most four neighbours. To identify points with only one neighbour, the pixel value (represented internally as an 8-bit character) is replaced with the results of the following binary classifier, where "$*$" represents the current pixel.

$$\begin{bmatrix} 0 & 4 & 0 \\ 1 & * & 0 \\ 0 & 8 & 0 \end{bmatrix}.$$

The resulting value for a pixel with an adjacent pixel to the north, and another to the west would be five, and so forth. The resulting pixel values now represent the connectivity and spatial relations of each point in the image. Identifying dead-end branches is simply a matter of finding pixels with only one neighbour, namely those with values of 1, 2, 4 or 8.

Once such points have been found and recorded as end points, their accompanying base point must be found. The base point is the point at which a branch either connects to the boundary pattern, or in the case of a totally isolated branch, simply ends. The algorithm makes use of the directional information produced by the classifier to follow the branch from its endpoint to the base point, which is then recorded for future use.

The equation of a line (l) from the base point passing through the endpoint of the segment is determined as illustrated in ➲ *Fig. 33.4*. Next, the point along the branch that is furthest from the line that maximises distance d is found which is designated as point A in ➲ *Fig. 33.4*. The point halfway from point A to the end point is then determined. The direction of the branch is then estimated by the line starting at point B and passing through the end point.

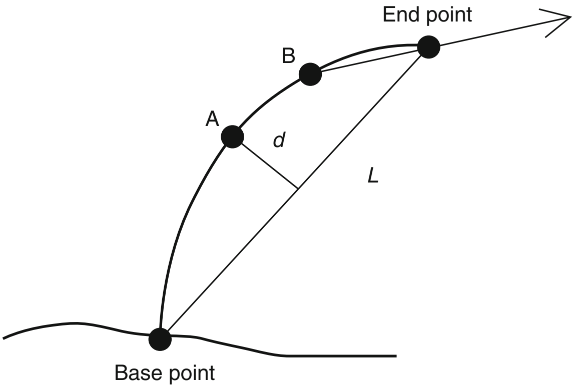


Fig. 33.4
Technique for measuring direction

It should be pointed out that simply using the last two points in the segment to determine direction would always result in directions of up, down, left or right due to the underlying four connectivity. Several other methods for obtaining the direction were tried, including fitting a spline to the points along the segment and taking the derivative near the end. It was found that the spline approach yielded inaccurate results due to the oscillations of the spline between the data points.

Note that the direction does not necessarily reflect the direction (derivative) at the end point. To improve the estimate further, the direction is modified by an amount proportional to the curvature of the segment. Note that the original estimate is correct if the segment is itself linear, and as the curvature increases, the estimate becomes worse.

First, a measure of the amount and direction of curvature is made. If the angle from the horizontal to line l is less than the angle from the horizontal to line BA, the segment is said to curve right as illustrated in ➔ *Fig. 33.5*. Similarly, as shown in ➔ *Fig. 33.6*, if the angle to line l is greater than the angle to line BA, the segment is said to curve left. An analogy can be made by considering how an automobile is steered to an endpoint along an arc.

The amount of curvature *C* to the left or right is assigned a value by the relationship

$$C = \frac{300d}{l} \tag{33.2}$$

This is a heuristic function that has been experimentally determined to yield satisfactory results. Note that only one direction of curvature can have meaning for any given segment. If the segment curves left the variable C_R (Curve Right) would be set to zero and C_L (Curve Left) would be found from (➔ 33.2). The direction μ of the segment is then updated to be:

$$\mu = \frac{\mu + (2\alpha\, C_L - 2\alpha\, C_R)}{2} \tag{33.3}$$

where α is a user specified constant, corresponding to the shift for the curvature associated with the given segment. Experimentation has shown values of α around 12 provide good results.

Once the direction of the segment is determined, a search is conducted along a linear path starting from the end point of the segment, about its direction, for connection to other candidate segments. All vectors starting at the end point and in directions from $(\mu - \delta\mu)$ to $(\mu + \delta\mu)$ are

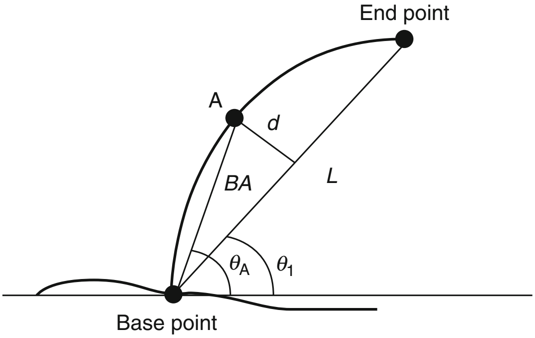


Fig. 33.5
Definition of "Curves Right"

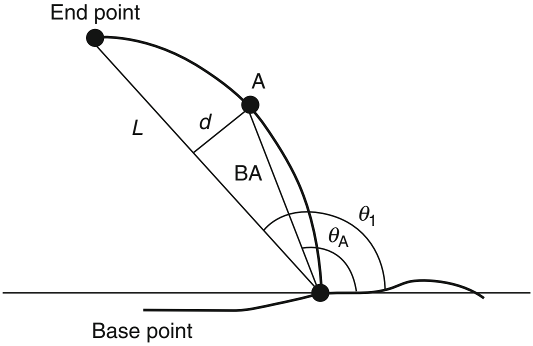


Fig. 33.6
Definition of "Curves Left"

examined to see if they contain any points with only one neighbour corresponding to the end point of another segment. The search along each vector is terminated when either a candidate is found or the vector reaches a boundary. Note that here a candidate means either an end point of another segment or a vertex. Typically, $\delta\mu$ should be between 30° and 45° for best results.

The algorithm identifies all regions where a particle boundary is incomplete and locates any possible end points which can be joined either to other end points or vertices. The next step is to examine this data and make the appropriate connections. This is accomplished by applying the following connection rules in sequence:

Rule 1: Connect all segments that are mutual candidates for each other where at least one of the segments has only one candidate, and mark both segments as connected. Such connections are typically made in regions where a short section of the boundary pattern is missing in a long shard.

Rule 2: Connect all remaining segments with only one candidate and mark both segments as connected. Typically, such connections are made when a short section of the boundary is missing, and one of the segments to be joined is a complex curve, usually having a "hook" at the very end.

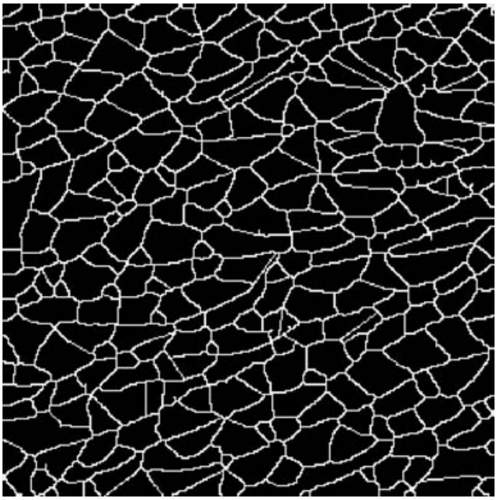


Fig. 33.7
Final image after boundary restoration

Rule 3: Connect all remaining segments which are mutual candidates for each other to the candidate closest to the direction of the current segment and mark both segments as connected.
Rule 4: Connect all remaining segments to the candidate closest to the direction of the current segment and mark both segments as connected if there are any candidates.
Rule 5: Connect all remaining segments to the vertex closest to the direction of the current segment and mark the segment as connected. Note that any connections made at this point are made to segments for which no other segments were found as candidates for connection.
Rule 6: Connect all remaining segments to the boundary pattern by extending them until they connect with the boundary.

At this point, every segment has been completely interconnected with the particle boundaries as shown in *Fig. 33.7*. It should be pointed out that the above rules only allow one connection to be made *from* each segment, but multiple connections *to* each segment are allowed. Hence, a segment can still be a candidate for another connection even if it has already been connected by a previous rule. It is this mechanism that provides reconstruction of missing vertices.

33.5 Segmentation by Watersheds

Segmentation by watersheds is a well-known morphological technique originally developed by Lantuéjol and has been used to analyse images that have many similarities to those obtained from shattered glass [1–3]. The watershed algorithm is quite different from the heuristic procedure described in the previous section and requires fewer assumptions about the nature of images being analysed. The watershed algorithm used here was developed by Luc Vincent and Pierre Soille [3]. Its relative speed and robustness is especially suited for purposes here.

Preprocessing issues are similar for the watershed transform in that a clean set of local minima must be attained prior to the application of the transform. This was accomplished by performing the background correction, noise filtering and positive gradient as before. Note that with the particle boundaries bright against a relatively dark background, the catchment basins correspond to the pieces of glass, while the watershed lines correspond to the particle boundaries. In this manner, the boundaries may be viewed as the watersheds of the gradient [3]. To create the marker set, a simple grey-scale stretch was performed on the gradient image, such that all grey-scale values from zero to three were set to zero, and the remaining values (4 to 255) were equally redistributed (from 1 to 255). This results in clear identification of the local minima. The watershed transform was then applied to this image.

It was found that great care must be taken when enhancing the local minima. Quite often, if the minima are not sufficiently enhanced, the watershed transform takes every impulse and local deviation and treats them as separate catchment basins. This causes too much segmentation which is an inherent difficulty with the watershed transform [3]. It may be possible to alleviate this problem with a more robust technique for detecting local minima.

However, if the preprocessing stage should inadvertently merge two or more real minima into one larger minimum, the resulting watershed lines will only reflect the single catchment basin, resulting in insufficient segmentation. While this methodology is quite powerful, it is very sensitive to noise and has the undesirable property of having poor repeatability when random noise is present in the image.

In general, segmentation was not satisfactory using the watershed approach. A comparison of the original and processed images indicated inconsistent segmentation with failure to reconstruct obviously missing boundaries and segmentation of particles that should not be split. The poor contrast of the boundaries and high noise caused significant problems. Better preprocessing may improve performance considerably given that this technique has been successfully applied in similar applications. The heuristic method's performance was much better for the given application.

33.6 Results and Conclusions

The heuristic reconstruction algorithm has been experimentally tested on a wide range on sample images, obtained both by camera and from scanning prints. Results are comparable with the two methods of image acquisition, although it is felt that scanning with a linear camera with more grey levels would provide better performance than scanning prints. In order to evaluate performance of the boundary restoration algorithm, 20 images of 5 × 5 cm regions were obtained and copies of these regions were given to three individuals for manual analysis. The particle counts obtained automatically compare favourably to those obtained manually with results most of the time falling between the three individuals. The average magnitude of the deviations from the mean of the three individuals was 2.53%, and in all cases the error was less than 6%. In general, automatic particle counting appears to be quite feasible using the technique described in ➲ Sect. 33.4.

After image acquisition, a region of interest is selected. Typically, this is a 5 × 5 cm square window. After background/illumination correction a positive-half gradient is used as a simple edge detector on the inverted image. Such an image is shown in ➲ *Fig. 33.2*. (See ➲ Sect. 33.3 for more details.) The resulting gradient image is then thresholded and thinned. See ➲ *Fig. 33.3*. There are obviously many branches of the crack pattern that simply dead end. Since the crack pattern is totally interconnected, such dead ends represent regions of image drop-out, or thinning anomalies. Any branch of only a few pixels is almost certainly due to effects associated with the thinning process and as such are removed from the image by a simple pruning algorithm. (See preprocessing issues for more information.)

Consider the image of shattered glass shown in ➋ *Fig. 33.1*. Notice that the crack pattern forms a completely interconnected web-like matrix, which forms the boundaries of each individual piece of glass. The objective is to identify each piece of glass in the image. Simply thresholding the image produces inadequate results due partially to the large amounts of noise in the image, but mostly due to local regions of image drop-out where the crack was simply not detected by the camera/scanner. These regions of drop-out usually occur when the crack is perfectly perpendicular to the plane of the camera, and thus refract insufficient light to be detected by the camera (or to prevent exposure of the print).

References

1. Serra J (1982) Image analysis and mathematical morphology. Academic, London
2. Beucher Serge (1990) Segmentation tools in mathematical morphology. SPIE 1350:70–84
3. Vincent L, Soille P (1991) Watershed in digital spaces: an efficient algorithm based on immersion simulations. IEEE Trans Pattern Anal Mach Intell 13(6):583–598

34 Determination of Geometric Stability of Glass Substrates

John W. V. Miller[1] · *Behrouz N. Shabestari*[2] · *Robert Sweney*[3]
[1]The University of Michigan-Dearborn, Dearborn, MI, USA
[2]Edison Industrial Systems Center, Toledo, OH, USA
[3]Electroplasma, Inc., Millbury, OH, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_34,

Abstract: The stability of glass substrates is an important concern for the flat panel display industry. High-resolution displays have very tight geometrical requirements and alignment of the various display components is critical if good performance is to be obtained. Prior to development of manufacturing processes for these displays, it was necessary to determine how glass substrates change during the various processing steps. This chapter describes a system to measure electrode patterns before and after critical processing steps for color plasma panels. The electrode patterns, which are made of thin-film gold, form a series of parallel electrodes. In order to measure electrode locations, a vision system consisting of an X-Y stage, a video camera, a frame grabber, and personal computer was used. Images captured with this setup were processed to minimize the effects of noise and improve accuracy. A gray-scale interpolation technique in which the centroids of the electrodes are calculated was used to enhance measurement resolution.

34.1 Introduction

Flat-panel displays represent one of the key technologies of the future. AC Plasma panels, one example of flat display, consist of two glass plates or substrates separated by a small and very uniform gap as shown in ➲ *Fig. 34.1*. Each plate has an array of parallel gold electrodes with a pitch of about 60 lines/in.. The electrodes are covered by a clear dielectric material and the two plates are oriented so that the electrode patterns are orthogonal to each other as illustrated in ➲ *Fig. 34.1*. The plates are sealed together to form a gap between the opposing dielectric surfaces. Each intersection of two electrodes defines a display pixel or cell that can be lit or extinguished under control of the driving electronics. Spacers are provided at regular intervals to maintain a uniform gap of approximately 4 mils over the panel area.

Traditional plasma panel displays use neon gas with trace amounts of argon and xenon. Since they are monochromatic, they are not suitable for television applications. The use of phosphors and a gas that emits large amounts of ultraviolet light is necessary to produce true-color images. Triads of red, green and blue phosphors must be deposited accurately to maintain the necessary alignment with the electrode intersections.

Maintaining this alignment, however, is quite difficult over large areas because the glass substrates tend to shrink during various high-temperature manufacturing processes, such as dielectric firing and sealing. An understanding of the exact manner in which substrates change during these processes is required in order to design phosphor deposition artwork that will be properly aligned with the electrode patterns. Dimensional changes in the glass substrate are less than the distance between electrodes, so measuring electrode positions before and after each processing step provides an accurate measure of substrate stability. The changes are typically on the order of parts/thousand so highly accurate measurements are needed. It was very desirable to use existing equipment in order to minimize cost. For this reason, an existing system for measuring spacers and dielectric thickness was modified to make the new measurements [1]. In the following sections, a description of the basic system will be provided along with the analysis techniques that were developed.

34.2 The Measurement System

The basic measurement system consists of an X-Y stage to move a given panel substrate for gauging and feature identification, a personal computer with image acquisition hardware and an illumination source. A light-section microscope with video camera and an

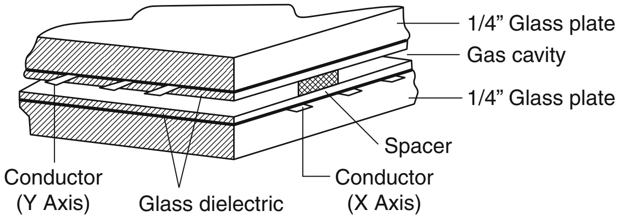


Fig. 34.1
Construction of an AC Plasma Panel

alignment camera are also present in the system but were not used for dimensional stability measurements. A third camera was added for the new measurements.

34.2.1 X-Y Stage

In order to perform measurements at various locations on a given substrate, an X-Y table with stepper motors was used to position it. With this arrangement, the substrate is mounted on the table using a fixture to secure and position the substrate in a consistent manner on the table. Since some variation in the electrode pattern relative to the substrate exists, X and Y offset adjustments are made using a fiducial reference point in the electrode pattern.

The stepper motors are driven by a pair of controllers that accept commands from the host computer via a serial port. The controllers provide the necessary velocity profile by controlling acceleration and deceleration to minimize the time required for movement without sacrificing accuracy.

34.2.2 Image Capture Hardware

For image acquisition, a standard RS-170 camera is used to observe selected substrate areas. A field of view approximately 5 × 3.75 mm is obtained with a 75 mm camera lens and extension tube. A frame-grabber card digitizes and stores the video signal. A live video image can be shown on the monitor so that an additional monitor is not required for setup. Text or graphics can be combined with a recently captured image.

34.2.3 Lighting

Since the electrodes are opaque, the best illumination technique is back illumination. However, no provision was made originally for this type of illumination since the system only used a light-section microscope for measurement. A back-illumination source would be difficult to implement so an alternative technique was used that took advantage of the specular properties of the electrodes.

The panel electrodes have a mirror-like surface, so dark field illumination was used (➜ *Fig. 34.2*). Incident light nearly parallel to the electrodes strikes them at a low angle. This light is reflected away from the camera so the electrodes appear dark. The white surface under the substrate reflects light diffusely and appears bright relative to the electrodes.

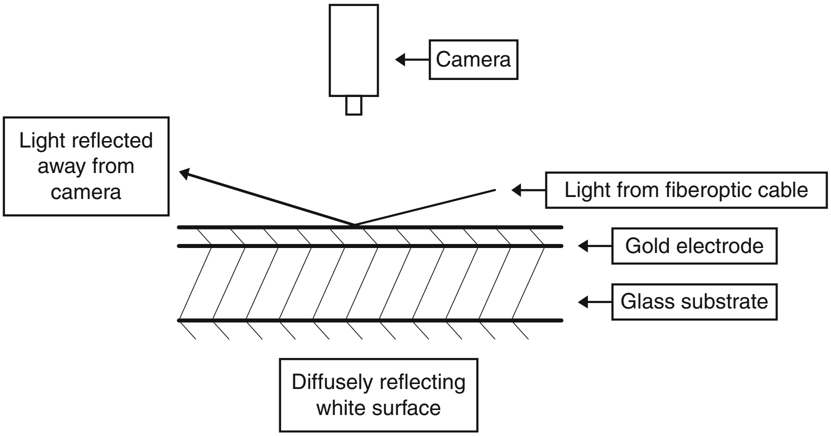

Camera
Light reflected away from camera
Light from fiberoptic cable
Gold electrode
Glass substrate
Diffusely reflecting white surface

Fig. 34.2
Light striking the electrode pattern at a low angle is reflected away from the camera and appears dark

As will be noted later, this type of lighting causes a noticeable gradient across the image in which the side nearest to the light source is markedly brighter than the opposite side. As described in the next section, simple image processing can eliminate problems resulting from this nonuniformity.

34.3 Image Analysis Algorithms

The basic approach to determine the degree of change between different manufacturing steps takes advantage of the regular parallel electrode pattern on a substrate. Measuring the location of each electrode relative to some reference point provides a straightforward technique to determine how a substrate changes.

34.3.1 Image Acquisition and Processing

A matrix of 5 × 3.75 mm measurement regions was defined for a given type of panel substrate and each was moved under the camera and captured at a resolution of 640 pixels/line × 480 lines. A typical image, given in *Fig. 34.3*, shows the panel electrodes as dark horizontal lines against a brighter background. A light gradient is evident in the image and is caused by lighting the substrate from only one side. Simple image processing is able to minimize any errors caused by the nonuniform illumination. The PC provided sufficient processing speed for real-time measurements [2].

It may be noted that there is a great deal of redundancy in the image. The important information can be retained by reducing the image resolution in the horizontal direction by summing adjacent horizontal pixels. This will provide faster processing since the image size is much smaller and, more importantly, reduce image noise because of pixel summing. A reduction factor of 16 was used, so that the final image resolution was 40 pixels × 480 lines.

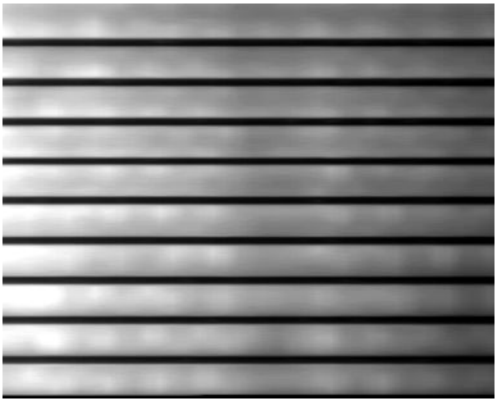


Fig. 34.3
Digitized image of electrode pattern with left-to-right light gradient

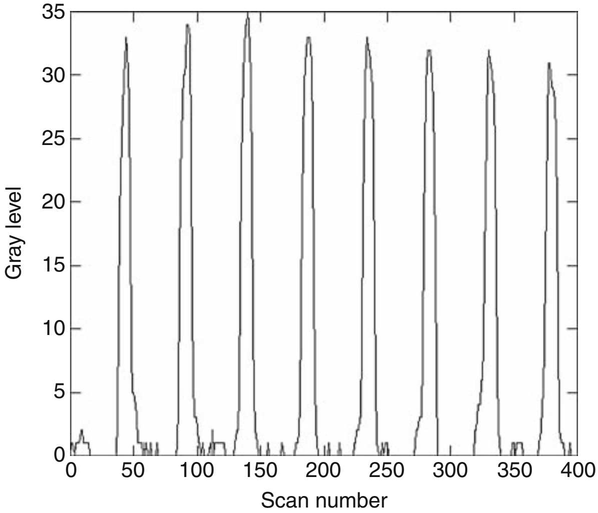


Fig. 34.4
Electrodes appear as bright peaks against a dark background after image processing

After reducing the horizontal resolution, a gray-scale logarithmic transformation is performed to minimize the effects of nonuniform illumination [3]. A gray-scale closing operation is used to estimate the background light level [3, 4]. Since the electrodes are less than 10 pixels wide, a vertical structuring element of this length or greater will remove them. Subtracting the original image from the closed image provides an illumination-corrected image in which the electrodes appear as bright horizontal lines against a dark background. A plot of light level versus scan number given in *Fig. 34.4* illustrates this.

To provide an accurate measurement of the electrode positions the centroid for each electrode at a given x (pixel) locations is calculated using

$$\mu_{xk} = \frac{\sum g(x,y) \cdot y}{\sum g(x,y)}$$

in which μ_{xk} is the centroid for the kth electrode and the column of pixels at x and $g(x,y)$ is the gray level of the pixel at (x,y). The pixels used for this calculation are those from the kth electrode that are above gray level 10 that was established heuristically. The location of each electrode was found by averaging all of the centroids associated with that electrode. An error estimate was calculated at the same time and in general was well under 0.1 mils which is better than the precision of the X-Y stage.

34.4 Results and Conclusions

Measurements were made before and after various processing steps and analyzed using statistical techniques. A model was constructed from these results which showed a linear correlation between distance from a reference point and the amount of compaction with a correlation of 98%. A plot of the results is given in ➲ *Fig. 34.5*.

From this model, artwork for phosphor deposition can be designed that will be properly aligned with the electrode pattern. Measuring electrode locations by calculating gray-scale centroids is an effective and accurate way to improve vision system resolution for this type of application.

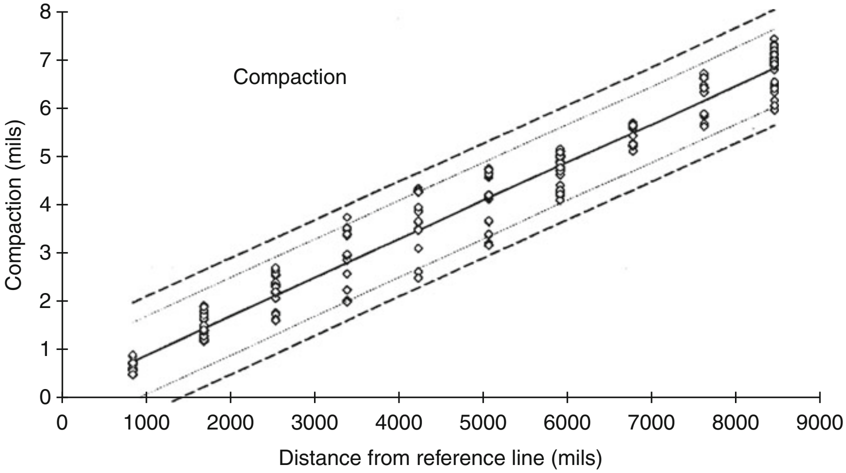


Fig. 34.5
Distance versus compaction from reference point

References

1. Shabestari BN, Miller JWV, Pawlicki H (1993) Automatic gauging using a light sectioning microscope. In: 36th Midwest symposium on circuits and systems, Detroit, 16–18 Aug 1993
2. Miller JWV, Shridhar M, Shabestari B (1995) Using Gnu C to develop PC-based vision systems. In: SPIE machine vision applications, architectures, and systems integration conference IV, Boston, 22–26 Oct 1995
3. Miller JWV, Shridhar M (1994) Hardware considerations for illumination-invariant image processing. In: SPIE machine vision applications, architectures, and systems integration conference III, Boston, 31 Oct–4 Nov 1994
4. Chu C-H, Delp EJ (Feb 1989) Impulsive noise suppression and background normalization of electrocardiogram signals using morphological operators. IEEE Trans Biomed Eng 36(2):262–273

35 Glass Thickness Measurement Using Morphology

John W. V. Miller[1] · *Behrouz N. Shabestari*[2]
[1]The University of Michigan-Dearborn, Dearborn, MI, USA
[2]Edison Industrial Systems Center, Toledo, OH, USA

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_35,

Abstract: This paper discusses simple but effective one-dimensional morphological techniques for the identification of primary and secondary peak locations associated with light patterns reflected from glass surfaces. A common optical technique for measuring glass thickness and related properties is to observe light reflected from the front and rear surfaces of the glass. Two reflections can be observed when an appropriate structured light source is used to illuminate the glass: a bright primary reflection from the front surface will be observed, along with a much fainter secondary reflection from the back. The secondary reflection is difficult to detect reliably, due to the large difference in intensity compared to the first reflection, the presence of noise, and possibly overlap between them. The methods described in the paper have been implemented successfully for two separate applications, using standard matrix and linear cameras. In both cases, the signal is pre-processed using one-dimensional morphological and linear methods, to normalize the background and remove noise. Further morphological operations are performed, to identify the peaks associated with the primary and secondary reflections.

35.1 Introduction

Morphology has been a significant and useful technique for image analysis and processing. It was initially developed for binary image processing and has been developed further to process gray-scale images by treating them as three-dimensional surfaces. [1] In this chapter, elementary gray-scale morphological concepts are applied to one-dimensional signals to find distances between primary and secondary peaks using light reflections associated with a change in the index of refraction. This is useful for detecting boundaries between different transparent materials.

The morphological techniques that are used here are based on the basic properties of erosion and dilation [2] Gray-scale *erosion* is defined as

$$(X\Theta B)(x,y) = \min_{(j,i)\in B}\{X(y+j,x+i) - B(j,i)\} \tag{35.1}$$

where X represents a gray-scale image and B is a structuring element. When the structuring element is placed over the pixel at (x, y), the difference between the image pixels and corresponding structuring element values is calculated. The minimum value becomes the value at pixel (x, y) in the eroded image. Similarly, the expression for *dilation* is

$$(X \oplus B)(x,y) = \max_{(j,i)\in B}\{X(y-j,x-i) + B(j,i)\} \tag{35.2}$$

This is similar to *erosion*, except that the pixel and structuring element values are added and the maximum value is chosen. Note that, if the structuring element is not symmetrical, its reflection about its origin must be used for erosion.

Performing a gray-scale erosion on an image produces a new image that is in most places darker than the original image. If there are small bright regions in an image that are spanned by the structuring element, they will be removed by erosion. The opposite effect is observed with gray-scale dilation: most parts of a dilated image will be brighter than the original image, while dark regions that are smaller than the structuring element will disappear.

Morphological *opening* and *closing* are obtained by using both erosion and dilation. To perform an opening on an image, it is first eroded and the resulting image is then dilated using the same structuring element. To close to an image, the operators are used in the reverse order, again with a common structuring element. An important property of opening and closing is that they have no effect on those areas of an image that lack features smaller than the structuring element. The general principle is that openings remove small bright features, while closings eliminate small dark ones. The size, shape and gray-scale values of the structuring element determine the effect that a given operation will have on an image.

In this chapter, two separate applications using morphology to identify primary and secondary reflections are presented. In the first, the measurement of dielectric thickness in AC plasma display panels, employs a light-section microscope (LSM). ➊ *Figure 35.1* illustrates the basic construction of this type of device. Dielectric material is applied over a glass substrate with thin-film metal electrodes. When the device is assembled, two parallel substrates are held a fixed distance apart by spacers. Their electrode patterns (arrays of thin parallel lines) are orthogonal to each other. Prior to final assembly, the dielectric thickness is measured, using a video camera and frame grabber with the LSM. The image of the top surface of the dielectric (glass plate) appears as a very bright horizontal line. The secondary reflection, associated with the rear dielectric–glass interface, is much dimmer. A plot of the intensity profile given in ➊ *Fig. 35.2* illustrates this. The distance between the two peaks is proportional to the dielectric

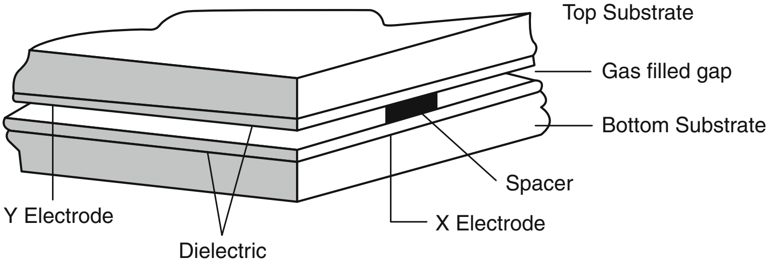


Fig. 35.1
Construction of AC plasma display panel

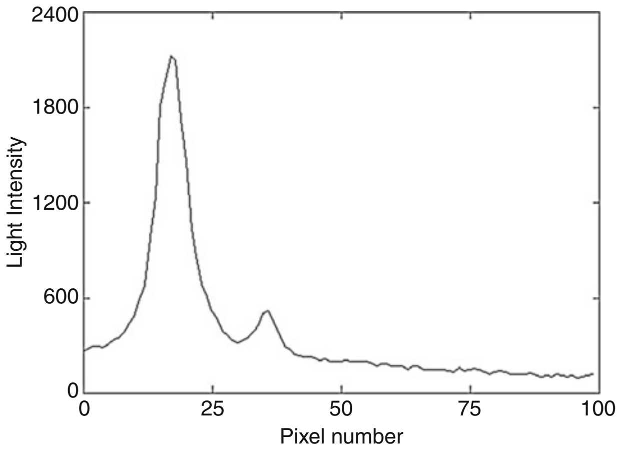


Fig. 35.2
Intensity plot of primary and secondary reflections from the AC plasma panel dielectric interfaces. Note that the *Y* axis has been logarithmically transformed

thickness. It should be noted that the difference between the heights of the two peaks is actually much greater than the plot indicates, since the data was logarithmically transformed prior to plotting as shown in ➋ *Fig. 35.2*.

The second application is similar but the data are of poorer quality, due to the manner in which the data acquisition was performed. A linear camera was used here. For reasons of confidentiality, no specific information about the application can be disclosed, except for the details of the data preprocess and finding the distance between the primary and secondary peaks.

In both applications, data analysis is performed to measure the distance between the two peaks but important differences exist. For dielectric thickness measurements, obtaining accurate measurements with limited spatial resolution was the prime objective. Averaging was used extensively so noise was not a problem but the coarse spatial quantization imposed special processing requirements. In the second application, preprocessing reduced the effects of high noise levels.

35.2 Data Acquisition

In order to achieve accurate measurements, the LSM, video camera, frame grabber, and motion-control hardware were assembled and measurement software was developed. The LSM was a critical component, since it provides images that could be analyzed to extract three-dimensional information, including dielectric thickness. A general overview of the system is in ➋ *Fig. 35.3*, illustrates the various system components.

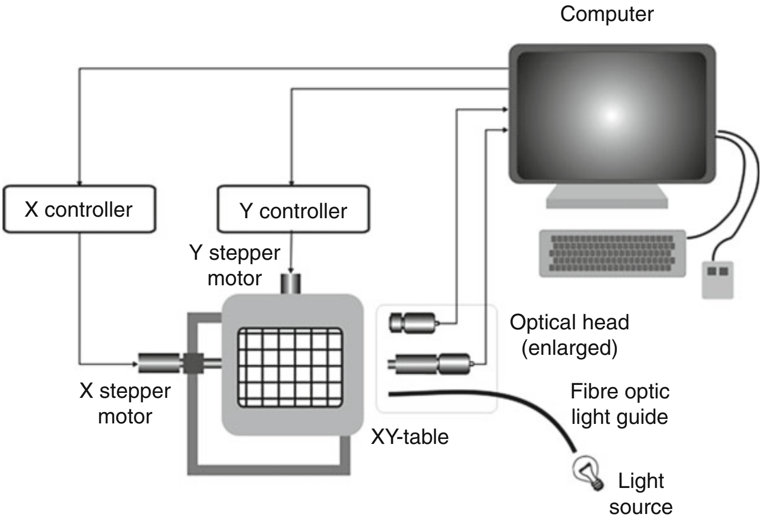


Fig. 35.3

Hardware used for dielectric thickness measurements. The box in the lower left corner provides additional information about the optical head

35.2.1 Surface Profile Measurements

The LSM produces a surface profile by making an optical "cut" using structured light. An incandescent lamp, masked by a slit is focused using an objective lens to produce a narrow band of light at 45° to the dielectric surface. The light stripe is reflected off this surface and is imaged by the microscope at 135°.

A cross-line reticule is provided in the eyepiece for manual measurements but is not used by the software. A camera port suitable for mounting a video camera is provided on the microscope. A moveable mirror is provided on the microscope to divert the image to the eyepiece or the camera.

35.2.2 Motion Control Hardware

In order to measure dielectric thickness at various locations on a glass substrate, either the substrate or microscope must be moved. Moving the substrate is easier, since it weighs less than the microscope with its video camera assembly and is unencumbered by cables. An *X*–*Y* table driven by stepper motors is used to position the substrate. The substrate is mounted on the table with a fixture that secures it in a fixed position. Given that there is some variation in the electrode pattern relative to the substrate, *X*- and *Y*-offset adjustments can be made using a fiducial reference point.

The stepper motors are driven by a pair of controllers that accept commands from the host computer via a serial port. The controllers provide the necessary velocity profile, by controlling acceleration and deceleration. This is necessary to minimize the time required for movement and avoid to positioning errors. A variety of commands may be presented to the stepper-motor controllers, including the option of making relative and absolute position adjustments. Status information is obtained by data returned from the controller after the appropriate command has been issued to it.

Since the panel substrates may not be perfectly flat, it is necessary to control the distance between the microscope objective lens and the surface being inspected. In this case, the microscope, not the substrate, is moved by small increments. As the *X*–*Y* stage moves to each new location, the microscope height is adjusted, so that the dielectric surface remains a fixed distance from it. After each new image has been acquired, the dielectric surface is identified and a correction applied to the microscope's vertical position. If the magnitude of the correction is above a defined value, a new image is acquired. However, a second image is normally only required when moving to a new spacer location.

35.2.3 Image Acquisition Hardware

A standard RS-170 camera was attached to the camera port of the LSM using a custom C-mount adapter. A combination VGA/frame-grabber card stores the digitised video signal. A "live" video image can be shown on the VGA monitor with this board so that an additional monitor is not required for setup. The VGA memory is also used for image capture and allows text and graphics to be combined with a digitised image.

35.3 General Approach for Identifying Primary and Secondary Peaks

As stated above, both applications provided measurements based on the distance between the primary and secondary peaks. The same basic technique for identifying peak values was employed in both applications although preprocessing differed considerably.

35.3.1 One-Dimensional Gray-Scale Morphology

The one-dimensional erosion and dilation operations are defined in Eqs. 35.1 and 35.2 respectively. However, in both applications, data can be analyzed using one-dimensional (signal) analysis techniques. Hence, erosion and dilation can be reduced to the following form:

$$(X\Theta B)(x) = \min_{(i)\in B}\{X(x+i)\} \tag{35.3}$$

$$(X\oplus B)(x) = \max_{(i)\in B}\{X(x-i)\} \tag{35.4}$$

Notice that the spatial extent of the structuring element is the only adjustable parameter.

As noted previously, these operations were combined to perform opening and closing operations. Using an opening operation with a structuring element wider than the base of the peaks provides an estimate of the sensor shading, or DC offset. Subtracting this from the original data generates a signal consisting only of the peak data. The results from each step in this procedure are illustrated in *Fig. 35.4*. The top curve (A) is the original data showing the primary and secondary peaks. The dashed horizontal line segments (B) below the peaks represent the processed data after an opening operation. In many locations the original and

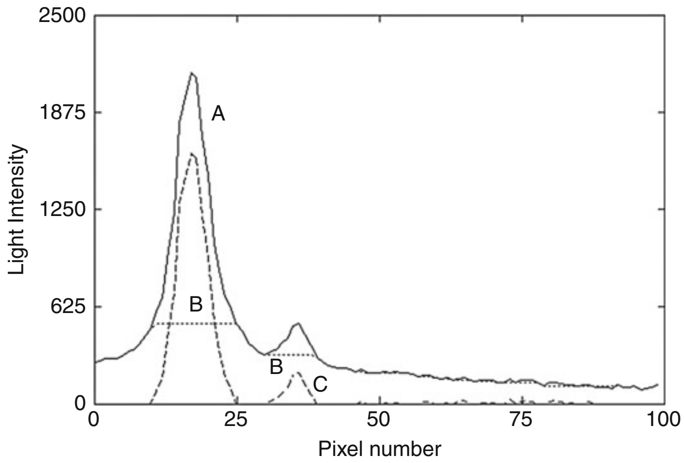


Fig. 35.4
Using morphological opening operation to remove undesired background information

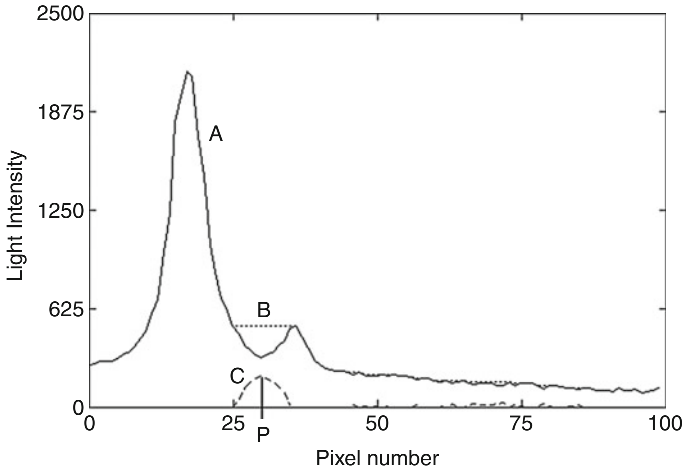


Fig. 35.5
Using morphological closing operation to identify the valley between two peaks

opened data coincide so they cannot be distinguished visually. Subtracting the opened data from the original data provides the third plot (C) in which all points are zero or nearly so except for the two peaks.

The next step is to find the location of the valley between the two peaks. The valley region is detected using a closing operation on the original data which fills in the valley as illustrated in *Fig. 35.5*. The horizontal dashed lines (B) in this figure correspond to the data from the closing. As with the opening, many of the closed data points are obscured by the original data (A). Subtracting the original data from the closed data produces the peak corresponding to the location of the valley. This is indicated by (C), which also contains some much small peaks due to noise.

The pixel number (P) corresponding to the maximum value of curve (C) in *Fig. 35.5* is used as the starting point of the search to locate the two peaks. Finding the maximum values to the left and right of P provides the locations of the two peaks.

This approach has proven to be very robust and will identify peaks correctly under a variety of conditions. The presence of noise can affect the results, so some preprocessing for noise suppression is very desirable. The next two sections describe the special processing unique to each application.

35.3.2 Dielectric Thickness Measurements

In order to measure the dielectric thickness of AC plasma panels with a light-section microscope, locations of reflections from both the dielectric surface and the dielectric/substrate interface need to be detected accurately and unambiguously. By finding the difference between the peak positions, the dielectric thickness can be estimated. Detection of the reflection from the dielectric is quite easy because the surface is highly reflective and produces a bright horizontal line in an LSM. However, the reflection from the dielectric/substrate interface is much fainter and requires processing to locate it accurately.

35.3.2.1 Noise Reduction Considerations

Video camera noise causes significant problems at low light levels, so steps must be taken to minimize it. Frame integration is one effective technique for reducing video noise; quadrupling the number of frames in a given integration sequence approximately doubles the SNR of the resulting image. The main drawback with this approach is the time penalty imposed when performing the frame integration.

A similar approach takes advantage of the nature of the image. Each column of pixels contains information degraded by noise about the two reflections. In principle, each column of the image could be summed, to provide a low-noise estimate of the light profile through the two reflections.

However, unless the camera is very carefully aligned with the horizontal reflections, the resulting estimate will be degraded and the peaks of the two reflections will be "broadened". This can be alleviated by summing over smaller vertical regions and obtaining a number of low-noise estimates of the light profile across the image.

35.3.2.2 Calculating Dielectric Thickness

Although an estimate of dielectric thickness can be obtained by finding the distance between the positions of the maxima of the two peaks, the results do not achieve the desired precision. Errors caused by quantization effects are large because the separation between peaks is only about 20 pixels, corresponding to 38 μm. The desired precision of the thickness measurement was ±0.25 μm. This level of precision was achieved by calculating the first order moment about each peak.

In order to estimate dielectric thickness, the procedures described previously are used in sequence. Frame integration reduces the effects of camera noise. To reduce noise further, the image is divided into a number of vertical strips and all columns of pixels in a given strip are added together. Morphological processing is then used to find the primary and secondary peaks for a given image strip and the first order moment is calculated about each peak to reduce quantization errors. In the last step, the dielectric thickness estimates for each image strip are averaged to obtain a better estimate.

35.3.3 Proprietary Application

It was not feasible to average a number of scan lines together to suppress noise for the proprietary application. In addition, the linear camera exhibited a significant amount of nonuniformity between odd and even pixels, which acted like a synchronous noise source. The top trace (A) in ❯ *Fig. 35.6* represents typical data with the odd/even pixel nonuniformity evident in low-gradient regions.

Opening and closing operations with five-element structuring elements were used to remove this noise. The procedure previously described was used to find the peaks. The lower trace (B) in ❯ *Fig. 35.6* illustrates the result. Once the valley between the two peaks was identified, the primary and secondary peaks could be separated. As before, the use of one-dimensional morphology provided a reliable and robust means for primary and secondary peak detection.

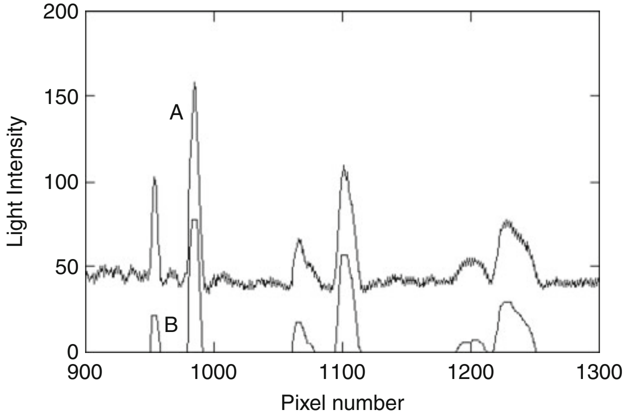


Fig. 35.6
Typical data from the proprietary applications. The original noisy signal (A) was processed to obtain the bottom trace (B)

Acknowledgments

This work was funded by Electro Plasma, Inc., the Libbey-Owens-Ford Co., and the Advanced Display Consortium.

References

1. Dougherty ER (1992) An introduction to morphological image processing, vol TT9, Tutorial texts in optical engineering. SPIE Optical Engineering, Bellingham
2. Abbot L, Haralick RM, Zhuang X (1988) Pipeline architectures for morphologic image analysis. Machine Vis Appl 1:23–40

36 Inspecting Food Products

E. Roy Davies
Royal Holloway, University of London, Egham, Surrey, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_36,

Abstract: An important aspect of machine vision is the inspection of products during manufacture, in order to achieve control of quality. This is as necessary for food products as it is for metal or plastic parts, though there is a difference in that food products are highly variable: e.g., even two fairly ideal apples will have marked shape, colour and other differences, and this will have to be taken into account when devising inspection algorithms. Another important aspect of food inspection is the speed with which product location has to be carried out, as the process of location involves unconstrained search: contrariwise, product scrutiny often involves examining a relatively small area of an image, and can often be carried out much more quickly. To understand the process of algorithm design, this chapter takes a number of case studies relating to the location of objects in food and cereal images. These range from finding Jaffacakes, cream biscuits and cereal grains to identifying insects, rodent droppings, moulds and foreign seeds amongst the cereal grains. In the first three cases the objects had to be scrutinised for defects, whereas in the other cases the cereal grains had to be scrutinised for contaminants.

36.1 Introduction

It is many years since computers were first used for analysing digital images, and thousands of industrial, medical, space, forensic, surveillance, scientific and other applications have now been recorded. This huge subject area is called machine vision and is characterised by the use of artificial vision to achieve actions in the real world. Thus it is a rather separate subject from computer vision, which endeavours to understand visual processes by experiments which test algorithms to see how well they permit vision to be achieved, and to determine what principles are involved in the process. In a sense, computer vision is the science and machine vision is its engineering colleague. However, this does not mean that machine vision has to be unscientific. Far from it: machine vision necessarily takes on board all the methodology and conclusions of computer vision, but also aims to arrive at cost-effective solutions in real applications. As a result, there is some emphasis on producing more efficient solutions, which are not only effective but to some extent minimal, with viable but rigorous short cuts being taken to save computation and hardware cost, but without significant invasion of robustness, reliability, accuracy, and other relevant industrial parameters. The way in which this end can be achieved is by noting that many applications are constrained in particular ways, so the full panoply and power of the most comprehensive algorithms will not always be needed. Naturally, this implies that some loss of adaptability in the chosen algorithms may have to be accepted in any one case. However, this is by no means unusual in other types of application, and in everyday life we are used to selecting appropriate tools to handle any task we tackle. Thus machine vision is intimately concerned with the selection of software tools and algorithms, and furthermore it is concerned with the specification and production of such tools.

Unfortunately, although the subject has developed vastly and even impressively over the past decades, it cannot yet be regarded as a fully mature subject. Software specification and design are not yet possible to the extent that they are in other areas of engineering. The blame for this lies in the complexity and variability of the image data that machine vision systems are expected to handle. Not only are the image signals highly variable, but also the noise within the images is highly changeable. For instance, noise may be white or “coloured,” Gaussian or impulse, position or time dependent. In addition, it may arise in the form of clutter, i.e., it may take the form of other objects which partly occlude those being observed, or other objects forming a variegated or even a time-varying background. In fact, the variability of real imagery is more often due to the effects of irrelevant objects than it is to noise, though noise itself does play a significant part in complicating the overall situation. Lastly, the objects being observed can vary dramatically, not only in their intensity, colour or texture, but also in their shape (especially when the 3D viewpoint is variable) and in changes that are the result of defects or damage.

To develop the subject further, it is necessary to take account of all these factors, and to try to understand the specification–application problem in more detail. However, theory on its own is unlikely to permit this to be achieved. The way forward seems to be to take a careful record of what happens in particular applications and to draw lessons from this. As more such records become available, sound all-embracing theories and guidelines will gradually emerge. However, at the very least, such case studies will permit design by analogy and by example, so very little should be lost. The case study approach is a powerful one which is particularly relevant at this point in the development of the subject. Accordingly, this chapter describes a number of case studies from which lessons can be drawn.

In fact, the subject of machine vision is very wide indeed, and there is a limit to what a single chapter can achieve. Hence it will be useful to consider a restricted range of tasks: by focusing attention in this way, it should be possible to make more rapid progress. On the other hand, many tasks in different areas of vision bear significant similarity to each other – notice the ubiquity of techniques such as edge detection, Hough transforms, neural networks, and so on – so the possibility of transfer of knowledge from one part of the subject to another need not be substantially reduced. For these reasons, this chapter is targeted at automated visual inspection applications, and in particular some tasks which relate to food products. Foodstuffs have the reputation of being highly variable, and so the tasks described here cannot be considered trivial – especially, as we shall see, because quite high rates of inspection are required.

In the following section we first describe the inspection milieu in rather more detail, and then we proceed to a number of case studies relating to food and cereals. Selected theoretical results are also considered at relevant junctures. Finally, the concluding section of the chapter underlines the lessons learnt from both the case studies and the theory, and shows that both are relevant to the development of the subject.

36.2 The Inspection Milieu

Computer vision is widely considered to have three component levels: *low-level vision*, *intermediate-level vision*, and *high-level vision* [1, 2]. Low-level vision consists of primitive processes such as removal of noise and location of image features such as edges, spots, corners and line segments. Intermediate-level vision involves the grouping of local features into boundaries or regions, and also the identification of geometric shapes such as circles. High-level vision analyses the purely descriptive groupings resulting from low and intermediate-level vision and makes careful comparisons with objects in databases: relational analysis, inference, reasoning and statistical or structural pattern recognition techniques are used to help achieve viable matchings, and the image features are finally identified as actual objects – together with various attributes such as motion parameters and relative placements.

As mentioned in ➋ Sect. 36.1, machine vision aims to go further and identify actions which should be initiated as a result of visual interpretation. However, it is also able to limit the practical extent of the algorithms at a level below that represented by the computer vision ideal, so that efficiency and speed can be improved. This means that instead of focusing on low, intermediate and high-level vision, we should concentrate on achieving the task in hand. In the case of industrial inspection, this means attending first to object location and then to object scrutiny and measurement. However, the very first task is that of image acquisition. This is especially important for inspection, as (a) careful illumination can ensure that the image analysis task is much simplified, so fewer advanced algorithms are needed, thereby saving computation and speeding up implementation; and (b) careful illumination and acquisition can ensure that sufficient information is available in the input images to perform the task in hand [3]. Here it has to be borne in mind that object recognition is less problematic when only one type of object can appear on a specific product line: the natural restrictions of machine vision permit the whole process to be made more efficient.

Overall, we now have the following four-stage inspection process:

1. Image acquisition
2. Object location
3. Object scrutiny
4. Rejection, logging or other relevant actions

Here the "object" may be the entire product, or perhaps some feature of it, or even a possible contaminant: note that in the first of these cases the object may have to be measured carefully for defects, while in the last case, the contaminant may also have to be identified. The situation is explained more fully in the case studies.

Set out in this way, we can analyse the inspection process rather conveniently. In particular, it should be noticed that object location often involves unconstrained search over one or more images (especially tedious if object orientation is unknown or if object variability is unusually great), and can thus involve considerable computational load. In contrast, object scrutiny can be quite trivial – especially if a pixel by pixel differencing procedure can be adopted for locating defects. These considerations vary dramatically from one application to another, as the case studies will show. However, the speed problem often represents a significant fraction of the system design task, not least because products are likely to be manufactured at rates well in excess of 10/s, and in some cases such as cereal grain inspection, inspection rates of up to 300 items/s are involved [4].

36.3 Object Location

In the previous section the problem of limiting the computational load of any vision algorithm was mentioned, with the indication that the demands for meeting real-time processing rates could be quite serious in food processing applications. In fact, this difficulty is inherent in the object location phase of any inspection process. The reason is that unconstrained search through one or more images containing some millions of pixels may be required before any object is encountered. Furthermore, at each potential location, a substantial template may have to be applied with a large number of possible orientations. If objects of 10,000 pixels are being sought, and 360 orientations can be defined, some 10^{12} pixel operations may have to be performed – a process which will take a substantial time on any computer. This clearly rules out brute force template matching, but turns out not to eliminate template matching per se, as it can instead be applied to small object features. Often the template can be as small as 3 × 3 pixels in size, and there will be a correspondingly small number of definable orientations, so the computational load can immediately be reduced to around 10^7 pixel operations for a 256 × 256 pixel image – a much more realistic proposition. However, this leaves the problem of how to locate objects from their features – a process that involves inference rather than logical deduction: it should be noted that inference can itself lead to computational problems and to potential ambiguities, unless care is taken in the choice of the relevant algorithms. ➲ Chapter 18 considers the situation in depth, and arrives at several general approaches to the problem. These include the Hough transform, which is particularly appropriate when edge features are used to locate objects, and graph matching, which is appropriate mainly when point features are used for the purpose.

36.3.1 Feature Detection

Before proceeding to discuss inference procedures in detail, we briefly examine how feature detection can be achieved. The most commonly used approach involves the use of convolution masks. If a feature of a certain shape is to be detected, this is achieved by applying a mask of the same intensity profile – insofar as this is realisable within a digital (pixel) representation. The method is amply illustrated by the case of corner detection. If corners are to be detected, convolution masks such as the following may be used (To be rigorous, the convolution mask *enhances* the feature, and this then has to be *detected* by thresholding or related operations [1]. In this section we follow common practice by referring to feature detection without further qualification of this sort.):

$$\begin{bmatrix} -4 & -4 & -4 \\ -4 & 5 & 5 \\ -4 & 5 & 5 \end{bmatrix}$$

Clearly, such a mask will only be able to detect corners of this particular orientation, but application of eight similar masks whose coefficients have been "rotated" around the central pixel essentially solves the problem. Notice how such masks are given a zero sum so that they can achieve a modicum of invariance against variations in image illumination [5]. Interestingly, while corners and other general features require eight such masks within a 3 × 3 window – and rather more in larger windows – edge detection only requires four masks of the following general types:

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

This is because four of the eight masks that might have been used differ from the others only in sign. However, if we permit ourselves to change the strategy for edge detection from pure template matching by noting that edges are vectors with magnitude and direction, it becomes possible to employ just two masks: these then detect the horizontal and vertical components of intensity gradient. The pattern of ±1 coefficients employed in the above edge detection mask is not the best possible, since it leads to inaccurate estimates of edge orientation – a parameter that is of particular importance for Hough transform and related methods of object location. Within a 3 × 3 window the most appropriate edge detection masks are thus those of the well-known Sobel operator:

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{36.1}$$

for which magnitude and direction can be estimated from the gradient components g_x, g_y using the equations:

$$g = (g_x^2 + g_y^2)^{1/2} \tag{36.2}$$

$$\theta = \arctan\left(g_y/g_x\right) \tag{36.3}$$

Next, we come to line segment detection. Line segments are not vectors as symmetry ensures that any vector that is associated with them will have zero magnitude. This suggests that we should employ an eight-mask set for line segment detection, though symmetry considerations reduce this set to just four masks. However, recent research has shown that if the following pair of masks is used [6]:

$$L_0 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix} \quad L_{45} = \begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix} \tag{36.4}$$

line features can be detected by combining the two outputs in a vectorial manner. The situation is curious, but this operator acts within an abstract space where the normal 360° rotation is transformed to 180°, with the result that the operator predicts a line segment orientation ψ which is twice as large as expected. However, dividing ψ by 2 leads to sensible results as symmetry only permits line orientations to be defined in the range 0–180°.

Finally, note that hole and spot detection masks only require a single convolution mask to be applied:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

36.3.2 Design of Template Masks

So far, we have seen that much of the weight of object location devolves onto the use of template masks for feature detection. While design of such masks can be dealt with intuitively and checked by experiment, design principles are hard to come by. The *zero mean rule* [5] is a generally useful one, but important questions remain – such as how large the masks should be and how the pixels should be divided between the foreground and background regions.

In fact, it is generally the case that greater sensitivity of detection results from the additional averaging achieved by larger masks. However, there is a limit to this which is determined by the probability of disruption introduced by impulse noise [7]. Specifically, if one pixel within a mask succumbs to impulse noise, this may well falsify the response from that mask, and (for example) cause a vote to be placed at a totally erroneous position in the Hough transform parameter space. Clearly, the probability P of this happening increases rapidly with the number of pixels within the mask. If the probability of a pixel intensity being corrupted in this way is p, and the area of the mask is A, we find [7]:

$$P = (1 - p)^A \tag{36.5}$$

If we set P at some reasonable limit of acceptability, such as 0.75, and if p is taken to have a value of about 0.01, we find that $A \approx 28.6$, and in that case the window size should not be larger than about 5×5.

Once the window size has been determined, the problem of disposition between background and foreground remains. A simple calculation can be used to solve this problem [8]. We first write the mask signal and noise power in the form:

$$S = w_\mathrm{f} A_\mathrm{f} S_\mathrm{f} + w_\mathrm{b} A_\mathrm{b} S_\mathrm{b} \tag{36.6}$$

$$N^2 = {w_\mathrm{f}}^2 A_\mathrm{f} {N_\mathrm{f}}^2 + {w_\mathrm{b}}^2 A_\mathrm{b} {N_\mathrm{b}}^2 \tag{36.7}$$

where A_f, A_b are the numbers of pixels allotted to the foreground and background regions, w_f, w_b are the respective mask weights, S_f, S_b are the respective signal levels, and N_f, N_b are the respective noise levels. Optimising the signal-to-noise ratio for a zero mean mask of fixed total area $A = A_\mathrm{f} + A_\mathrm{b}$ now leads to the equal area rule:

$$A_\mathrm{f} = A_\mathrm{b} = A/2 \tag{36.8}$$

though complications arise when $N_\mathrm{f} \neq N_\mathrm{b}$, a situation that becomes important when the foreground and background textures are different. Finally, when the equal area rule does apply, the mask weights must follow the rule:

$$w_\mathrm{f} = -w_\mathrm{b} \tag{36.9}$$

Thus theory can help substantially with the design of masks, once the design constraints are known reliably.

36.3.3 The Hough Transform

The Hough transform is a model-based inference technique which may be used for locating objects of known shape. It has been used in many applications for locating lines, circles, ellipses, and shapes composed of these basic structures. It is also known how to apply the Hough transform to locate arbitrary shapes [1]: however, these shapes must first be defined unambiguously, and this may be achieved using analytical formulae or with the aid of look-up tables. The basic principle of the Hough transform is described in detail in ➋ Chapter 18, together with its application to detect lines, circles, ellipses and general shapes. However, it will be useful to note here that the Hough transform is particularly suited to inferring the presence of objects from their edges. It is also useful to point to the particular advantage of the Hough transform, and indeed of more recent, related projection-based transforms [9]: this resides in its impressive robustness against noise, occlusion and object distortion – or even breakage. It is achieved because edge pixels arising from noise, occluding objects or distorted parts of the object boundary give rise to candidate centre locations that are spread out in parameter space: these are unlikely to focus on particular locations and to give rise to misleading solutions. In any case, quick tests will usually eliminate any false positives that may arise (see the detailed discussion in ➋ Chap. 18).

36.3.4 The Maximal Clique Graph Matching Technique

Graph matching provides another approach to model-based inference which may be used for object location. However, the graph matching approach is aimed at locating objects from point features such as corners and small holes rather than from edge or line segments which are only localised in a single dimension.

The concept of graph matching is to imagine the point features in the image as forming a graph, and those in the model as forming a separate graph. The matching problem then amounts to finding the isomorphism with the largest common subset in the two graphs, as this represents the greatest degree of agreement between the model and the image data. The solution is represented by a subset of the model graph because part of any object in the image may be occluded or broken away; likewise, it is represented by a subset of the image data as there will usually be features from other objects or from noise which have to be ignored.

One way of achieving graph matching is by the maximal clique technique. Because this and related graph matching techniques search for valid subsets of the input data, they are, like the Hough transform, robust against noise, occlusion and object distortion [1]. Unfortunately, graph matching per se can be computation intensive, and methods have been described for using the generalised Hough transform to achieve the same end, with far less computation. Details of this approach are presented in ➋ Chap. 18, and will be considered further below.

36.4 Case Study: Jaffacake Inspection

Jaffacakes are examples of food products with high added value, as a significant number of processes are required during their production [10]. In addition, they are enrobed with a substantial layer of chocolate – an expensive commodity. These factors mean that, ideally, each stage of manufacture should be monitored by an inspection system to ensure that further value is not wasted on any defective products. However, inspection systems themselves cost money, both in the necessary hardware and software, and also in the need for careful setting up, calibration, and continual re-validation. In such cases it is usual to employ a single inspection system: there is some advantage from placing this at the end of the line so that the product is given a full check before it finally goes off to the consumer. This was the situation for the Jaffacake inspection system described here. Note, however, that in other applications a different judgement may be made – as in the case of cereal grain inspection (see ➋ Sects. 36.6–36.8), where it is crucial to inspect the raw commodity.

Jaffacakes are round shallow cakes resembling biscuits which are topped by a small quantity of jam before they are enrobed with chocolate (➋ *Fig. 36.1*). Cooling is necessary before packing to ensure that the chocolate has solidified: this also helps the final inspection process as liquid chocolate gives rise to glints which could confuse the inspection software.

Many problems arise with such products. First, the spread of cake mixture depends on temperature and water content, and if this is not monitored carefully the products may turn out too large or too small. While 10% variation in diameter may be acceptable, larger variations could lead to packing machines becoming jammed. Second, the jam or the chocolate could cause products to stick together, again causing problems at the packing stage. These factors mean that a variety of parameters have to be monitored – roundness, product diameter, the

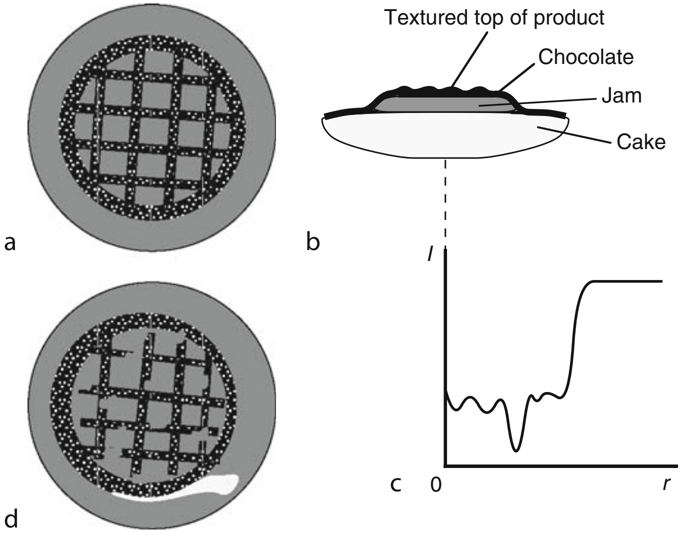


Fig. 36.1

Jaffacake characteristics. (a) shows an idealised Jaffacake on which the boundary of the jam appears as a slanted, darker region. (b) shows its cross section, and (c) shows its radial intensity histogram to the same scale as in (b). (d) shows a less ideal Jaffacake in which some "show-through" of the cake appears, and the surface texture is more fuzzy. © M^{2}VIP 1995

diameter of the spot of jam, the presence of excess chocolate, and any "show-through" (gaps in the chocolate). In addition, the product has to be examined for general appearance, and in particular the textural pattern on the chocolate has to be attractive.

To achieve these aims, the products first have to be located and then suitable appearance matching algorithms have to be applied. In fact, the Hough transform provides an excellent means of locating these round products, and is sufficiently robust not to become confused if several products are stuck together, as all can be positively identified and located. To perform the matching several processes are applied: assessment of product area, assessment of degree of chocolate cover (area of dark chocolate region), computation of radial intensity histogram (This sums and averages the intensities of pixels within concentric rings around the centre of the product [1].) to assess the jam diameter and offset (determined from the region bounded by the very dark ring on the chocolate) and the overall appearance of the product (measured from the radial intensity histogram correlation function) (➲ *Fig. 36.1*).

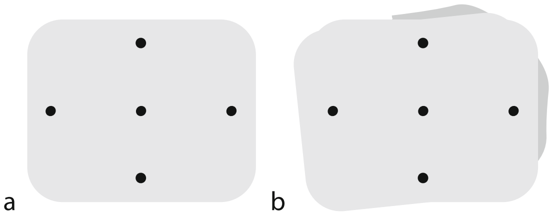
Considerable speed problems were encountered during this work, and special hardware accelerators had to be devised so that inspection could take place to perform 100% inspection at around 20 products/s. Hardware design was eased by making extensive use of summation processes, such as area measurements, computation of radial intensity histograms, and computation of the radial intensity histogram correlation function [10]. Finally, some care was required to obtain uniform lighting: this was achieved by using a set of four lights symmetrically placed around the camera. While lighting did not have to be set critically, because of the robustness of the algorithms used, reasonable uniformity in the lighting was useful to minimise the number of separate measurements that had to be made on each product.

36.5 Case Study: Inspection of Cream Biscuits

This case study describes the inspection of cream biscuits which consist of two wafers of biscuit with cream sandwiched between them (➋ *Fig. 36.2*). There are two particular reasons for inspecting these biscuits: one is to check on the alignment of the two biscuit wafers; the other is to check whether there is substantial leakage of cream from between the wafers. Clearly, both of these aspects have a distinct influence on the appearance of the product and its acceptability to the consumer. They also reflect the possibility of jamming the packing machine.

Conventional shape analysis techniques can be used to check the overall shape of the final product and thus detect these defects. However, the overall shape can be quite complex even if small amounts of misalignment or cream are evident (➋ *Fig. 36.2*). Hence considerable simplification of the overall algorithm results from adopting an alternative strategy – to locate the biscuits by the small "docker" holes on the upper wafer, and then to check the overall shape against a template [1]. In fact, a further simplification is to determine how much of the product lies outside the region of the upper wafer as computed from the positions of the docker holes (➋ *Fig. 36.2*).

To achieve this, it seemed appropriate to employ the maximal clique technique as it is suited to location of objects from point features. While this worked well with biscuits containing up to six docker holes, it was found to be far too slow for biscuits with many holes (in some cases biscuits are decorated with patterns of well over 100 holes) [4]. The reason for this is that the maximal clique computation is "NP-complete," and this results in its having a computational load which varies approximately exponentially with the number of features present. Because of this, other procedures were tried, and it was found that the Hough transform could be used for this purpose. To achieve this, a reference point G was selected on the biscuit pattern (the centre of the biscuit was a suitable point), and all pairs of features located in the image were used to estimate the position of G, the votes being placed in a separate parameter space. This permitted the positions of all biscuits to be found with much the same degree of effectiveness as for circle location. The only problem was that it was not known a priori which was which of the holes in each pair, so two votes had to be cast in parameter space in each case [11]. While this gave a considerable number of false votes, no difficulty was experienced in determining the centre of each biscuit by this method. For further details of the technique, see ➋ Chap. 18.



◘ Fig. 36.2

Problems encountered in inspecting cream biscuits. (a) shows the plan view of an ideal cream biscuit, and (b) shows a less ideal product with misaligned wafers and leakage of cream. The "docker" holes by which the upper wafer can be located are clearly indicated

36.5.1 Problems with Maximal Cliques

The computational problems with the maximal clique approach have been known for some time, though as noted above they were not a particular problem for biscuits with up to six features. However, in this application another problem with the technique was identified. This arose because in the end the technique only makes use of features that are part of the largest available maximal clique: features which are not compatible with all the other features in the maximal clique that is finally taken to represent the object are in the end ignored. Unfortunately, distortions can result in some features not being compatible with a limited number of other features on the same object, while still being compatible with the remainder. In that case some information which is available from these features and which would be useful for interpretation is ignored.

As the method is potentially highly robust, it may still result in the object being recognised. However, when only a small number of features are available on any object, there is little latitude for ignoring any of them and some loss of robustness then results. A further factor is the loss of location accuracy that results from the decrease in position data. In general, then, the maximal clique method does seem to lose out significantly relative to the Hough transform approach when locating distorted objects. Food products are quite variable and are often slightly distorted during manufacture, so they are especially affected by these problems. Specifically, changes in the fluidity of the biscuit mixture may lead to size and shape changes which can push a number of inter-feature distances outside accepted tolerance limits.

Naturally, many distortion problems can be overcome by permitting a certain tolerance in the inter-feature spacings. However, if too much tolerance is permitted, this ultimately results in a large increase in the number of false alarms (whether these are recorded as votes in parameter space or lines in the match graph). Hence in practical situations, with given tolerance levels, the maximal clique approach results in reduced levels of robustness and location accuracy.

36.6 Case Study: Location of Insects in Consignments of Grain

Automated visual inspection has now been applied to a good many food products, and as indicated in ➋ Sect. 36.4, it is often appropriate to apply it at the end of the production process as a final check on quality. Nevertheless, there is a distinct need to attend to the quality of the raw products, such as the grain from which flour, breakfast cereals and many other derived products are made. In the case of grain, attention has so far been concentrated on the determination of grain quality [12], the determination of varietal purity [13, 14] and the location of damaged grains [15]. Relatively little attention has been paid to the detection of contaminants. In fact, there is a need for an inexpensive commercial system which can detect insect infestations and other important contaminants in grain.

This case study describes research which has been orientated towards the design of such a system. Of especial interest is the location of insects amongst the grains, as these have the potential for ruining a large store of grain in relatively few weeks, because of the rather short breeding cycle. This means that a highly reliable method is needed for detecting the insects, involving a far larger sample than human inspectors can manage: in fact, a sample of some 3 kg (∼60,000 grains) for each 30 t lorry load of grain appears to be necessary. As lorries arrive at intervals ranging from 3 to 20 min, depending on the type of grain terminal, this represents a significant challenge from both the effectiveness and the computational speed points of view [16, 17].

Preliminary analysis of the situation suggested that insects would be detected highly efficiently by thresholding techniques, as they appear dark when compared with light brown wheat grains. However, careful tests showed that shadows between the grains and discolorations on them can produce confusing signals, and that further problems can arise with dark rapeseeds, fragments of chaff, and other artefacts (➋ *Fig. 36.3*). This meant that the thresholding approach was non-viable, though further analysis showed that line segment detector algorithms (➋ Sect. 36.3.1) were well adapted to this application because the insects approximated to short rectangular bars. It turned out that high orientation accuracy is not required in this instance, though it is attractive to use just two vectorial masks to save computation.

The next significant problem was to design larger versions of these masks so as to obtain good signal-to-noise ratio at higher resolution with insects about 3 pixels wide. This was achieved with 7 × 7 masks which approximated to rings of pixels, for which about half of each mask would be in the background and half within the insect region. Geometric arguments

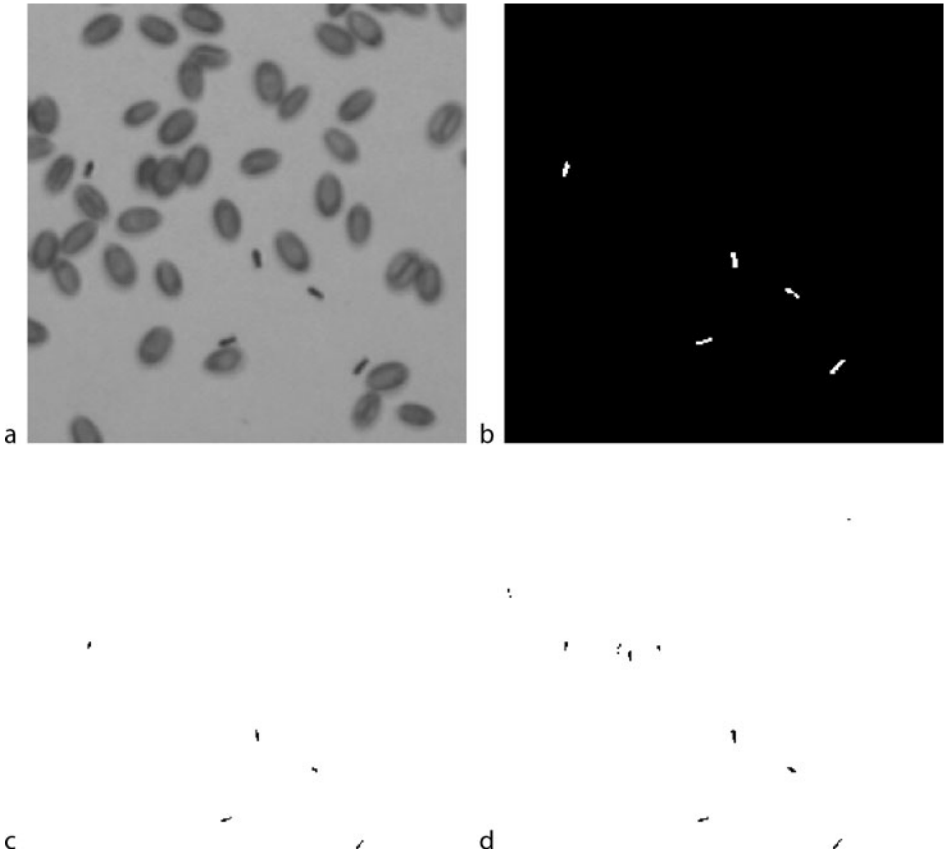


Fig. 36.3

Insects located by line segment detector. (a) Original image. (b) Result of applying line segment detection algorithm. (c) Result of selecting optimum (minimum error) threshold level on (a). (d) Effect of small increase in threshold. © EURASIP 1998

showed that this would be achieved when the mask radius was about $\frac{1}{\sqrt{2}}$ of the width of an insect (➋ *Fig. 36.4*).

Tests showed that the strategy was successful on images containing real grains and insects such as *Oryzaephilus surinamensis* (saw-toothed grain beetle). To perform the final detection, thresholding of the line segment operator signal was required; in addition, a test had to be applied to check that the signal resulted from a dark rather than a light patch in the image, as the line segment detector is sensitive to both. This procedure resulted in a false negative rate of around 1% against a total of 300 insects: minor detection problems arose from insects being viewed end-on or being partly obscured by grains. Another problem was a small false positive rate arising from the dark boundaries on some of the grains and particularly from "black-end-of-grain." Other problems occasionally arose from pieces of chaff resembling insects (➋ *Fig. 36.5*): one solution to this problem would be to increase resolution to improve recognition, though this would give rise to a drop in the processing rate and a faster, more expensive processor would be needed. Such issues are beyond the scope of this case study.

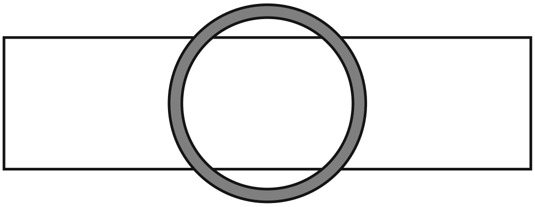


Fig. 36.4
Geometry for design of a line segment detector using ring masks. For optimum results the ring must be symmetrically placed and of such a size that it lies half within and half outside the rectangular object to be detected

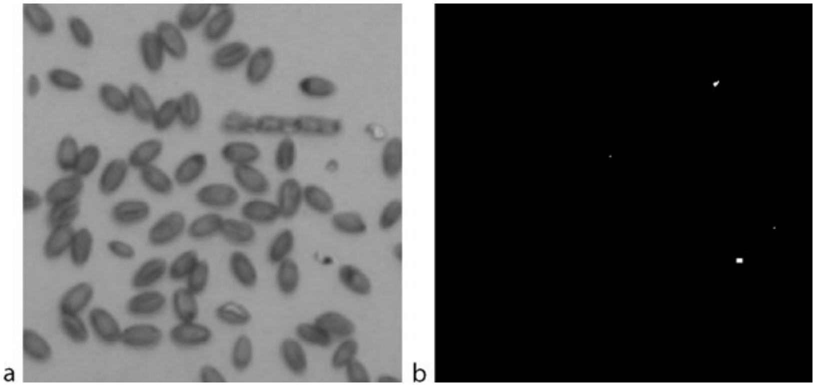


Fig. 36.5
Incidence of false alarms. (a) original image; (b) thresholded output from line segment detector. The false alarms are due to chaff and the darkened boundaries of grains: all are easily detected and eliminated via their shapes. For the specific type of line segment detector used here, see [17]. © IEE 1999

However, the line segment detector approach seems the right one to solve the problems of insect detection; and the vectorial strategy seems capable of limiting computation with the larger masks required in this application [16].

It would be interesting at this point to compare the work described above with that of Zayas and Flinn [18] who adopted a discriminant analysis approach to the insect detection problem. However, their work is targeted primarily at a different type of insect, *Rhyzopertha dominica* (the lesser grain borer beetle) and it would appear to be difficult to make a realistic comparison at present. More generally, the reader is referred to an interesting appraisal of different ways of finding pests in food [19].

36.6.1 Obtaining Improved Capability for Discriminating Insects

While the line segment detector type of approach worked very well, further improvements in any technique are always welcome, and in this case it seemed worth aiming for improvements in the discriminability of insects. In this regard, a template matching (TM) approach seemed to represent the ultimate in capability. However, the whole point of using the vectorial line segment detector approach had been its high efficiency, whereas the TM approach requires a large number of substantial masks for whole-object detection, thereby entailing excessive computation. However, it seemed that it might be possible to employ a two-stage system that would combine the speed of the line segment detector with the accuracy of TM. To achieve this, potential object points were identified using the line segment detector, and then the TM was applied to finally locate the object points. It was found to be important to keep the line segment detector threshold low so that no insect points were eliminated at that stage: the task of eliminating any false positives thus fell squarely on the TM operator. The final result was an error rate exactly matching that of the TM operator, but a considerably reduced overall execution time – composed of the line segment detector execution time plus the TM execution time for the set of regions of interest. One source of error was found to be fragmented insects, but it was found that employing a dilation operation within the line segment detector, before applying the final TM operator, could largely eliminate this problem.

The overall technique gave a welcome improvement in speed of operation – typically by a factor ∼10 – with no loss in accuracy, because a lower line segment detector threshold could be avoided. The technique was also found to be excellent at discriminating insects from dark regions on the grains and reducing the false alarm rate to a negligible level. In fact, it was able not only to detect the insects but also largely to maintain them at the correct size and shape. This was a crucial aspect of the overall inspection system, as the final stage of confirming the presence of insects and identifying artefacts such as chaff, foreign seeds and other items was via shape and size. Thus the overall insect detection scheme is a three-stage algorithm incorporating a binary shape analyser (➲ *Fig. 36.6*). It should perhaps be remarked that a further factor influencing the overall design is the need to be able to detect insects of a variety of sizes, and not just the main target, *O. surinamensis*. Details of how this was achieved will become clear in ➲ Sect. 36.7.

In this work [20, 21], a number of methods for the detection of insects in grain images were tested: their performance is summarised in ➲ *Table 36.1*. On the one hand there are methods based on template matching of whole bars, and variations that are designed to save computation. On the other hand there are methods based on edge detection (see [22]) that deduce or infer the presence of the whole object. In principle, these two approaches are merely different representations of the same underlying task and taken to their individual limits might be

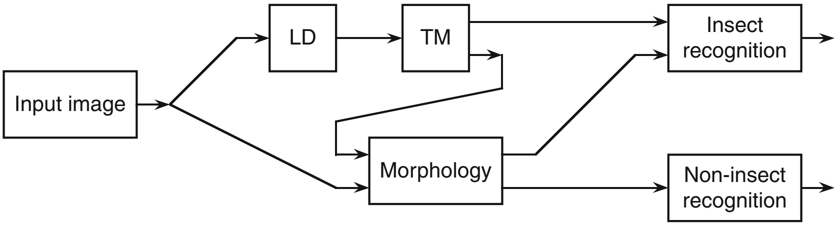


Fig. 36.6
Complete three-stage inspection system. This figure shows how morphology is used both as a component of the insect detection system and for non-insect detection. This aspect is also valuable in providing the capability for detecting larger types of insect. LD is the line segment detection module, and TM is the template matching module. © IMechE 2003

Table 36.1
Performance of the various operators on the grain images

Method	Sensitivity[a]	False alarm performance	Speed
Thresholding	Fair	Very poor: uncompetitive	Extremely high
Vector-based	Good	Good	High
Edge-based	Good	Good	High
TM	Very good	Very good	Very slow
2-stage TM	very good	Very good	Very high
Hough-based	Fair[b]	Poor[b]	Moderately high[b]

[a] Sensitivity means capability for detecting insects: for crucial false-alarm performance see next column
[b] Not tested on these data: performance deduced from the characteristics of the data; not a natural representation for this type of fuzzy, distorted, noisy data. © IMechE 2003

expected to yield the same results. In particular, both approaches in one way or another embody the spatial matched filter paradigm. However, if gross shape distortion or occlusion were to be a crucial factor, a Hough transform approach based on edge detection might prove more sensitive. Nevertheless, as a starting approximation, the Hough transform approach proved to be limited, because the insect profiles are fuzzy, distorted and noisy, and there is significant probability of interference from nearby grains. As a result, edge detectors of low support are needed, and it is difficult to use Hough transform or other methods when objects have fuzzy, ill-defined edges. (The reason for this is that it works on the basis of feature detection followed by inference, while whole-object TM performs all of this in a single integrated process.)

36.7 Case Study: Location of Non-insect Contaminants in Consignments of Grain

As noted above, there is a need for a commercial system for the inspection of consignments of grain. In fact, grain is subject to many contaminants other than insects. In particular, rat and mouse droppings have been reported, and moulds such as ergot are especially worrying as they are toxic to humans. There is also a need to keep a check on foreign seeds such as rape, though their presence is not especially worrying at low concentrations. It turns out that rodent droppings and ergot are dark and have characteristic elongated shapes, while rape seeds are dark and round: all these contaminants are readily detected by human operators. This again suggests that such contaminants could be detected automatically by thresholding. However, rodent droppings are often speckled (➔ *Fig. 36.7*), while ergot is shiny and tends to exhibit highlights, so again it has light patches – albeit for different reasons. There is also the possibility of confusion between the speckled regions and the grainy regions which are light but with a variety of dark patches caused by shadows and some dark patches on the grains themselves [23].

These considerations suggest that morphological techniques [24] should be used to help with detection of the dark contaminants mentioned above. More particularly, as human operators are able to recognise these contaminants from their characteristic shapes and spatial intensity distributions, a suitable combination of morphological operators should be sufficient to achieve the same end.

An obvious way forward is to apply erosion operations to eliminate the shadows around, and dark patches on the grains. These erosion operations should be followed by dilation operations to restore the shapes of the contaminants. While these actions are reasonably effective in eliminating the dark patches around and on the grains, they leave holes in some of the contaminants and thus make it difficult to identify them unambiguously (➔ *Fig. 36.7*).

An alternative approach is to consolidate the contaminants by performing dilation operations followed by erosion operations, but this has the predictable effect of giving significant numbers of false positives in the grain background. Eventually a totally different strategy was adopted – of consolidating the whole image using very large median filters. This procedure is able to virtually eliminate the speckle in the contaminants, and is also successful in not generating many false alarms in the grain regions. However, a certain number of false positives do remain, and it has been found that these can be eliminated by applying additional erosion operations (➔ *Fig. 36.7*).

Once good segmentation of the contaminants has been achieved in this way, with good preservation of their original shapes, recognition becomes relatively straightforward. Perhaps the main disadvantage of the method is the need to apply a large median filter. However, many methods are known for speeding up median operations, and this is not especially problematic. In addition, it could be argued that the methods described here constitute a form of texture analysis, and that there are many alternative ways of tackling this type of problem. However, texture analysis also tends to be computation intensive, and a considerably larger number of operations would be needed to segment the contaminants in this way. Overall, the approach adopted here seems to have the advantages of simplicity, effectiveness and also to be sufficiently rapid for the task. Perhaps the important lesson to be learnt from this case study is that the usual types of morphological operator are not as powerful as is commonly thought, so they have to be augmented by other techniques if optimal solutions are to be found [23].

Finally, an added bonus is that this final algorithm was found to be capable of locating large insects as well as non-insect contaminants. This is reflected in the overall system design (➔ *Fig. 36.6*).

36.7.1 Problems with Closing

The above case study was found to require the use of a median filter coupled with a final erosion operation to eliminate artefacts in the background. An earlier test similarly involved a closing

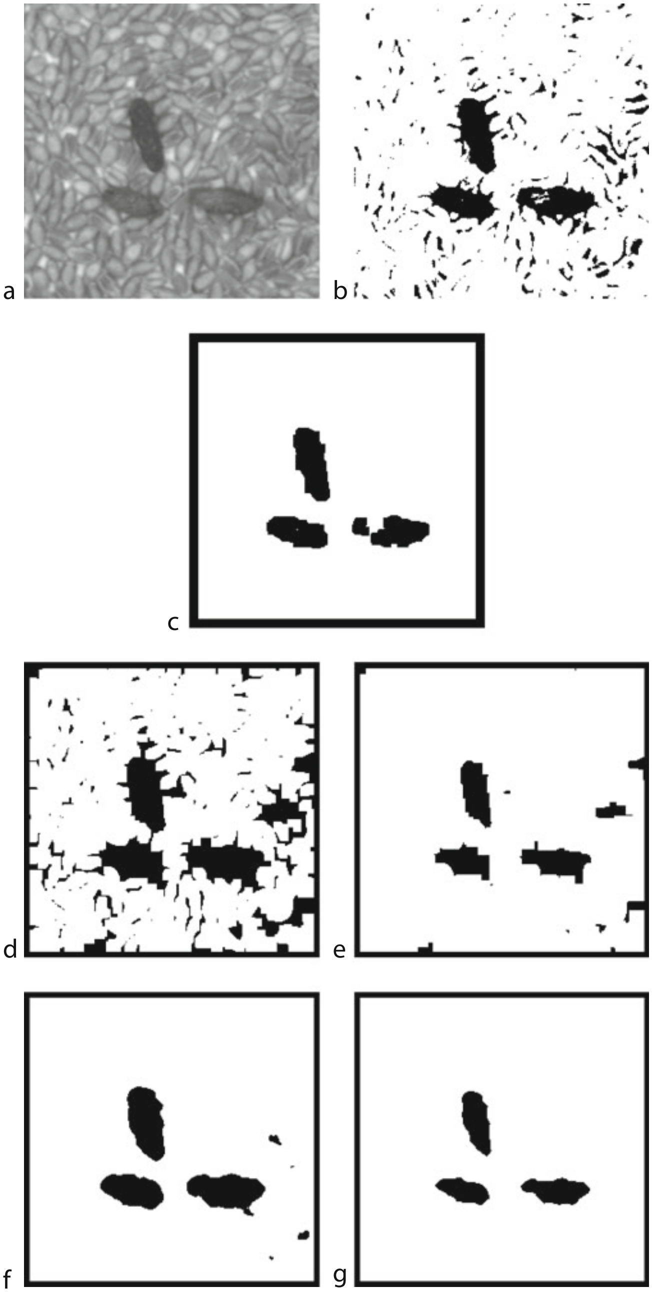


Fig. 36.7

Effects of various operations and filters on a grain image. (a) Grain image containing several contaminants (rodent droppings); (b) thresholded version of (a); (c) result of erosion and dilation on (b); (d) result of dilation and erosion on (b); (e) result of erosion on (d); (f) result of applying 11 × 11 median filter to (b); (g) result of erosion on (f). In all cases, ''erosion'' means three applications of the basic 3 × 3 erosion operator, and similarly for ''dilation.'' © IEE 1998

operation (a dilation followed by an erosion) followed by a final erosion. In other applications, grains or other small particles are often grouped by applying a closing operation to locate the regions where the particles are situated. It is interesting to speculate whether, in the latter type of approach, closing should sometimes be followed by erosion, and also whether the final erosions used in our tests on grain were no more than ad hoc procedures or whether they were vital to achieve the defined goal.

To analyse the situation, we consider two regions containing small particles with occurrence densities ρ_1, ρ_2, where $\rho_1 > \rho_2$ [25]. Clearly, the mean distances d_1, d_2 between particles will depend on the occurrence densities. Dilation will tend to group the particles, but the real aim is to dilate sufficiently to group the particles in region 1 but not those in region 2 (➋ *Fig. 36.8*). Assuming that this is possible, a subsequent erosion should give a good approximation to region 1. However, if any region 2 particles are near the boundary of region 1, they will be enveloped by region 1, which will then become too large, and a subsequent erosion will be needed to restore region 1 to its proper size and shape. The shift δ can be calculated on the basis that it is essentially a one-dimensional effect. However, corrections need to be applied as the particle density in the second dimension in region 2 is not unity and produces a reduction relative to the one-dimensional shift. The final result is:

$$\delta_{2D} = 2ab\rho_2(a + b) \qquad (36.10)$$

where a is the radius of the dilation kernel and b is the width of the particles in region 2. It is important to notice that if $b = 0$, no shift will occur, but for particles of measurable size this is not so. Clearly, if b is comparable to a or if a is much greater than 1, a substantial final erosion may be required. On the other hand, if b is small, it is possible that the two-dimensional shift will be less than 1 pixel. In that case it will not be correctable by a subsequent erosion, though it should be borne in mind that a shift has occurred, and any corrections relating to the size of

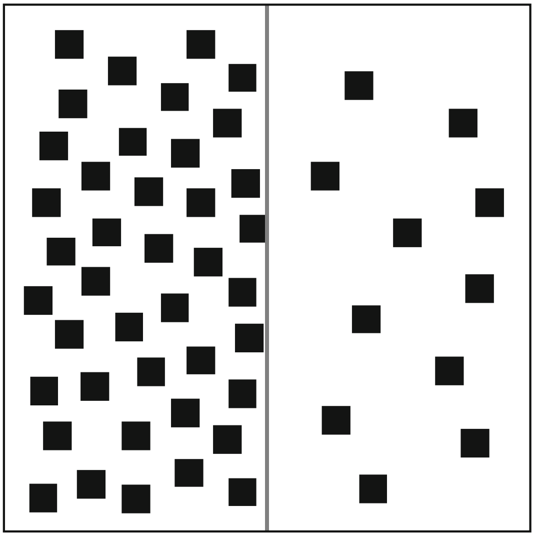


Fig. 36.8

Model of the incidence of particles in two regions. Region 2 has sufficiently low density that the dilated particles will not touch or overlap. © IEE 2000

region 1 can be made during subsequent analysis. While in our work the background artefacts were induced mainly by shadows around and between grains, in other cases impulse noise or light background texture could give similar effects, and care must be taken to select a morphological process which limits any overall shifts in region boundaries. In addition, it should be noticed that the whole process can be modelled and the extent of any necessary final erosion estimated accurately. More particularly, the final erosion is a necessary measure, and is not merely an ad hoc procedure [25].

36.8 Case Study: High Speed Grain Location

It has already been noted that object location can be significantly more computation intensive than the scrutiny phase of inspection, as it involves unconstrained search over the entire image area, and sometimes over several images. This means that careful attention to high speed object location procedures can be extremely useful. Of course, hardware accelerators can be designed to bring about any required improvement in speed, but if inspection systems are to be cost-effective, rapidly operating software is required.

In his earlier work on biscuit inspection (see the Jaffacake case study in ➋ Sect. 36.4 for the general milieu), the author solved this problem by scanning the image along a restricted number of horizontal lines in the image, and taking the mid-points of chords across objects [26]. Suitable averaging in the x and y-directions then permitted the centres of circular objects to be located highly efficiently, with speedup factors as high as 25. While some loss in robustness and accuracy occurred with this approach, performance degraded gracefully and predictably.

In the cereal grain inspection problem, even higher speedup factors were required, and an even faster type of algorithm was required. The only way of obtaining further substantial improvements in speed appeared to be to concentrate on the object regions rather than on their boundaries. Accordingly, a sampling technique was developed which sampled the whole image in a regular square lattice of spacing s pixels. This provided an intrinsic speedup of s^2 coupled with a loss in accuracy by the same factor [27].

Not surprisingly, the technique has significant limitations. In particular, the ideal saving in computation only occurs for square objects with their sides orientated parallel to the image axes. However, if a square object lies at 45° to the image axes it can miss all the sampling points, with the result that the latter must be placed $\sqrt{2}$ times closer together than the ideal to ensure intersecting the objects (➋ *Fig. 36.9*). Similarly, circular objects can miss the sampling points unless their spacing is $\sqrt{2}$ times the circle radius.

To properly understand the sampling process it is necessary to imagine the image space being tiled with shapes identical to those to be located. For simple shapes such as squares, the tiling can be perfect, but for other shapes such as circles, the image space will not be entirely covered without expanding the tiles so that they overlap – thus making the sampling procedure less efficient than the supposed ideal. In fact the situation is even worse when the varying orientations of real objects are taken into account: i.e., a greater number of tiles has to be used to cover the whole image space, and thus an increased number of sampling points is needed; a full explanation appears in [4, 28]. However, although the ideal is not achievable, the results can still be extremely useful, at least for convex objects, and impressive speedup factors can be obtained.

Another limitation of the technique is that accuracy of location is necessarily limited. However, this problem can be overcome once an object's approximate position has been found by sampling. At worst, a region of interest can be set up: within this region a conventional

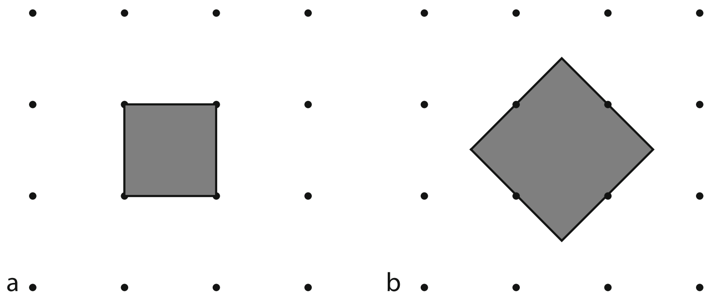


Fig. 36.9
Sampling points for location of *square objects*. (a) shows the spacing of sampling points for locating square objects aligned parallel to the image axes. (b) shows the reduced spacing required for square objects aligned at 45° to the image axes

object location method can be applied, and the object can be located with the usual accuracy. An overall gain in speed results because large tracts of image space are quickly by-passed by the sampling procedure. Interestingly, there is the possibility of finding an accurate location technique whose speed matches that of the sampling procedure. For circles and ellipses the *triple bisection* algorithm operates extremely rapidly: it has some resemblance to the earlier chord bisection strategy, but requires the location and bisection of only three chords of the object. For further details of the algorithm, see ➋ Chap. 18.

The technique has been applied to the location of peppercorns, which are approximately circular, and wheat grains, which are approximately elliptical with an aspect ratio of about 2:1. The method is highly efficient when used to detect circles, as is evidenced by the fact that ideally only one or two sampling points intersect with each object. For elliptical wheat grains, the method is rather less efficient and in the ideal case, up to three sampling points may appear in each object, though the average is close to two (➋ *Fig. 36.10*).

Overall, the technique works well, and saves considerable computation, but only if the objects are well separated. However, alternative procedures can be devised for accurately locating the centres of objects when these are in contact: the basic sampling procedure does not fail in such cases: it is merely more difficult to find accompanying procedures for accurate location whose speed matches that of the basic sampling technique.

36.9 Case Study: Detection of Insect Larvae Growing Within Grain Kernels

The previous three case studies on cereal grain inspection were fairly standard in using normal visible light. The next case study employed near-infrared (NIR) imaging to assess the possibility that insect larvae are growing inside grain kernels. This is important because a batch of grain may appear uninfested, yet after a few weeks insect offspring can emerge from the grain kernels. The strategy adopted to achieve this was aimed not at detecting the heat emitted by the insect larvae, but at observing the change in infrared emission from grains in which starch is becoming depleted, and in which the water content is changing as a result of feeding and

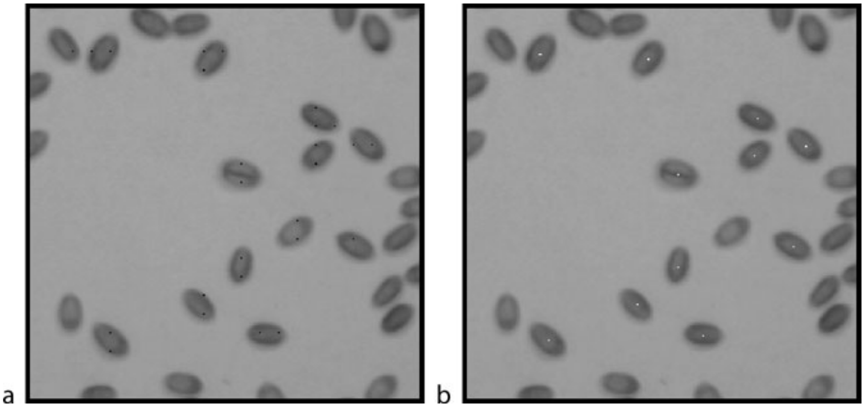


Fig. 36.10
Image showing grain location using the sampling technique. (a) sampling points; (b) final centre locations. © EURASIP 1998

excretion. The relevant spectroscopic lines are around 980 and 1,200 nm [29, 30], though the former offered a better prospect for application of machine vision, as CCD sensors have much greater sensitivity in this region of the spectrum. It should also be remarked that while X-rays offer the possibility of inspecting cereal grains for internal insect infestation, a machine vision system working in the NIR region of the spectrum would offer major advantages in terms of hardware cost, simplicity and safety [31]. In particular, the system would be far lighter and more convenient than an X-ray system, as the heavy shielding needed in the latter case could be dispensed with [4].

Preliminary tests using human inspectors indicated that infested kernels could sometimes be detected on the NIR images (➲ *Fig. 36.11*) by the presence of characteristic bright patches, which are generally absent from the uninfested grains. A variety of machine vision methods was used in attempts to detect the insects via these patches. Notably, initial trials of global and adaptive thresholding led to little success. Attempts were then made to normalise the grain images, to bring them to a standard shape and intensity profile, so that inter-grain variation would be cancelled out and detection would be facilitated. In that case each grain kernel would be compared with a normalised "average" grain kernel. However, all these procedures proved fruitless, ultimately because the inter-grain variation was far too large. Hence a novel approach was adopted – of comparing each grain with a normalised version of itself. This meant (1) filtering the grain images to eliminate any bright patches, and (2) subtracting any grain image from its self-normalised version to reveal the bright patches.

Many morphological erosion techniques were tried, with the aim of achieving the required grain normalisation, but most proved inadequate for this particular application. Perhaps oddly (considering that it had been presumed from the NIR images that only bright patches on the grains needed to be detected), it was found that the best type of filter was one using a centre-seeking statistic, such as the mean or median, which has no potential preference for either bright or dark patches on the grains. The reason this approach worked appeared to be because of the much superior noise suppression capabilities of these types of filter which proved capable of giving a robust well-defined level for comparison, thereby favouring bright patch

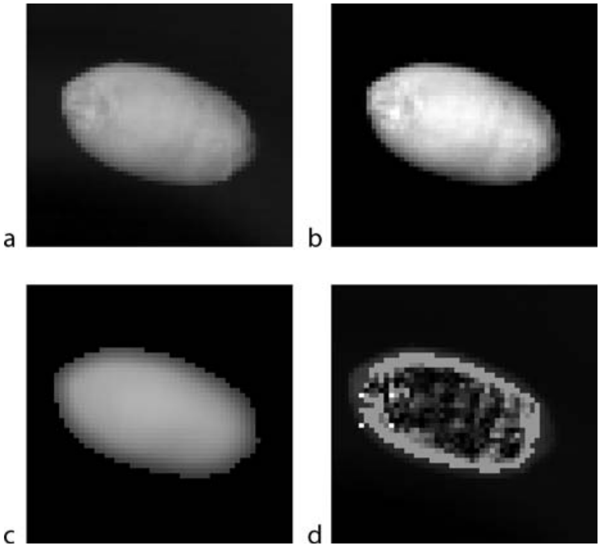


Fig. 36.11
Results for an infested kernel. (a) original NIR image; (b) enhanced version of (a); (c) model derived from (a); (d) difference image after masking: the brightest patch is highlighted with four dots. © IEE 2003

detection. Contrariwise, the reason that a minimum intensity filter did not work well was probably because it produced such an extreme reference that bright or other patches could not be compared with the reference with sufficient reliability. Overall, the median-based reference was distinctly better than the mean type of reference.

Following these strategic observations and decisions, more detailed experiments were undertaken to optimise the effectiveness of the bright patch detector. In particular, it was found that the grain kernels needed to be masked to exclude the outermost regions where there was little useful information for insect detection, and the optimum size and shape of the mask was determined (➲ *Fig. 36.11*). Other optimisations included (1) finding the most suitable size of the bright patch detection window; (2) determining the most appropriate thresholdable indicator of infestation; (3) estimating the optimum size and shape of the initial median filtering grain normalisation procedure; and (4) finding how to correct for the shape distortions produced by the median filtering procedure. The overall algorithm is shown in ➲ *Table 36.2*. ➲ *Figure 36.11* shows the difficulty of seeing anything by eye in the NIR images: note that the "bright patches" have to be enhanced to improve visibility.

In the majority of food inspection applications, it is normal to err on the side of rejection. However, this may not be the best way of dealing with large consignments of grain. In fact, it is better to err on the side of reducing the number of false positives (FPs), so that when the rejected grains are examined, they will be seen to be the ones containing insects. This could help to justify the rejection of any particular consignment (in general, consignments of grain are at least 30 t in weight).

These considerations make it necessary to examine the receiver operating characteristic (ROC). In many works the ROC curve is defined as the graph of true positives (TPs) against

Table 36.2
Stages of the basic insect detection scheme

1. Apply smoothing filter to model grain profile
2. Apply filter correction procedure
3. Apply subtraction procedure to reveal patches
4. Apply mask to locate relevant part of grain
5. Apply patch detection template
6. Apply threshold

Note: these stages ignore the need for setup, training and overall control of the system. In all, nine parameters had to be adjusted for optimal operation and the final threshold set by training on sample data. © IEE 2003

FPs, but here we define it as the graph of FPs against false negatives (FNs); the reason for this is to emphasise that we need to obtain an optimum balance between two undesirable quantities, which is often achieved by making them about equal. In this case, curve-fitting showed that both wings of the characteristic approximated closely to exponential form. As indicated above, the relevant wing is the one with low FP values, and this is shown on a logarithmic scale in *Fig. 36.12*. Notice that the errors increase towards the low FP end, but the fit is actually within one sample error throughout the whole range. Accurate modelling of the ROC characteristic is advantageous as it permits the curve to be scaled to cope with variations in the proportions of infested and uninfested grain kernels. Next, to achieve an optimum balance between FPs and FNs, the solution was taken to be given by the formula:

$$N_{\mathrm{FP}} = nN_{\mathrm{TP}} \tag{36.11}$$

in which we are aiming to minimise the parameter n. Noting that:

$$N_{\mathrm{TP}} + N_{\mathrm{FN}} = I \tag{36.12}$$

where I is the number of infested grains in the total number of grains T, we find:

$$N_{\mathrm{FP}}/n = I - N_{\mathrm{FN}} \tag{36.13}$$

To obtain a valid solution, this straight line has to intersect the ROC curve, and for an optimum solution (with minimum n) it has to be a tangent (*Fig. 36.13*): this follows as a tangent represents an extremum. Finding the tangent uniquely defines n in those cases where a solution exists.

In the experimental tests, the classification accuracy for individual grains was found to be around 80–85% for samples containing ~150 grains and 50% infestation: while this accuracy is not very much higher than the best achieved by human inspectors, the vision approach is naturally far more consistent and also is conducive to absolute measurement. The ROC-based analysis presented above was needed to estimate how to extrapolate the experimental results to situations where the infestation rate is far lower than 50%, e.g., 1% or less. Also, while the above classification accuracy applies to individual grains, analysis (based on the reduced probability of entirely missing infestation when there are a number of infested grains) indicated that batch infestation will be detectable with considerably greater accuracy – around 95% for a 1% infestation level. This is high enough to be practically useful, and also much less costly and more convenient than the obvious alternative of X-ray imaging, though further assessment is needed with larger numbers of images than were available in this study [32, 33].

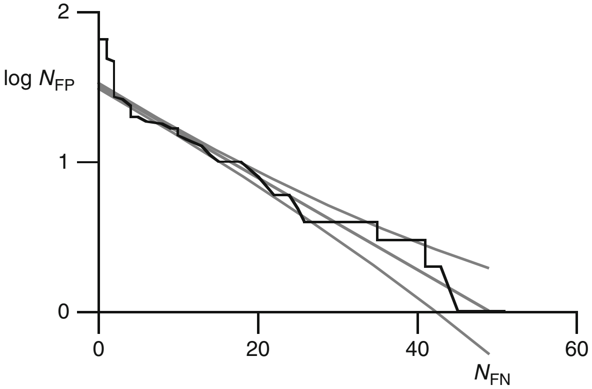


Fig. 36.12

ROC curve for the raw data. The *solid plot* shows the ROC curve for experiments with 74 infested and 74 uninfested grains, plotted on a log–linear scale. The *grey lines* show how curve fitting gives nearly exact agreement for low N_{FN}. The *error bars* show the effect of $\pm$1 sample error at every location. © IEE 2003

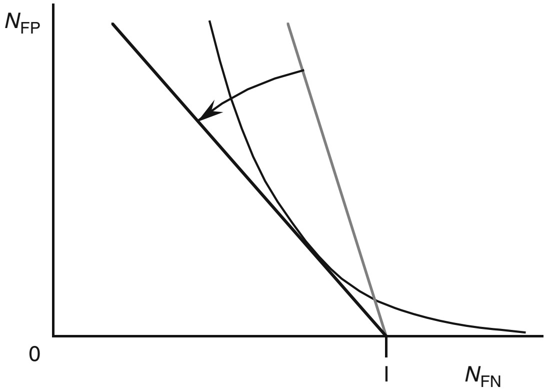


Fig. 36.13

Obtaining an optimal working point on the ROC curve. Valid solutions occur when the straight line (Eq. 36.13) intersects the ROC curve: for an optimum solution, the *straight line* has to be a tangent

36.10 Concluding Remarks

This chapter has described a number of case studies relating to the location of objects in images containing food and cereals. These have ranged from location of Jaffacakes, cream biscuits and cereal grains to location of insects, rodent droppings, moulds and foreign seeds amongst the cereal grains. In all cases the aim has been inspection, though in the first three of these cases the inspected objects had to be scrutinised for defects, whereas in the other cases the main commodity, cereal grains, had to be scrutinised for contaminants.

It is worth remarking that the reason for the emphasis on object location is that this is often, as in all these cases, the most computation-intensive part of the inspection task. Scrutiny itself takes a fraction of the computational effort taken by object location – even when this is carried out by feature detection followed by inference of the presence of the whole object. In fact, the problem of efficient object location for real-time implementation is so serious that in applications related to the Jaffacake inspection task it was necessary to develop an especially fast line-sampling technique. In the later cereal grain inspection problem an even faster technique had to be developed, based this time on a fast point-sampling method. Though sampling was not performed in the other case studies, very careful attention nevertheless had to be paid to achieving high speeds.

In the cases studied, the reasons for inspection ranged from ensuring that the products were the right size and shape – especially with reference to the need to ensure that packing machines are not jammed – and that the products were attractive in appearance. While this criterion obviously benefits the consumer, it also reflects the quantity of product that can be sold and the profit that can be made. However, there are other reasons for checking that products have a normal attractive appearance: specifically, there is then significantly reduced likelihood of foreign objects having alighted on or having been mixed into the product. With raw materials such as cereal grains, the possibility of contaminants is significantly higher. Indeed, cereal grains start their lives on the land, and are thus liable to be contaminated by mud, stones, twigs, insects, droppings, chaff, dust, and foreign grains. In addition, they are susceptible to moulds, sprouting and physical damage. Thus there are distinct health and safety aspects to the inspection task – the more so as certain moulds such as ergot are poisonous to humans. These factors make it all the more essential to inspect consignments of grain for insect infestations and excessive amounts of non-insect contaminants. This would seem to constitute sufficient reason for including these particular case studies in this chapter.

From the academic point of view, these case studies are mere vehicles for the exploration of the problems of inspection, and in the wider field, of machine vision. The similarities between the methods used to search for objects and their features vary very little from topic to topic in machine vision: there are distinct transferable skills and techniques between a good many of them. This point is illustrated by the theoretical sections of the chapter, which help to bridge the gaps between the case studies and to extend the area of application of the ideas they embody far more widely. In particular, it has been shown how speed can be optimised, how problems with matching algorithms (particularly the paradigm maximal clique technique) can arise and be by-passed, how an apparently ad hoc solution to a morphological technique (closing) can be rigorously solved, and how template matching masks can be designed to optimise signal-to-noise ratio and to prevent over-exposure to impulse noise or small amounts of occlusion.

In an ideal world it would be possible to design vision algorithms on paper, starting from careful specifications, but the present reality is that the situation is too complex and such specifications are seldom available, and in many application areas such as food the data images are too variegated and variable to permit us even to approach the ideal. In such circumstances case studies have to be the main route forward, though on occasion this route can be bolstered up and its elements bound more firmly together by theoretical analysis. It is hoped that this chapter will help show the way forward, and firmly demonstrate, with the other chapters in this volume, the enormous value of the case study approach. The reader will also find useful techniques and case studies in [1–3, 34–36].

Acknowledgements

The author is pleased to acknowledge financial support from United Biscuits (UK) Ltd, Unilever UK Central Resources Ltd and the UK Science and Engineering Research Council (EPSRC) for financial support for the baked product work; and from the UK Home-Grown Cereals Authority (HGCA) for financial support for the cereal grain inspection work.

The author is grateful to Dr. J. Chambers and Dr. C. Ridgway of the Central Science Laboratory, MAFF (now Defra), York, UK for useful discussions on the needs of the grain industry, and for providing the original images used to test the cereal grain inspection algorithms. He is also grateful to the HGCA Cereals R&D Committee for guidance on the relative importance of different types of contaminant. Finally, he would like to thank former RHUL colleagues Mark Bateman and David Mason for vital assistance with the implementation of key cereal inspection algorithms.

➲ *Figure 36.1* is reproduced from [10] with permission from the M^2VIP'95 Organising Committee. ➲ *Table 36.1* and ➲ *Fig. 36.6* are reproduced from [20]: permission is granted by the Council of the Institution of Mechanical Engineers. ➲ *Figures 36.3* and ➲ *36.10* are reproduced from [16, 28] with permission from EURASIP. ➲ *Table 36.2* and ➲ *Figs. 36.5*, ➲ *36.7*, ➲ *36.8*, ➲ *36.11*, ➲ *36.12* and some of the text are reproduced from [17, 23, 25, 33, 35] with permission from the IEE (now the IET).

References

1. Davies ER (2005) Machine vision: theory, algorithms, practicalities, 3rd edn. Morgan Kaufmann, San Francisco
2. Ballard DH, Brown CM (1982) Computer vision. Prentice-Hall, Englewood Cliffs
3. Batchelor BG, Hill DA, Hodgson DC (eds) (1985) Automated visual inspection. IFS, Bedford/North Holland
4. Davies ER (2000) Image processing for the food industry. World Scientific, Singapore, pp xx + 289
5. Davies ER (1992) Optimal template masks for detecting signals with varying background level. Signal Process 29(2):183–189
6. Davies ER (1997) Vectorial strategy for designing line segment detectors with high orientation accuracy. Electron Lett 33(21):1775–1777
7. Davies ER (1999) Effect of foreground and background occlusion on feature matching for target location. Electron Lett 35(11):887–889
8. Davies ER (1999) Designing optimal image feature detection masks: the equal area rule. Electron Lett 35(6):463–465
9. Davies ER, Atiquzzaman M (1998) Special Issue on Projection-based transforms. Image Vision Comput 16(9–10)
10. Davies ER (1995) Machine vision in manufacturing – what are the real problems? Invited paper in Proceedings of the 2nd International Conference on Mechatronics and Machine Vision in Practice, Hong Kong (12–14 September), pp 15–24
11. Davies ER (1992) Locating objects from their point features using an optimised Hough-like accumulation technique. Pattern Recogn Lett 13(2):113–121
12. Ridgway C, Chambers J (1996) Detection of external and internal insect infestation in wheat by near-infrared reflectance spectroscopy. J Sci Food Agric 71:251–264
13. Zayas IY, Steele JL (1990) Image analysis applications for grain science. Optics in Agric, SPIE 1379:151–161
14. Keefe PD (1992) A dedicated wheat grain image analyser. Plant Var Seeds 5:27–33
15. Liao K, Paulsen MR, Reid JF (1994) Real-time detection of colour and surface defects of maize kernels using machine vision. J Agric Eng Res 59:263–271
16. Davies ER, Mason DR, Bateman M, Chambers J, Ridgway C (1998) Linear feature detectors and their application to cereal inspection. In: Proceedings of the EUSIPCO'98, Rhodes, Greece, 8–11 September, pp 2561–2564
17. Davies ER, Bateman M, Mason DR, Chambers J, Ridgway C (1999) Detecting insects and other dark line features using isotropic masks. In: Proceedings of the 7th IEE International Conference on Image Processing and its Applications, Manchester (13–15 July), IEE Conference Publication No 465, pp 225–229

18. Zayas IY, Flinn PW (1998) Detection of insects in bulk wheat samples with machine vision. T Am Soc Agr Eng 41(3):883–888
19. Chambers J (1997) Revealing invertebrate pests in food. In: Crop protection and food quality: meeting customer needs. In: Proceedings of the BCPC/ANPP Conference, Canterbury, Kent (16–19 September), pp 363–370
20. Davies ER, Chambers J, Ridgway C (2003) Design of a real-time grain inspection system with high sensitivity for insect detection. In: Parkin RM, Al-Habaibeh A, Jackson MR (eds) Proceedings of the International Conference on Mechatronics (ICOM 2003), Loughborough, 18–20 June, Professional Engineering Publishing, pp 377–382
21. Davies ER, Bateman M, Mason DR, Chambers J, Ridgway C (2003) Design of efficient line segment detectors for cereal grain inspection. Pattern Recogn Lett 24(1–3):421–436
22. Jang J-H, Hong K-S (1998) Detection of curvilinear structures using the Euclidean distance transform. In: Proceedings of the IAPR Workshop on Machine Vision Applications (MVA'98), Chiba, Japan, pp 102–105
23. Davies ER, Bateman M, Chambers J, Ridgway C (1998) Hybrid non-linear filters for locating speckled contaminants in grain. IEE Digest No. 1998/284, Colloquium on Non-Linear Signal and Image Processing, IEE (22 May), pp 12/1–5
24. Haralick RM, Sternberg SR, Zhuang X (1987) Image analysis using mathematical morphology. IEEE Trans Pattern Anal Mach Intell 9(4):532–550
25. Davies ER (2000) Resolution of problem with use of closing for texture segmentation. Electron Lett 36(20):1694–1696
26. Davies ER (1987) A high speed algorithm for circular object location. Pattern Recogn Lett 6(5):323–333
27. Davies ER (1997) Lower bound on the processing required to locate objects in digital images. Electron Lett 33(21):1773–1774
28. Davies ER (1998) Rapid location of convex objects in digital images. In: Proceedings of the EUSIPCO'98, Rhodes, Greece, 8–11 September, pp 589–592
29. Ridgway C, Chambers J (1998) Detection of insects inside wheat kernels by NIR imaging. J NIR Spectrosc 6:115–119
30. Ridgway C, Chambers J, Cowe IA (1999) Detection of grain weevils inside single wheat kernels by a very near infrared two-wavelength model. J NIR Spectrosc 7:213–221
31. Sugiyama J (1999) Visualization of sugar content in the flesh of a melon by near-infrared imaging. J Agric Food Chem 47:2715–2718
32. Chambers J, Ridgway C, Davies ER (2001) Design of an integrated machine vision system capable of detecting hidden infestation in wheat grains. HGCA Project Report No. 262, HGCA, London (October)
33. Davies ER, Ridgway C, Chambers J (2003) NIR detection of grain weevils inside wheat kernels. In: Proceedings of the IEE International Conference on Visual Information Engineering, VIE 2003, Surrey (7–9 July), IEE Conference Publication 495, pp 173–176
34. Chan JP, Batchelor BG (1993) Machine vision for the food industry. Chap. 4 in Pinder AC, Godfrey G (1993), pp 58–101
35. Naim MM, Chan JP, Huneiti AM (1994) Quantifying the learning curve of a vision inspection system. In: Proceedings of the IEE 4th International Conference on Advanced Factory Automation, New York (November), IEE Conference Publication No. 398, pp 509–516
36. Graves M, Smith A, Batchelor BG (1998) Approaches to foreign body detection in foods. Trends Food Sci Technol 9(1):21–27

37 Automatic Produce Grading System

Donald Bailey
Massey University, Palmerston North, New Zealand

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_37,

Abstract: The development of a machine vision system for automated high-speed produce grading is described. Image processing techniques were used to obtain an estimate of the volume of each item, which was then related to the weight through a closed-loop calibration. Accurate weight estimation led to more accurate and better control over the spread of package weights. This reduced the average package weight by approximately 20%, with a standard deviation of 2.5 g for a nominal 100 g package. Improved processing efficiencies doubled the throughput and significantly increased the profitability of the packinghouse.

37.1 Introduction

It is usual to sell produce (fruit and vegetables) by weight. When sold in a supermarket, the produce often is prepacked into fixed weight packages. This introduces three criteria on the packing process. First, all of the items within a package should be approximately the same size. A uniformity of size is not only aesthetically appealing, but when the produce is cooked it ensures that all of the items will have the same cooking time. A second, legal, requirement is that there should not be less than the stated weight within the package. There may be an allowance for a small percentage of packages to be slightly underweight, depending on the jurisdiction. The third criterion is that the grower or packinghouse will not want to significantly exceed the stated weight because this is effectively giving away produce.

When each item is a small fraction of the total package weight, and either the items or the packages can be weighed while packing, these criteria can be achieved readily, without giving away much produce. With small package sizes, where each package contains only a few items, the variability in package weight can be significant. The need for a uniform item size results in the weights being quantised. Unfortunately, most produce does not grow in discrete weights or sizes, but has a continuous distribution. The packing process therefore consists of weighing each item to assign it to a size grade, and then distributing the items within a size grade over several packages to achieve the weight criteria.

A packinghouse must maintain a high throughput to be profitable. Grading is usually a bottleneck in this process, so it is desirable to minimize the time spent on each individual item. Commonly, each item is weighed by passing it over a load cell. The load cell, which is effectively a very stiff spring, deflects slightly because of the weight of the item. A strain gauge converts this deflection into an electrical signal, which is then amplified and digitised to give the weight. The vibration and bouncing of the produce as it deflects the load cell combined with the intrinsic settling time of the load cell provide an upper bound to the throughput of the complete grading system.

Because a load cell is just a spring, when the load changes it will tend to oscillate. The time that it takes these oscillations to decay is the settling time of the load cell. A typical load cell has a settling time of 2–3 s, which is excessive in a high-speed grading application. The settling time can be reduced by damping the oscillations, either passively or actively. Commercially available damped load cells can reduce the settling time to about 100–200 ms, although such high-speed load cells are very expensive. When the target throughput exceeds ten items per second, physically weighing the individual items is very expensive and impractical.

To overcome these problems, size-based grading is often used as a substitute for weighing each item. An accurate estimate of the weight from the size requires the density to be constant, which is often the case. Any variations in the density will increase the uncertainty in the weight estimation, and the average package weight must increase to ensure that the minimum weight criterion is met.

For items that are approximately spherical, mechanical screens provide a fast and effective method of size grading. Each screen usually consists of a mesh with certain hole size. When the mesh is vibrated, any items that are smaller than the holes will fall through the mesh, while those larger than the mesh size remain on top. By having a series of meshes, the items can be sorted into several size categories. This approach is generally unsuitable for produce grading because it relies on mechanical jostling and this may bruise or damage some items. It is also unsuitable for items that are long and thin, because the size of each item then depends significantly on its orientation.

Machine vision can provide a viable alternative to mechanical weighing systems. One or more images are captured of each item (or part of each item). From each image, the object is segmented from the background to derive information on the object's shape. An image only provides a two-dimensional projection of the three-dimensional object so it is necessary to make assumptions about the shape of the object in the third dimension. These assumptions are usually embodied in a three-dimensional model, with the data derived from each image used to adjust the parameters of this model to match the view. The volume, hence the weight, can then be estimated from the model.

37.2 Application Details

We were approached by a packinghouse that was grading asparagus for export. They wanted to upgrade their existing grading line with a view to automating some of the packing process. Asparagus is quite perishable once harvested, with a storage life of less than 3 weeks [14], so it is necessary to have the produce on the shelves for sale as fast as possible. The steps within this process are as follows.

The asparagus is harvested manually in the field, often using migrant labour. They first select the spears that are sufficiently long and then cut these with a sharp knife just below ground level. Those that are too short are left to continue growing to be harvested later. Removing the spears stimulates the tuber to send out more shoots. This cannot continue indefinitely as the tuber only has limited energy reserves, so the harvesting season is usually quite short, typically 2–3 months. The harvested asparagus is placed in bins, which are regularly collected and taken to the packinghouse for packing. In the packinghouse, the bins are emptied into a water bath to wash any soil and sand from the harvested shoots. After the water bath, the produce is loaded onto a conveyor for grading. The conveyor consists of a chain of V-shaped cups which singulate the spears as shown in ➋ *Fig. 37.1*. The conveyor runs at a constant speed, which can be adjusted from 0 to 30 cups per second. A belt aligns the tips of the spears before a blade trims them to a uniform length, removing the field cut. The spear is then graded, with the result used to direct the spear down a particular chute. When the appropriate quantity has been collected by a chute it is released as a bundle to be packaged. Each bundle is banded and taped before being packed into small wooden crates (➋ *Fig. 37.1*). Wooden crates provide rigidity, protecting the produce during shipping. The produce is then heat-treated to prolong the shelf life by passing the crates through a tepid water bath for a predefined time, followed by rapid cooling using chilled air. Finally, the produce is exported to its destination by air-freight. The total time from harvesting to arriving in the country of destination is typically 24–36 h.

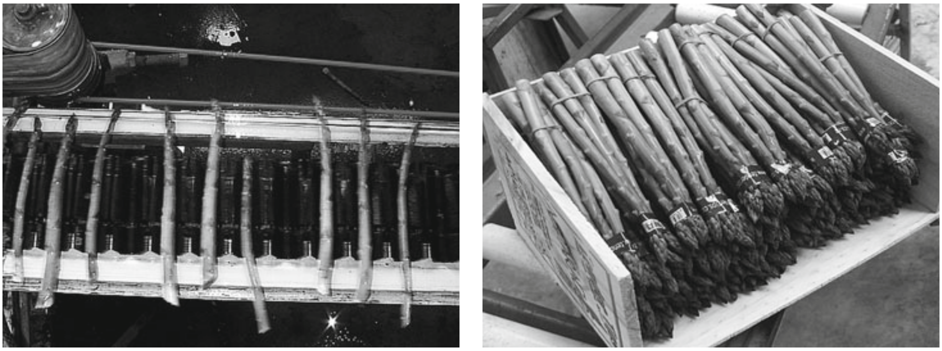


Fig. 37.1
***Left*: produce singulated onto the conveyor. *Right*: packaged produce in a wooden crate**

The throughput of the packinghouse is limited by the grading speed. The individual spears vary in weight between 5 and 50 g. These are to be packaged into 100 g bundles. Each bundle should contain between three and seven similarly sized items, with the very large and very small items rejected. Such a wide weight range per item, and a small item count per bundle, requires that each spear be weighed so that it may be allocated to an appropriate bundle.

37.2.1 Previous Grading System

When the grading line was first set up, size, resolution, and speed constraints meant that it was impractical to measure the whole spear. Each spear is approximately cylindrical, and the length is known (each item has been trimmed to the same length prior to grading) so a single diameter measurement was made at the base of each spear and used as a proxy for the weight. The camera was free-running with the aperture held open. As each cup passed the camera, it triggered a solid-state flash unit to effectively freeze the motion of the conveyer. The next frame after the flash was triggered contained the image of the cup and spear. Since each spear extended past the end of the cup, the produce was measured at the base where it appeared against a black background, enabling a simple global threshold to be used for segmentation. The diameter was estimated by measuring the area of the object within the field of view, which corresponded to a fixed length along the spear. Calibration then associated a range of diameters to each produce weight class.

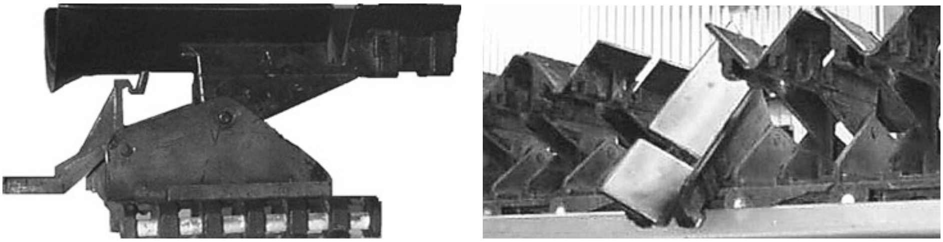
The system was calibrated in an ad hoc manner, through experience. Small adjustments were made to the diameter thresholds until there were fewer than the allowed number of underweight bundles. Unfortunately, the thresholds could not just be set once and left. Subtle changes in the shape and density of the produce during the season meant that the thresholds had to be adjusted slightly from day to day.

Unfortunately, there were several problems with the previous system. The biggest of these was the very wide spread of weight in each package. The primary cause of this was the fact that the whole spear was not imaged, and the single diameter measurement was not a very accurate predictor of the weight. The aspect ratio of a spear is such that to obtain an image across its complete length, the resolution in measuring the diameter is significantly reduced. This, combined with the greater processing power required to segment the spear from the conveyor system, led to unavoidable compromises being made in the original system design. The weight was not actually being estimated but instead the produce was separated into classes based on spear diameter at the butt. This resulted in a very wide spread in package weight. To avoid underselling, the packages had to be significantly overweight on average. This combination of factors meant that the packages assembled by the previous system were 20–30% overweight on average. Consequently, the packinghouse was effectively giving away 20–30% of the produce.

A second problem with the previous system was the limited throughput. The grading system was not only inaccurate, but it was also slow with a maximum processing speed of eight cups per second. The constraint here was the limited processing power available at the time that the system was developed.

A third problem was an actuation issue. The cups on the conveyor are mounted with an offset hinge (this is shown in ❯ *Fig. 37.2*). They are held in their normal position by a small latch with a protruding tag containing a soft iron armature. When a cup containing a spear reaches the desired chute, a solenoid is activated which pulls on the tag, releasing the latch. The cup then tips, dropping its contents down the chute. The solenoid must be activated precisely as the cup is travelling past. Since the conveyor speed can be adjusted, using a fixed time delay to account for the distance between the camera and the solenoid is not practical. The system counted cups as they passed the camera, and used the known constant spacing between the cups to determine the location of each cup. The distance (in cups) between the grading point to the release point for each chute is known enabling the solenoid to be activated at the correct time.

This worked fine, most of the time. Unfortunately, the cup spacing changed slightly as a result of both belt wear and temperature. The grading line was over 10 m long, and with a cup pitch of 5 cm a 0.5% change in belt length was sufficient to trigger the wrong cup at the end of the line. While this is a large change, the belt did stretch by this amount over the course of one or two seasons. A significantly smaller change was sufficient to give unreliable triggering. Consequently, as the packinghouse, and conveyor belt, warmed up during the day, the triggering mechanism began to fail. This necessitated stopping the grading line while the solenoid positions were manually adjusted so they would trigger reliably again. Such adjustment was needed once, and sometimes twice, on most days. While the adjustment only took about 10 min, grading was halted during this time.



■ Fig. 37.2
Cup showing release mechanism, and the cup in the released position (Reproduced from [10], used with permission)

37.2.2 Replacement Grading System

The goal of the project was to significantly improve the profitability of the packinghouse by reducing the average weight of each package and increasing the grading throughput. A secondary goal was to increase productivity by increasing the throughput, and reduce the amount of manual labour required as part of the grading process. This can result in a significant return on the investment in developing the grading system [5].

The legal requirement of maintaining the minimum weight implied that it was necessary to obtain a better estimate of the weight of each spear. This would be accomplished through more sophisticated image analysis techniques. A more accurate weight estimate would then allow more effective grading and package allocation strategies to be devised. The target was to reduce the spread in package weight such that the average package weight could be reduced to 105 g.

A target throughput of up to 15 spears per second was set. This speed makes standard weighing techniques prohibitively expensive, although a new measurement method based on the impact on load cells was investigated [7,8]. This technique used digital signal processing to cancel the resonant response of the load cell, and potentially enabled a standard load cell to make up to 20 measurements per second. However, the mechanical handling difficulties of bouncing spears off a load cell at this rate without collision made this technique impractical in this application.

This left image analysis methods to estimate the weight indirectly from the volume. The target throughput imposes a maximum processing time of 65 ms for each spear. This very tight time constraint limited the type of image processing operations that could be applied to the images. The initial design of the replacement system was documented in [4].

For practical reasons, the grading line had to maintain the same structure as the existing mechanical system. The produce was still loaded onto V-shaped cups and trimmed using the same mechanisms. The previous vision system was replaced with a new design based on imaging the complete spear. As the cup release mechanism was unchanged, it was necessary to develop a new approach to overcome the problems of belt stretching. Changes were made to the chute design to reduce the time between completing a bundle, and starting a new bundle. This was a first step within a larger project of automating the packaging of the produce. The redesign of the chutes and subsequent packaging technologies are beyond the scope of this chapter.

37.3 Modelling

Image processing techniques cannot directly measure the weight, or even volume of an object. Therefore, it is necessary to develop a model that relates data that can be measured to the weight. Some prior work has been carried out on 3D modelling of asparagus [11], although the focus there was estimating the surface area of the spears for calculating heat and mass transfer during post harvest processing. In contrast with this, the work in this chapter requires estimating the weight of each spear.

37.3.1 Verification of Constant Density

The closest that image processing can come to measuring weight is to estimate the volume of an object. This requires that the density of the produce is approximately uniform throughout the volume, and that the density varies little between items. Any variations at this stage will limit the accuracy with which the weight may be estimated, and consequently whether or not the target package rate could be practically achieved. Therefore, the first step before starting the project was to check that the project was not destined to failure.

A sample of about 100 spears was collected from over the full range of sizes to be graded. Each spear was weighed using precision scales. The volume was measured by immersing each spear in a measuring cylinder containing water, and the volume determined from the difference in water level before and after the spear was added. Although the density did vary slightly with spear size, it was possible to estimate the weight from the volume alone to within 1% accuracy.

This verified that estimating the volume of each item provided a viable alternative to actually weighing the items.

37.3.2 Volume Estimation

As each spear is approximately cylindrical, an obvious model is a generalised cylinder. An image of the spear provides a projection of the cylinder from which the volume is to be estimated. The volume of a generalised cylinder can be calculated by integrating the cross-sectional area as a function of the length

$$V = \int A(x)\mathrm{d}x \tag{37.1}$$

However, the cross-sectional area cannot be obtained from a projection. Referring to ➋ *Fig. 37.3*, the diameter, D, can be measured as a function of length, and assuming a circular cross-section the volume from ➋ Eq. 37.1 becomes

$$V = \frac{\pi}{4}\int D^2(x)\mathrm{d}x \tag{37.2}$$

When processing images, this calculation may be simplified by aligning the horizontal axis of the camera with the spear and ensuring that the camera line of sight is perpendicular to the spear. The diameter can then be measured by counting pixels within each column of the image within the spear, D_x. The discrete version of ➋ Eq. 37.2 is then

$$V \propto \sum_x D_x^2 \tag{37.3}$$

where the constant of proportionality depends on the size of each pixel and is determined through fitting the calculated result to the known volume. Estimating the volume from a single projection was only accurate to within 10%. Therefore, while such a model would enable more

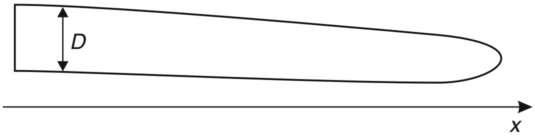


Fig. 37.3

The projection of a spear (Reproduced from [3], used with permission). Measuring the diameter as a function of length allows the volume to be estimated

accurate weight estimation than previous system, it is not sufficiently accurate to meet the target performance. Clearly, the assumption of circular cross-section is limiting the accuracy of this initial model.

To estimate the volume more accurately, it is necessary to take into consideration the eccentricity of the cross-section of each spear. A separate image was therefore captured of the end of each spear at the same resolution as the original image. From this the cross-section area, A_0, and diameter, D_0, at the base were measured. It was then assumed that this cross-section scaled in proportion to the visible diameter along the length of the spear:

$$V \propto \sum_x A_0 \frac{D_x^2}{D_0^2} = \frac{A_0}{D_0^2} \sum_x D_x^2 \tag{37.4}$$

This model improved the accuracy of the volume estimation to about 6%. The accuracy of this approach could not be increased any further because of the limited resolution in measuring both the spear cross-section and the diameter at each point along the spear. While an improvement, this was still outside the target accuracy requirements of the project.

The next approach was to measure projections from two perpendicular views. If an elliptical cross-section is assumed, this enables the cross-sectional area to be estimated from the projected diameters:

$$A = \frac{\pi}{4} D_{\text{minor}} D_{\text{major}} \tag{37.5}$$

In general, the perpendicular views will not be aligned with the major and minor ellipse axes. Therefore, the two measurements, D_1 and D_2, while perpendicular, will be randomly aligned with the ellipse axes. This situation is illustrated in ➋ *Fig. 37.4*. Substituting ➋ Eq. 37.5 into ➋ Eq. 37.1 gives

$$V \propto \sum_x D_1 D_2 \tag{37.6}$$

which results in significant error when the cross-section is more elliptical and is oriented close to 45°. However, it is known from moment invariants that

$$m_{02}^2 + m_{20}^2 = \text{constant} \tag{37.7}$$

where m_{02} and m_{20} are the second central moments in two perpendicular axes. For an elliptical shape, the second moment along an axis is proportional to the diameter in that direction. Therefore, inspired from these relationships the volume can be modelled as

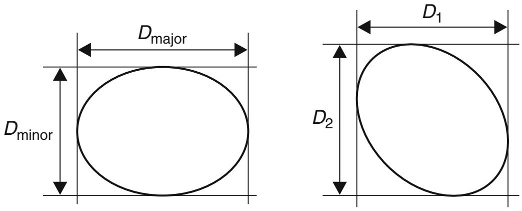


Fig. 37.4
The perpendicular projections of an elliptical cross-section (Reproduced from [3], used with permission). *Left*: ideal projections of the major and minor axes. *Right*: a more realistic set of measurements

$$V \propto \sum_x \left(D_1^2 + D_2^2\right) = \sum_x D_1^2 + \sum_x D_2^2 \tag{37.8}$$

Comparing ➋ Eqs. 37.8 and ➋ 37.3, it is clear that this is just the average of the volumes estimated from the two individual views. Initial measurements indicated that this model could estimate the volume of an individual spear to within about 3% accuracy. This was considered to be adequate for achieving the targeted goals.

37.4 Image Capture Setup

37.4.1 Optical Arrangement

The next stage was to obtain two perpendicular views of each spear as it moved along the conveyor at up to 15 cups per second. The first alternative was to have two cameras, each mounted at 45° to the conveyor. However, as each spear is significantly longer than its diameter, most of the image would be empty. There is sufficient space within a single image to capture multiple views of each spear. The two views can be obtained using a pair of mirrors [4], as shown in the left panel of➋ *Fig. 37.5*. Although two views can actually be obtained with a single mirror with one direct and one reflected view, the two-mirror arrangement has two advantages. First, the optical system is symmetric so that the two perpendicular views have the same path length, hence the same scaling. It also provides a third, direct, view that can be used for quality grading. The third view cannot be used to improve the volume estimation. Incorporating the central view destroys the moment invariant of ➋ Eq. 37.7, increasing the error.

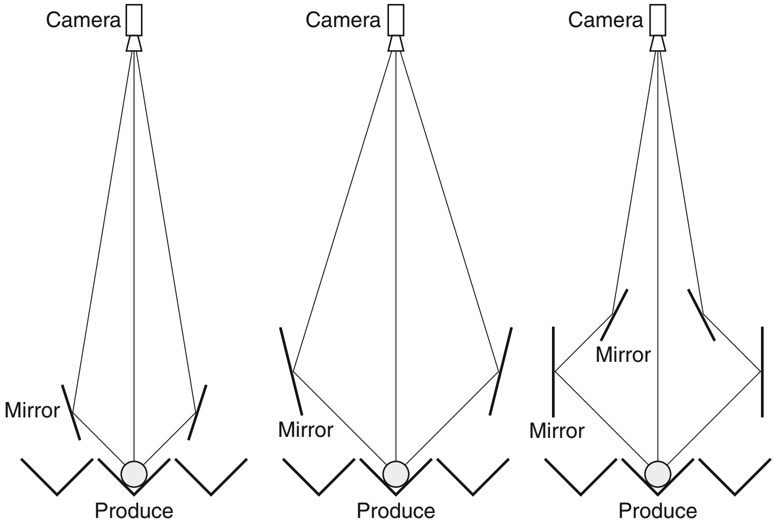


Fig. 37.5
Optical arrangements for capturing two perpendicular views using mirrors (not to scale). *Left*: initial arrangement (Reproduced from [3,4], used with permission). *Centre*: raising the mirrors. *Right*: final arrangement

To maintain adequate focus with the difference in path length between the central and side views, the camera has to be positioned as far as possible from the conveyor. This must be balanced by practical considerations, which limit the size of the grading system. A compromise was reached by placing the camera about 0.8 m above the conveyor and using a slightly telephoto lens.

To test the arrangement in the laboratory, two polished aluminium mirrors were mounted on a wooden substrate. While the aluminium mirrors were not imaging quality, they allowed several mirror configurations to be investigated, and to determine what adjustments would be needed when setting up the grading line.

The mirror system used for the actual grading line consisted of front-silvered optical glass mirrors. Normal mirrors are back-silvered, with the reflective coating behind the glass. While this protects the delicate, silvered, surface, it also results in a double image with reflections from both the front and back surfaces. This can be avoided by using front-silvered mirrors. The silvered surface is protected by a scratch resistant anti-reflective coating. The mirrors were mounted within an aluminium frame to facilitate assembly and removal for maintenance.

A packinghouse is quite a dusty environment. Any dust on the mirrors would reduce the quality of the image. To minimise dust on the mirrors, the complete imaging system was mounted within an enclosed box (apart from where the conveyor ran through). Filtered air was blown into the imaging box to create a downwards airflow to reduce the amount of dust rising from the conveyor. The box also blocked ambient light, giving more control over the lighting.

The arrangement worked fine in the laboratory, and also when first set up for testing in the factory. However, when the grading line was operated at full speed an occasional spear that was not quite sitting flat in the cup would catch on the bottom edge of the mirror and flick out. This would then catch other spears, and foul the system. When this occurred, the grading line had to be shut down and the conveyor cleared. Unfortunately, this happened sufficiently frequently to make the system impractical.

The problem is the relatively small clearance between the bottom of the mirror and the top of the cup. To achieve better clearance, the mirror needs to be lifted, and moved further from the centre-line, as shown in the central panel of ➲ *Fig. 37.5*. To fit all three views within a single image then required a significantly wider angle lens. Unfortunately, this also reduced the resolution by about a factor of two, with a consequent increase in the measurement error.

To retain the original field of view (and accuracy) but raise the mirrors, a second set of mirrors was introduced as shown in the right panel of ➲ *Fig. 37.5*. The disadvantage of this arrangement is that the path length of the side views is increased. This slightly reduced the resolution of the side views. It also required a longer depth of focus to obtain well focussed images in all three views. The depth of focus was increased by reducing the aperture, at the expense of requiring stronger illumination to maintain an adequate exposure.

Although the extra set of mirrors makes the optical arrangement more complex, it actually simplified setting the system up. The angle of all four mirrors was made adjustable, allowing control over both the angle of view and the position of the spear within the captured image.

This final arrangement doubled the clearance between the mirrors and the cups and overcame the initial problems with produce being caught on the bottom. Once the camera was focussed, the focus ring was locked with a grub screw to prevent unwanted changes as a result of vibration.

37.4.2 Lighting

The lighting must be such that the spear can be segmented from both the cup and the background. The cups are made of black plastic, which aids segmentation of the lighter spears. The ends of the spear extend past the ends of the cup so the background clutter was removed by mounting a matt black shroud between the conveyor and the cups.

With the cups moving continuously past the camera, it was important to freeze the motion in the images. With a throughput of 15 cups per second, the motion within the image is approximately 2,250 pixels per second. Therefore, to keep the motion blur to less than one pixel, an exposure time of 1/2,500 s or shorter was necessary. High intensity flash illumination was considered, but ruled out because flash frequencies of 10–15 Hz are known to trigger epileptic attacks in susceptible people. While the camera box is enclosed, some light would inevitably escape through the port where the conveyor runs through.

The exposure time was well within the range of electronic shutter speeds of solid-state cameras. Therefore, continuous illumination was used combined with electronic shuttering to freeze the motion. The short exposure time combined with the narrow aperture needed to give a broad depth of focus means that the camera receives very little light. This necessitated high intensity illumination, which was provided by a bank of quartz halogen lamps. Flicker was avoided by running the lamps from an independent DC power supply.

The initial lighting arrangement had the lights mounted adjacent to the camera. This gave good even illumination of the direct view, and provided additional light to the side views through reflection off the mirrors. This arrangement, however, was much less effective after changing to the optical system with four mirrors. Considerable time was spent, both in the laboratory and in the factory to find a new arrangement that gave even illumination for all three views. To facilitate this experimentation, the lamps were mounted on magnetic clamps that enabled both the position and angle of the lamps to be adjusted easily. In the final system, of course, the lamp brackets were firmly fixed to the frame.

The final lighting arrangement is illustrated schematically in ➋ *Fig. 37.6*. A pair of lamps was mounted above the cup, illuminating the spear from the ends. These are positioned at an angle to maximise the light between the sides of the spear and the cup. It was found that if these illuminated the near end of the spear, it was hard to obtain even illumination, and the centre of the spear was either significantly darker or lighter than the ends. This made segmentation from the cup more difficult. The presence of the imaging mirrors also restricted where the lamps could be positioned. The most even illumination was obtained by directing the light from narrow beam lamps to the far end of the spear, as shown in the left panel of ➋ *Fig. 37.6*. A polished aluminium reflector was placed at each end to direct light back along the spear. Sometimes the ends of the spear drooped slightly where they extended past the ends of the cup. An additional pair of wider beam lamps was mounted at each end of the cup to improve the illumination in this case. In total, eight 40 W lamps were used to provide the high intensity illumination needed for the short exposure. In the right panel of ➋ *Fig. 37.6*, the four lamps at the base of the spear can be seen. The central two are illuminating the length of the spear and the outer two providing additional illumination for the base. Only the two wide beam lamps are visible at the tip of the spear; the other two are just out of the field of view.

Once the lighting had been set up, the aperture ring on the camera lens was also locked with a grub screw. This precaution also prevents unwanted movement as a result of vibration.

To reduce any vibrations from the conveyor from affecting the images, both the camera and mirror arrangement were mounted on a separate frame from the imaging box, which was

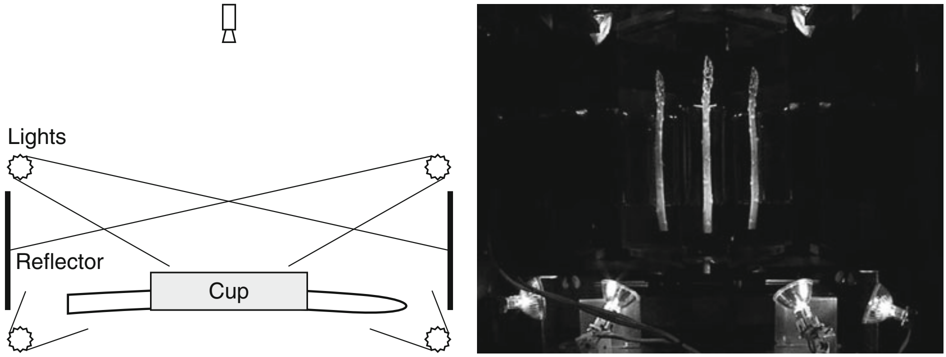


Fig. 37.6
Lighting arrangement to give even illumination. *Left*: relative light positions (Reproduced from [3], used with permission). *Right*: view from the camera

mounted directly on the conveyor frame. It is not known if this precaution was strictly necessary, but as a result, the camera and optical system experience very little vibration.

37.5 Image Processing

A sensor mounted beside the conveyor detected each cup as it passed. This was used to trigger the capture of an image and begin processing. Processing speed is critical because at 15 cups per second there is only approximately 60 ms to process each image. Several experiments were performed using both colour and infra-red imaging, but these gave very little additional information to that present in a monochrome image. A Sony XC-55 camera was chosen because it allowed the necessary asynchronous triggering to match the speed of the conveyor. It also provided a progressive scan output, enabling the full resolution image to be captured and output for each frame.

A typical image captured by the system is shown in ➜ *Fig. 37.7*. From this image, several difficulties are apparent from an image processing perspective. The two side views are slightly darker than the central view because of the slight attenuation from each reflection by the mirror. The limited illumination between the spear and the cup reduces the contrast making segmentation more difficult. The roughness of the surface of the spear results in significant variations in intensity, especially at the tip of the spear, and there are specular reflections from the bracks (the small triangular leaf-like growths) along the length of the spear. There are also specular reflections from end of the cup. This is because the tip end of the cup has a rounded profile to prevent the spear from catching on the cup as it is unloaded into a chute.

37.5.1 Object Detection

The first step was to determine whether or not the cup contains a spear. There was specular reflection from the bottom of an empty cup, which made it appear as if a spear was present. Since the spear extended past the ends of the cup and there was a dark background, this was the best place to detect the presence of a spear. The mean and variance were calculated of the pixel

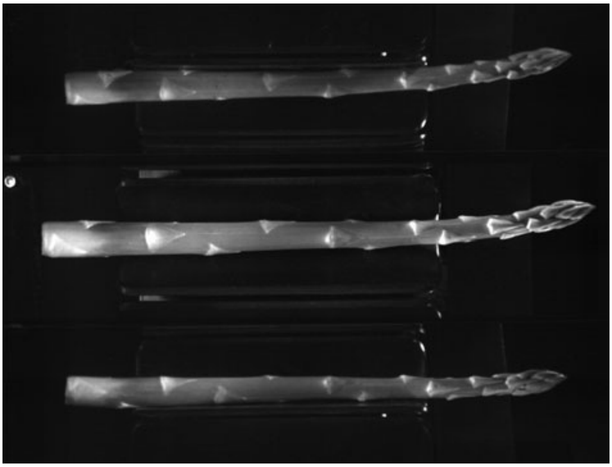


Fig. 37.7
A typical image captured by the system

values within a 96 × 1 window [3] at a predefined position between the end of the spear and the end of the cup from the central view. When no spear was present, only the background was within the window, so the mean and variance were both small. When both exceeded a threshold, the presence of a spear was indicated.

The image was also checked for the presence of the index LED, seen on the left side of *Fig. 37.7*. This was used to associate an image with the reference cup, as described in a later section. The index LED is in a fixed location enabling its state to be determined easily.

If no spear was detected, processing returned at this point until the next image.

The next step was to verify the exposure of the image. Since electronic shutter control was used, it is possible to compensate automatically for any variations in exposure resulting from aging of the lamps or variations in the average colour of the produce throughout the season. An intensity histogram was accumulated of the captured image, and the 99th percentile determined. This gave the highlight pixel value while ignoring the specular reflections from the cup. If the highlight value was greater than 250 there was a danger of losing important detail through the image saturating. The over-exposure was compensated by increasing the shutter speed for subsequent images. If the highlight value was less than 192 there was a danger of losing detail because the full dynamic range of the image was not being exploited. This under-exposure was compensated by decreasing the shutter speed for subsequent images. Such exposure adjustment was effectively using fixed step size integral feedback control. The feedback gain (step size) must be sufficiently small to prevent instability resulting from natural differences in colour from item to item. The large dead band allowed for the natural range of normal highlight pixel values and avoided changing the shutter speed with each image.

37.5.2 Segmentation

In the earlier images (before the mirrors and lights were changed) there was a stronger specular reflection from the sides of the cups. The initial approach to segmentation was to mask out the edges of the cup and threshold the image to obtain the shape of the spears [4]. This worked reasonably successfully in the laboratory, but did not work reliably when tried in the factory. First, the position of the cup varied slightly from one image to the next. This, combined with the fact that the spear is immediately adjacent to the cup in the side images, meant that either part of the cup would remain in the image or part of the spear would be clipped out. The second problem was that there was limited light on the spear between the cup and the spear. Therefore, the edge of the spear adjacent to the cup was not segmented reliably by a simple threshold.

Attempts to first detect the cup and then the edge of the spear gave improved results, but was still not sufficiently reliable for grading in the factory.

After modifying the mirrors and lighting, a new approach was tried. This used a simple 3 × 3 Prewitt filter to detect horizontal edges, as shown in the left panel of ➲ *Fig. 37.8*. Then the light and dark boundaries were traced using a boundary tracking algorithm. Such an approach only actually processes around the boundary points it tracks, making the processing is very efficient. Again, the basic principle worked most of the time, with the most common fault being the detection and tracking along the cup edge. More complex preprocessing (right panel of ➲ *Fig. 37.8*) gave better boundary definition, and more sophisticated boundary tracking rules improved the performance. However, it still failed to detect the boundary reliably once in about every 200 or 300 images. While a 99.6% success rate would be enviable in many applications, this corresponds to one error every 15–20 s when operating at full speed.

Since boundary tracking could not be made to work with sufficient reliability, it was necessary to revert to full object detection. The processing sequence [3] is illustrated in ➲ *Fig. 37.9* for a typical image. The first step was to enhance the contrast between the spear and the background, while suppressing irrelevant details within the spear. To keep the processing fast, the contrast was enhanced using a fixed lookup table at the same time as calculating the histogram for exposure adjustment.

The specular highlights were then removed by a greyscale closing operation with a 1 × 9 pixel window. Rather than using the usual sequence of an erosion followed by a dilation, the closing was performed in one pass by scanning to locate the local maxima and then back-filling to remove the peaks.

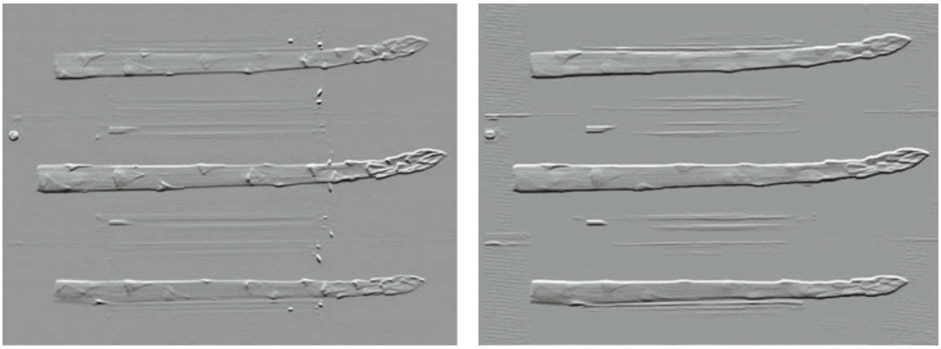


Fig. 37.8

Edge detected image used for boundary tracking. *Left*: original filtered image. *Right*: image with preprocessing prior to filtering

To remove much of the remaining background clutter, a background subtraction was performed. The image was split into three sections (one for each view) and the average pixel value, μ, calculated within each column for each section. Pixels with value smaller than the column average were considered background and were set to zero. The remaining pixel values were linearly expanded to maintain optimum contrast:

$$\hat{p}_i = \begin{cases} 0 & p_i \leq \mu \\ \frac{p_i - \mu}{255 - \mu} & p_i > \mu \end{cases} \tag{37.9}$$

This then enabled the image to be segmented using a fixed threshold level. Some of the cup edges were still classified as object; these were almost completely removed by a binary opening filter using a 9 × 1 element window. Again, an efficient implementation was used – if there were fewer than nine consecutive white pixels in a column, they were set to black. A similar opening and closing were performed using a horizontal 1 × 5 window to clean up the image. The final processed image is shown in the bottom right panel of *Fig. 37.9*. Note that some images still had a few pixels incorrectly segmented, although these represent a relatively minor error. This processing was sufficiently reliable that accurate weight estimates were provided for all spears correctly loaded into the cups.

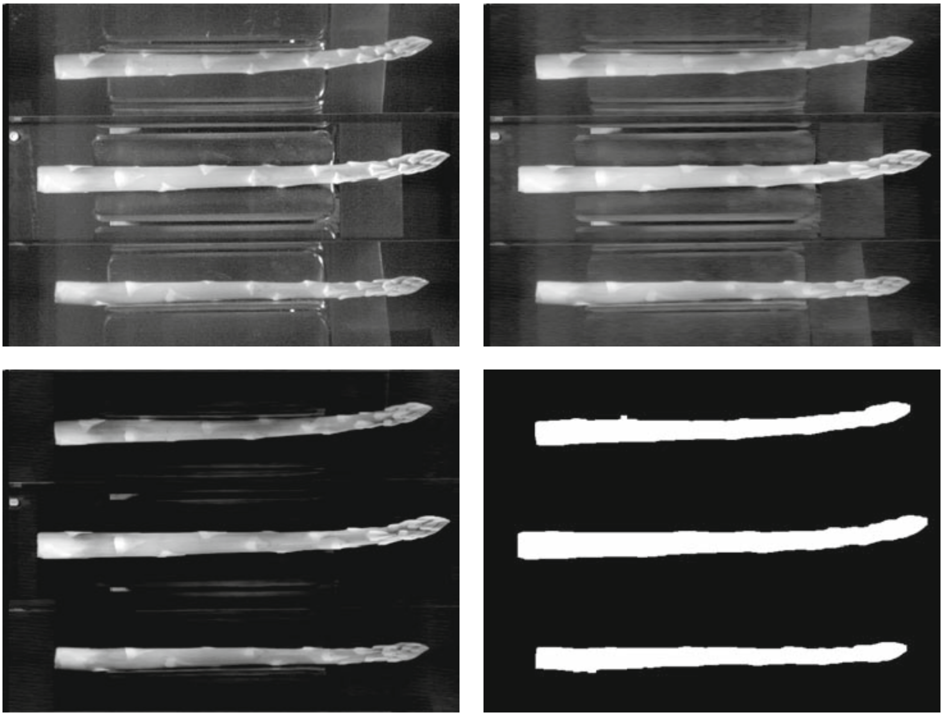


Fig. 37.9

Processing sequence. *Top left*: contrast enhancement. *Top right*: morphological filtering to remove specular highlights. *Bottom left*: background subtraction. *Bottom right*: thresholding and morphological filtering to clean the shape

37.5.3 Weight Estimation

The volume was calculated from the two side views using ➋ Eq. 37.8, where the diameter was determined by counting the pixels across the spear in each column. Since the edge of each spear was detected to the nearest pixel, at each position the measured diameter may be up to one pixel out. As a typical spear is 30–40 pixels diameter, this limited the accuracy to about 3%.

It would be possible to measure the diameter to sub-pixel accuracy by refining the detected edges using the gradient information from ➋ *Fig. 37.8*. The locations of the local minima and maxima gradient can be estimated to a fraction of a pixel [1] by fitting a parabola to the values of the extreme pixel value and the values immediately above and below it in the column. However, this was not needed, as the measured volume was already sufficiently accurate.

To convert the volume estimate in pixel units to a weight, two factors are required. The first is the size of each pixel, and the second is the density of the produce. These could be obtained through open loop calibration with a constant parameter for the proportionality constant in ➋ Eq. 37.8. A limitation of this approach is that there are small, but consistent, variations in density throughout the season and possibly throughout the day. The most accurate weight estimate may be obtained by dynamically determining the calibration as the density changes. A closed loop calibration is effective if the density is consistent between spears, and only changes slowly relative to the frequency of calibration.

Closed loop calibration was performed by actually weighing a random sample of spears as they were measured. Every 100th to 500th item (one spear every 10–30 s) was directed down a chute onto a set of electronic scales. This associated the digital volume with the known weight, providing a single point on the calibration curve. A least squares fit through the data from the last 100 calibration points provided a dynamic calibration curve. It also automatically compensated for any small variations in density with volume. The complete curve was replaced every 30–60 min, tracking small variations even during the day. After being weighed, the calibration items were redirected back to the input of the grading line where they were graded with the rest of the produce.

A further calibration issue is the difference in scale between the direct, central view and the two reflected side views. This is required to relate measurements between the three views for quality grading. To estimate the scale factor for a single spear, the length in pixels from the central view was divided by the average length in pixels from the two side views. Robust averaging (by eliminating outliers) over a large number of spears was used to provide an accurate estimate of the scale factor.

37.5.4 Quality Grading

One common application of machine vision is in quality grading, and this is particularly so in the case of produce. There is little prior work recorded in the literature of quality grading of asparagus. Rigney and Brusewitz [12] investigated 12 parameters derived from spear boundary data. They found error rates of up to 30% over manual classification. They also developed a machine vision system [13] for detecting spreading tips, broken tips, and scarred or cracked spears, with mixed results. Part of their context was mechanised harvesting; in the context here, with manual harvesting, many of these defects would be identified and culled in the field before grading.

In our system, in addition to estimating the weight, image processing was also used for a limited quality assessment of each spear. A complete inspection of each spear is not possible because only about 65% of each spear is visible to the camera. This is further reduced because the parts of the spear adjacent to the cup have limited lighting, making subtle defects all but impossible to detect in the side views.

The most important assessment is based on the ellipticity of the cross-section. While in principle it is possible to determine the ellipticity from three projections, the formula is quite complex. There is also little point in estimating the true ellipticity because the cross-section is seldom a true ellipse. However, a significant difference in the diameter of the three views (after correcting the central view for the difference in scale) indicates a significant deviation from circularity. If the ratio between the largest and smallest diameters at the base of the spear exceeded a threshold, the spear was rejected as being too flat.

It was also important to check that the field cut from harvesting had been completely removed when the spear was trimmed prior to grading. The angle of the butt end of the spear was measured in all three views. If any part of the base in any of the views deviated from perpendicular by more than a threshold angle then the field cut has not been completely removed. There was another trimming saw part way down the grading line that can further trim the spear. The known position of the saw was used to calculate the trimmed volume (and hence weight) for grading a spear into chutes beyond the second saw.

The field cut was usually made below the ground. The base of the spear from below soil level is lighter in colour than that from above the ground. The central view was examined for a change in contrast that indicated the position of ground level when the spear was harvested. If a contrast change was detected, as shown in ➲ *Fig. 37.10*, the spear was directed for trimming by the second trimming saw, and the volume (and weight) estimate adjusted accordingly. If this was insufficient to remove the base, then the spear was rejected.

The final quality assessment was to estimate the curvature of each spear from the three views. While there is no different in the taste between straight and bent spears, straight spears are significantly easier to package, especially by automated machinery. The deviation of the spear centreline from straight provided a simple measure of curvature. If the total deviation was greater than a preset threshold then the spear was rejected.

37.5.5 Measurement Summary

Speed required the image processing operations to be kept very simple. Standard operations used were histogramming, lookup tables, column averaging, and relatively simple morphological filters. Details of such operations may be found in any image processing text (see earlier

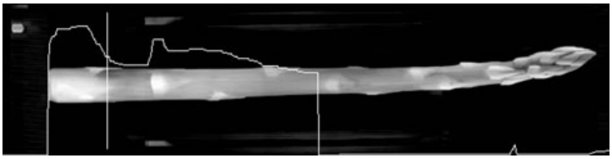


Fig. 37.10
Intensity profile of the base of the spear for detecting ground level

chapters in this handbook, or refer for example to [6,9,15]). The difference between what was used here, and a conventional implementation was that all of the code was optimised specifically for this processing task to minimise the computation. Successive operations were combined together where possible to reduce the overhead of unnecessary memory accesses.

The accuracy of the weight estimate of each spear is determined by a number of factors. These include variations in density from one spear to the next; errors in estimating the volume from the image; and the accuracy of the calibration from volume to weight. As described earlier, there is about a 1% variation in density, and the volume estimation has an uncertainty of about 3%. Since the calibration is determined by actually measuring the spears, each calibration point would have an expected uncertainty of about 4%. However, since the calibration results from fitting to 100 points, then the expected calibration error should reduce by a factor of 10. Therefore, the total error in measuring any one item is expected to be about 4.5%.

➋ *Table 37.1* summarises the results of weighing 2,500 items. For convenience, the spears have been divided into weight ranges based on the number of spears per 100 g bundle. Note that this might not be the actual grade that an individual spear is assigned, but it serves to provide a convenient reference. The RMS error is the difference between the true measured weight and the weight estimated by the system. The error is about the level expected, between 4.2% and 5.2%, with larger errors for the smaller spears. This is not unexpected, since the smaller spears have fewer pixels across the diameter, which will reduce the accuracy of the volume estimation. However, when the weight of a bundle is considered, the central limit theorem will reduce the relative error because the errors from measuring each spear are independent. The final column in ➋ *Table 37.1* therefore reflects the expected error in a 100 g bundle.

37.6 Grading Algorithm

Assigning a spear to a bundle based purely on a weight range results in a very broad spread of bundle weights, even if the weight of each spear is known exactly. This is illustrated in the left

Table 37.1
Summary of results from weighing 2,500 items

Bundle size	Weight range (g)	Count	Mean weight (g)	RMS error (g)	Error (%)	Bundle error (%)
Underweight	<13.3	159	11.7	0.70	6.04	
Seven per bundle	13.3–15.4	210	14.4	0.75	5.19	1.96
Six per bundle	15.4–18.2	430	16.8	0.78	4.67	1.91
Five per bundle	18.2–22.2	606	20.2	0.89	4.41	1.97
Four per bundle	22.2–28.6	649	25.0	1.13	4.51	2.26
Three per bundle	28.6–40.0	383	32.7	1.38	4.24	2.45
Overweight	>40.0	63	46.5	2.34	4.97	

panel of ➋ *Fig. 37.11* for an average bundle weight of 100 g. This was one of the problems with the previous grading scheme.

An accurate weight estimation allows a more complex grading scheme. It is now possible to allocate a spear to a particular bundle based on the weight accumulated so far [4]. For this, a target weight is calculated for each chute based on what it has already been allocated. For example, in a four items per bundle grade, the initial target weight would be 100 g/4 = 25 g. Suppose that a 22.6 g spear is assigned to that package. The new target weight would be (100–22.6)/3 = 25.8 g. This example is continued in ➋ *Table 37.2*. Each spear was allocated to the bundle with a target weight that most closely matches the estimated weight of the item. Underweight and overweight items were easily handled through the use of two-item (for the oversize) and ten-item (for the undersize) bundles.

To make effective use of the target weight scheme, it was necessary to be assembling several bundles within each grade, to allow more opportunity for finding the spear of the right size to complete the package. For bundles with small item counts, it is more difficult to achieve close

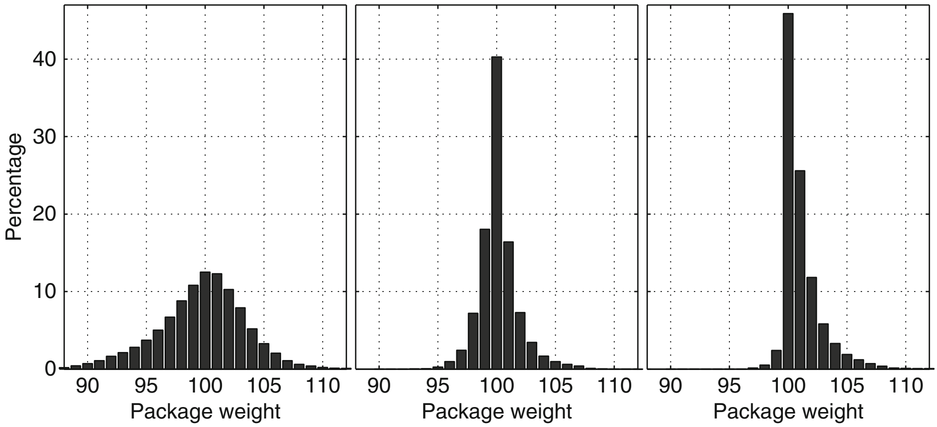


Fig. 37.11

Bundle size distributions with perfect weight measurement. *Left*: assignment based on weight range only. *Centre*: target-based assignment as described in the text. *Right*: target-based assignment with underweight penalty

Table 37.2

Example calculation of the target weight for a four per bundle grade as spears are assigned

Spear weight (g)	Total so far (g)	Remaining (g)	Spears remaining	Target weight (g)
	0	100	4	25.0
22.6	22.6	77.4	3	25.8
26.2	48.8	51.2	2	25.6
28.1	76.9	23.1	1	23.1
24.7	101.6			

to the target weight. A three-item bundle has an initial target weight per spear of 33.3 g. However, the next package size has an initial target weight of 25 g. So for a three-item bundle, items are selected from a relatively broad distribution, and there are few opportunities for selecting the best spear to complete the package. As the item count increases, the distribution becomes narrower, with the weights being added to a bundle with each spear being more similar. Consequently, the three-item bundles have a significantly larger spread in bundle size than the other counts. A balance was found by allocating four chutes for building three-item bundles and three chutes for each of the other grades. The effect of using a target weight for each spear is to give a much tighter bundle weight distribution, as shown in the centre panel of ➲ *Fig. 37.11*.

A problem remains with underweight packages. When a bundle requires only one more spear for completion, a severe penalty is applied to spears that are lighter than the target weight. The effect of such a penalty is to skew the distribution so that significantly fewer underweight bundles are produced. The resulting distribution of bundle weights is shown in the right panel of ➲ *Fig. 37.11*.

All of the distributions in ➲ *Fig. 37.11* are based on the assumption that the spear weights are known exactly. In practice, the measurement of each spear has a 4.5% error as described in the previous section. The effect of this error is to broaden the bundle weight distribution as shown in the left panel of ➲ *Fig. 37.12*. Note that the skew to overweight bundles is retained, but there is now a significant number of underweight bundles. The underweight tail cannot be eliminated completely, but can in part be compensated by increasing the target weight for each package. Setting the target weight to 104 g reduced the number of underweight bundles to an acceptable level. The resulting distribution is shown in ➲ *Fig. 37.12*.

➲ *Table 37.3* summarises the statistics for the different bundle counts. The average bundle weight was within the target of 105 g, with only about 2% of the packages being underweight. Most of these come from the smaller three-item and four-item bundles, which have a larger spread because of their small item counts.

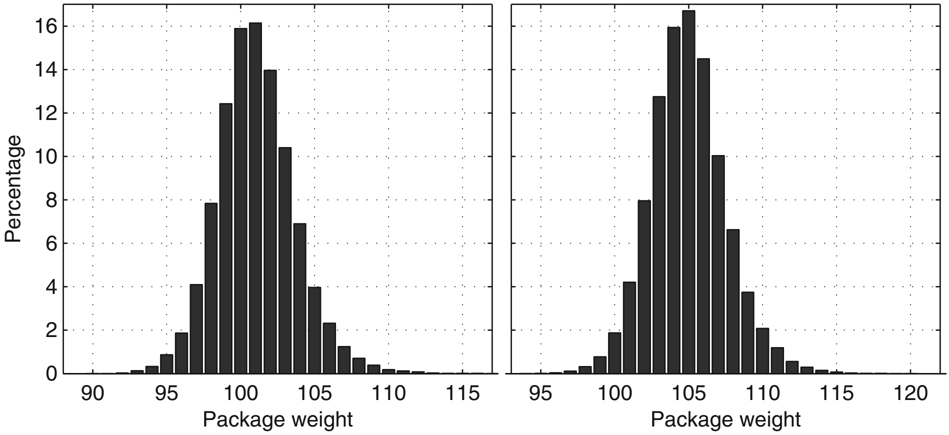


Fig. 37.12

Actual bundle weight distributions. *Left*: with a target weight of 100 g. *Right*: with a target weight of 104 g

Table 37.3
Statistics for the package weight distributions of the final system

Bundle size	Proportion (%)	Average (g)	Standard deviation (g)	% < 100 g	% > 110 g
seven per bundle	11.7	105.46	2.05	0.25	1.82
six per bundle	12.8	105.07	2.00	0.31	1.19
five per bundle	25.2	104.83	2.27	1.00	2.03
four per bundle	31.7	104.92	2.69	2.61	3.86
three per bundle	18.6	104.84	3.11	4.58	6.08
Overall	100	104.96	2.54	1.99	3.23

37.7 Software Implementation

LabVIEW for Windows was chosen as the platform for implementing the control and vision system because it readily enabled the different hardware components (image capture, chute control, and electronic scales) to be integrated within a single system. It also allowed different control and sorting algorithms to be simulated, and then used in the final production system without change. Tools were also available within LabVIEW for constructing a user interface that could be used as a control panel for the system.

Images were captured from the Sony XC-55 camera via a National Instruments frame grabber card. This also provided direct hardware triggering for image capture, which avoided problems with a variable software latency in responding to the trigger signal. Images were saved in multiple image buffers. This was necessary because Windows is not a real-time operating system, and can periodically switch to other tasks for periods of up to 200 ms. (All unnecessary functionality within Windows was disabled to reduce this. Networking in particular can induce some long delays, so all networking was switched off.) It is essential that no frames be missed or lost during this time, so a pool of 50 image capture buffers was pre-allocated.

All of the image processing routines were written as optimised C routines, and provided to LabVIEW using a DLL. These were then accessed through LabVIEW's DLL interface as custom operations. This approach was taken rather than use the image processing operations provided by the LabVIEW image acquisition module because it enabled the operations to be specifically tailored and optimised for this application. The use of custom image analysis software enabled several generic operations to be combined into a specific operation for this application. Algorithms for implementing standard operations were modified to significantly reduce the processing time required. The overheads of repeatedly allocating and freeing memory, and the associated memory fragmentation issues were avoided by pre-allocating sufficient scratchpad memory in LabVIEW, and passing this to the image processing routines with the image. Such optimisations enabled the final implementation to process each image in 35 ms on a 1.8 MHz Pentium P4 (a high end desktop PC when the system was developed). This corresponds to a maximum processing rate of 28 items per second, which was well within the target rate of 15 items per second. The image processing routines automatically measured and compensated for the scale difference between the side and central views and returned all of the measurement data as described above.

A LabVIEW module was developed to manage the dynamic calibration, which utilised a commercial electronic scale interfaced via an RS232 connection. The calibration curve was used to convert the measurements from pixel units to a weight. Based on the weight and quality measurements, another LabVIEW module graded the spear and allocated it to a chute. The number of the cup containing the spear was communicated to the actuator associated with a chute using an RS485 connection. Each spear was processed independently; once it had been allocated to a chute, and the data associated with the chute state updated, the spear was considered to have been processed.

The main control panel within LabVIEW contained stop and start buttons, the most recently captured image, and a summary of processing statistics. The latter two items were not strictly necessary, but were provided to enable the operator to see that the system was operating. In addition to the main panel, several setup screens enabled the target weight and grades for each chute to be specified. The settings of the trimming saws and the thresholds for the quality grading were also entered here. Several debugging screens were also developed, although they were disabled and inaccessible during normal operation.

37.8 Actuator System

To overcome problems with belt stretch, a distributed intelligent actuator system was introduced [10]. Each chute required an actuator, so sensing the cups at the point of actuation will be immune to variations in distance between the grading system and chute. Each actuator had to perform the following tasks: identify individual cups on the conveyor; communicate with the central grading system; buffer the list of cups containing spears destined for the associated chute; activate the solenoid for the specified cups; and release the completed bundles for packaging.

37.8.1 Actuator Controller

Several options were considered for implementing the actuator controller. Initial investigations considered using digital I/O cards mounted in the controlling PC. Typically, such cards have up to 32 I/O channels available so several would be needed to provide a sense input and two outputs for each of up to 60 chutes. Two problems were identified with this option. First, the operating system cannot time to the precision required. Second, the sensors and switching hardware would be mounted at some distance from the PC. The resulting long lines (up to 10 m) would be susceptible to noise interference. It would also be necessary to use individual solid-state relays to handle the current and voltage requirements of the solenoid and bundle release pneumatics.

A programmable logic controller (PLC) is designed to perform a function in response to attached sensors. Modern PLCs are networkable, so can be set up to receive instructions from the grading PC. Scan times can be as low as 1 ms, giving a 2 ms timing uncertainty which is satisfactory in this application. Again, to handle the current and voltage requirements it would be necessary to use external solid-state relays. To manage all of the chutes, several PLCs would be required, making this a relatively expensive option.

The relatively simple requirements for the actuator indicated that a small microcontroller would be a viable option. A Winbond W78E516B, an 8051 variant, was selected because it was flash programmable, and integrated all of the required features within a single chip: it is capable of accurate timing, includes serial communications, has ample I/O pins, and has a small amount of memory that can be used to buffer cup data. Power MOSFETs were used to provide the required actuation. The low cost of the microcontroller enabled one to be used for each chute, simplifying the software and making the system modular and flexible. The microcontrollers were powered from the 48V actuation system power supply using an integrated switch-mode regulator.

37.8.2 Communication

The intelligent actuators were connected to the grading system using an 115,200 bps multi-drop half-duplex RS485 network. The RS485 protocol uses two balanced lines, making it resistant to noise and DC offsets. This was important in this application as the momentary high currents when activating the solenoids could result in considerable movement of the ground voltage. MAX487 devices were used to interface the microcontrollers and the grading PC to the communication line.

Almost all communication was in packets of five bytes in the following format: address, command, two data bytes, and a checksum. A command/response protocol was used with all commands echoed back to the controlling PC to verify reception. This meant that the data direction on the bus was predictable, which was necessary with the half-duplex system. Failure to respond to a command indicated that the addressed actuator has encountered a fault, enabling the operator to be alerted.

Address zero was used to broadcast to all actuators; broadcast commands were obviously not echoed. Eighteen commands were defined. These included a number of query commands that were used primarily for debugging purposes; the echoed response contained the requested data.

Manual configuration of actuator addressing was avoided by using a self-addressing approach. The actuators were daisy chained using a select-in and select-out connection, with the select-out of one actuator connected to the select-in of the next. On power-up, all actuators started with an address of zero, and asserted their select-out low. Only the first actuator in the chain had a high select-in, so it would respond to a SET ADDRESS command and be assigned an address of 1. It then set its select-out high so that the next actuator in the chain would respond to the next SET ADDRESS command. In this way, successive actuators were assigned successive addresses. When no further replies were echoed, all actuators had been assigned addresses. If this differed from the expected number of actuators, the grading system could alert the operator which actuator had failed.

Automatic configuration in this way enabled all of the actuator controllers to be identical. This avoided the need to set jumpers to indicate the address, which is inevitably an error prone process. It also simplified maintenance, as all of the units were interchangeable.

With the large number of controllers, it was convenient to have in-circuit programming of the microcontrollers. A LOAD NEW PROGRAM command switched execution to a boot loader section of program memory, erased the main program flash, and waited to receive new code via the communication line. Either the whole chain could be reprogrammed at once, or an individual controller could have its code updated (for example if it was a replacement unit running an old version of the code).

There was considerable latency between when a spear was graded and when the solenoid needed to be activated to drop the spear into a chute. As soon as a spear was graded, a CUP ID command was sent to the actuator for the intended destination. The set of cup IDs for a chute were buffered by the actuator until the corresponding cups arrived and were triggered.

37.8.3 Cup Sensing and Identification

Several possible techniques were considered for cup identification. Since each cup must have a unique ID an obvious approach would be to install an RFID tag on each of the cups. Each chute would then require an RFID reader to sense the cup and its identity. However, at the time that the system was developed, none of the team had experience with RFID technology, and it was perceived to be overly complex for this application.

The initial approach was to mount a small neodium magnet on the cup trigger tags, and sense these using an Allegro 3121 Hall effect sensor, as shown in the left panel of ➲ *Fig. 37.13*. This enabled each cup to be detected as it passed the actuator. Individual cups were identified by counting the number of cups from a reference cup, which was assigned an ID of 0. The reference cup had two spatially separated magnets mounted on the trigger tag, which gave a double pulse when it passed the sensor, enabling it to be identified easily.

This approach appeared to work successfully, until after about a week it began behaving erratically, becoming increasingly unreliable. The cause was traced to the magnets mounted on the trigger tags. Each activation of the solenoid subjected a magnet to an electromagnetic pulse. The cumulative effect of these pulses was sufficient to partially or completely demagnetise some of the magnets depending on their polarity and their precise location relative to the solenoid. The net effect was that the signal from the Hall effect sensor became weaker, making cup detection unreliable.

The next approach was to use a reflective optical system based on a Sharp IS471. This is an integrated synchronous infrared modulator and detector, making it inherently immune to stray illumination. The modulator output directly drives an infrared LED, with the tag reflection detected and converted to a digital output indicating the presence of a tag. Both were positioned within the actuator enclosure, facing upwards through a thin window. The LED was inclined by 20° so that both the LED and detector were focussed on the same spot (see the right panel of ➲ *Fig. 37.13*). This gave a narrow detection range, with tags positioned between 5 and 10 mm above the sensor detected reliably. Objects further than 20 mm from the sensor were not detected.

The reference tag was made 50% wider than the other tags to enable it to be identified. The leading edge was the same, to enable reliable timing for solenoid activation, with the trailing edge extended. The length of the pulse produced by a tag depended on the speed of the conveyor, so the threshold was set at 125% of the normal tag transit time. To facilitate startup and shutdown when the conveyor speed was not constant, the tag transit times were filtered using a first order IIR filter.

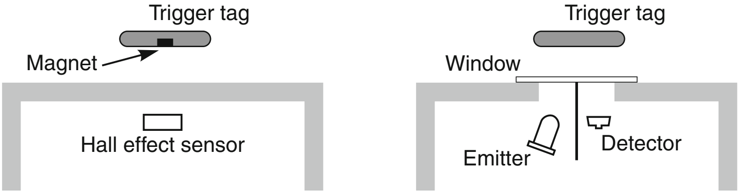


Fig. 37.13

Tag detection. *Left*: using magnetic sensing. *Right*: using synchronous optical sensing (Reproduced from [10], used with permission)

One limitation of the optical detection approach was that the viewing window became dusty from water droplets off the produce and general dust from within the packinghouse environment. This required cleaning approximately once every week as part of the regular maintenance cycle. The optical system was used within the packinghouse successfully for two seasons.

After this time, magnets with a higher coercivity were sourced that did not suffer from demagnetisation. The actuators were switched back to the Hall effect sensors because these did not have the same maintenance requirements as the optical system.

A cup sensor was also mounted at the camera to trigger image capture. When the reference cup was detected, the index LED was illuminated, enabling the image associated with cup 0 to be identified. This was important because with Windows there was a variable latency between capturing the image and the actual processing and subsequent grading.

37.8.4 Cup and Bundle Release

The actuators were required to trip the selected cups as they arrived at their destination chute. As the cup sensor was located ahead of the solenoid, when the required cup was detected, the signal to trigger the solenoid had to be delayed depending on the speed of the conveyor. Reliable triggering required the delay to be accurate to within about 5 ms. A cup was released by activating the solenoid via a power MOSFET for 20 ms. The solenoid was connected to the 48 V system power supply, drawing 4 A while activated.

When multiple solenoids were activated simultaneously, this injected considerable noise into the power supply. In the initial system, this voltage droop was sufficient on occasion to reset a microcontroller. This was overcome by providing each microcontroller with a 1,000 μF electrolytic capacitor to supply power during the short outages.

Off-the-shelf 24 V pneumatic valves were used to operate pneumatic cylinders that released completed bundles for packaging. The cylinder toggled with each bundle so on average each valve was energized half the time. If connected directly to the 48 V power supply, the valves would overheat. It was found that a 25% duty cycle was sufficient to reliably hold the valve closed. The microcontroller provided the required waveform via another power MOSFET.

A short delay was required between triggering the cup containing the last spear for a bundle and activating the pneumatic cylinder valve to allow time for the produce to fall from the cup into the chute. While a bundle was being released, spears from the next four or five cups could not be released at that chute. This was implemented by locking such a chute out of consideration during the grading process.

The actuator system worked reliably at up to 25 cups per second. Chute design and produce handling considerations limited the maximum practical speed to about 15 cups per second. Beyond this speed, the chutes needed to be repositioned or made wider to catch the spears being released from the conveyor.

37.9 Debugging

The initial system was installed on a test grading line within the laboratory. This enabled most of the obvious errors to be identified and corrected before installation in the packinghouse. Unfortunately, not everything scaled well between the installations. The conveyor within the test system had to be loaded manually, so it could not be operated fully loaded at full speed.

When installed within the packinghouse, the system operated fine when lightly loaded and running at low speed (conditions tested in the laboratory). However, when running under normal operating conditions, most of the produce went off the end of the conveyor, rather than into chutes. Debugging a system operating at 10–15 cups per second initially proved to be quite challenging, especially when the observed symptoms could result from many possible errors. Failure to detect a cup at the camera or chute would lead to miscounting and loss of synchronisation, resulting in the wrong cups being triggered. Image processing errors could lead to faulty weight estimates, or in severe cases result in the program terminating. Grading failure could result in a spear being allocated to the wrong bundle. Inaccurate timing of the solenoid could result in a cup failing to release. During the debugging phase, all of these errors occurred at various times. When there were so many possible sources of errors, finding the actual cause of a problem was difficult, especially when the system behaved correctly with low speed or throughput. As the system became more reliable, however, there were fewer errors and identifying the cause for a particular error became easier.

Debugging the image processing code was particularly challenging especially when an intermittent array bounds error caused the whole grading system to terminate. This was overcome by streaming each image to disk from LabVIEW before calling the image processing code. Then, when the programme failed, the image that caused the failure was available for analysis off-line. To facilitate debugging, the source code was also compiled within VIPS, a general purpose image processing algorithm development environment [2]. The image could then be analysed, the code edited and retested on the image until the bugs were removed. If necessary, the code could be single stepped in the debugger enabling the cause of the failure to be detected and remedied.

The other significant issue during the debugging stage was that the clients had the expectation that the system would work when first switched on. They took the debugging problems as evidence that the approach we had taken had failed. It took considerable persuasion to convince them that such problems were in fact expected when developing a complex system such as this. However, even with such reassurance, the visible delays in getting the system operational also led to pressure from the packinghouse, especially as the grading season approached. Even after the system was mostly working and in regular use, one or more members of the development team visited the packinghouse regularly to resolve the remaining problems.

After the system was working, it took some time for the results of the system to be accepted by the clients. During this period, they manually weighed every bundle produced by the system to verify that they were neither underweight nor significantly overweight. Eventually, however, they realised that this was unnecessary, and that the system was working as designed.

37.10 Summary

This chapter has described the process of developing a machine vision system for automated produce grading. It was a multi-disciplinary engineering effort, involving optics, lighting, modelling, image processing, statistics, software development, real-time systems, mechanics, electronics, and debugging skills.

Conventional weighing methods are very expensive for high speed grading and packing. Accurate, high-speed weight estimation of produce using machine vision has been demonstrated. Two perpendicular views were utilised to obtain an estimate of the volume of each item, which was then related to the weight through a closed-loop calibration. Processing speeds approaching 30 items per second on a desktop PC have been demonstrated, with an accuracy of 4–5% in measuring each individual item. This led to more accurate grading and better control over the spread of package weights. An average package weight of 105 g was achieved, with approximately 2% of the packages slightly underweight.

Attempts were made to automate as much of the system as possible, to reduce the opportunities for operator error. This led to automatic exposure adjustment, dynamic calibration, and automated configuration of the actuators. The system was also able to self-diagnose many of the faults that could occur.

Improved processing efficiencies resulted in a doubling of packing throughput. Better control over the bundling process enabled this higher throughput to be managed with fewer staff (the relatively short packing season made obtaining appropriate staff difficult), resulting in significant cost savings for the packinghouse. More accurate weight estimation reduced the average bundle weight by approximately 20%, increasing the sales and profitability for the grower by 20% for the same amount of produce. The reduced weight per package also decreased the air freight costs.

The success of the project is best demonstrated by the fact that the grading system is still in use in the packinghouse 6 years after it was developed.

Acknowledgements

A large project such as this is never a one-man effort. My role in the project related to the image and signal processing aspects. The other members of the development team were: Ralph Ball and Harvey Barraclough, who focussed on the mechanical engineering aspects of the project; Ken Mercer, who developed the intelligent actuators; Colin Plaw, who developed the LabVIEW based control system; Andrew Gilman, who investigated high-speed weighing using load cells; and Geoff Lewis from the packinghouse, who provided the test system and copious quantities of asparagus for testing.

We also acknowledge funding for this project from Technology New Zealand through a Technology for Business Growth Grant.

References

1. Bailey DG (2003) Sub-pixel estimation of local extrema. In: Image and Vision Computing New Zealand (IVCNZ'03), Palmerston North, New Zealand, 26–28 November, 2003, pp 414–419
2. Bailey DG, Hodgson RM (1988) VIPS – a digital image processing algorithm development environment. Image Vis Comput 6(3):176–184. doi:10.1016/0262-8856(88)90024-8
3. Bailey DG, Mercer KA, Plaw C, Ball R, Barraclough H (2004) High speed weight estimation by image analysis. In: 2004 New Zealand National Conference on Non Destructive Testing, Palmerston North, New Zealand, 27–29 July, 2004, pp 89–96
4. Bailey DG, Mercer KA, Plaw C, Ball R, Barraclough H (2001) Three dimensional vision for real-time produce grading. In: Machine Vision and Three-Dimensional Imaging Systems for Inspection and Metrology II, Boston, Massachusetts, USA, vol SPIE 4567, 29–30 October, 2001, pp 171–178, doi: 10.1117/12.455254
5. Ball T, Folwell RJ (2004) Economic analysis of alternatives to manual sorting using selected electronic graders in asparagus fresh packing. J Food Process Preserv 28(6):405–416. doi:10.1111/j.1745-4549.2004.22150.x
6. Davies ER (2005) Machine vision: theory, algorithms, practicalities, 3rd edn. Morgan Kaufmann, San Francisco

7. Gilman A, Bailey DG (2005) High speed weighing using impact on load cells. In: IEEE Region 10 Conference (IEEE Tencon'05), Melbourne, Australia, 21–24 November, 2005, doi: 10.1109/TENCON.2005.301118
8. Gilman A, Bailey DG (2002) Using the impact on load cells for high-speed weighing. In: Ninth Electronics New Zealand Conference (ENZCon'02), Dunedin, New Zealand, 14–15 November, 2002, pp 12–17
9. Gonzalez RC, Woods RE (2008) Digital image processing, 3rd edn. Prentice-Hall, New Jersey
10. Mercer K, Bailey DG, Plaw C, Ball R, Barraclough H (2002) Intelligent actuators for a high speed grading system. In: Ninth Electronics New Zealand Conference (ENZCon'02), Dunedin, New Zealand, 14–15 November, 2002, pp 61–65
11. Morales-Blancas EF, Cárcamo RC, Cisneros-Zevallos L (2001) Determination of shape parameters for asparagus spears by 3-D digitizing. In: 2001 Institute of Food Technologists Annual Meeting, New Orleans, Louisianna, 23–27 June, 2001, paper 88C–32
12. Rigney MP, Brusewitz GH (1992) Asparagus shape features for quality assessment. Trans Am Soc Agric Eng 35(5):1607–1613
13. Rigney MP, Brusewitz GH, Kranzler GA (1992) Asparagus defect inspection with machine vision. Trans Am Soc Agric Eng 35(6):1873–1878
14. Robinson JE, Browne KM, Burton WG (1975) Storage characteristics of some vegetables and soft fruits. Ann Appl Biol 81(3):399–408. doi:10.1111/j.1744-7348.1975.tb01656.x
15. Russ JC (2002) The image processing handbook, 4th edn. CRC Press, Boca Raton

38 Analysing the Creasing Properties of Fabric

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_38,

Abstract: A large clothing retail store wishes to quantify the creasing properties of the fabrics used in shirts, blouses, dresses, etc. Hitherto, creasing has been judged subjectively and inaccurately, by comparing a sample with carefully designed models, photographs, etc. There is a need for a more objective measurement than this provides and a laboratory-based vision system to automate this task has been proposed. A sample of plain unprinted fabric is first creased in a standard way and then stretched gently as it is placed onto a table. Lighting at a low angle is then applied from one side. The image generated in this way has a low contrast but it is sufficient to allow the crease furrows to be seen. Results are presented for several non-linear filters that convert the grey-scale texture image to binary form. Measurements describing the resulting binary texture are defined. A multi-element vector created by combining several such measurements is then presented to a Pattern Recognition system. This is then trained to identify the creasing coefficient defined by the existing human inspector.

38.1 Background

In this short chapter, several methods are presented for analysing texture. These are discussed in relation to measuring the susceptibility of plain white fabric to creasing. No single method is identified as being superior to all others, as our aim is to demonstrate that there may be many *reasonable* solutions for a given application. It is important to be able to evaluate these easily and an interactive image processor (QT ➲ Chap. 21) has proved to be invaluable. Using QT in this role requires a slightly different approach from that needed for discrete machine parts. As usual, a single image is studied to find tentative ideas for filtering the texture. However, these must then be tested on many different samples of texture, to verify that the measurements and pictorial results that the program is robust. Texture often requires the derivation and analysis of several different measurements and this suggests the use of Pattern Recognition and/or regular statistical techniques. It would not be meaningful to present a complete solution to the task of characterising the creasing properties of fabrics, without including a lot of unnecessary detail about the application requirements and other background information. The author worked on the assumption that the texture analysis procedure would consist of three parts:

- *Stage 1*: Converting the texture in the original grey-scale image to binary form.
- *Stage 2*: Characterising the binary texture. (It is axiomatic that this is easier than characterising the original grey-scale image directly.)
- *Stage 3*: Relating the measurements obtained in Stage 2 to the subjectively defined texture classification currently used by the customer. The details are specific to each application and so will not be discussed further. Let it suffice to say that, to do this properly, requires experiments properly designed in accordance with the established experimental practice of psychology and formal statistical analysis.

38.1.1 Inspection Tasks

A certain UK clothing retailer wishes to improve the quality of the products it offers for sale. To do this, the company will impose strict in-house testing and inspection of the garments it receives from its suppliers. Among the important features that are to be measured is the resistance of fabric to creasing. At the moment, creasing is measured subjectively, by comparing the test sample with a set of standard images. The test sample is a piece of plain fabric that is crumpled manually and then flattened by hand. It is then viewed in a specially designed cabinet which provides low-angle (grazing illumination) Lighting is provided by a linear fluorescent tubes and the cabinet limits the effects of any variation in the ambient lighting. Since any form of physical contact with the fabric will distort the undulations due to creasing, optics and Machine Vision are the obvious technologies and are probably the only ones that offer any realistic hope of automating the measurement function. The task, then, is to design a vision system that can quantify the creasing characteristics of plain fabric. The garment supplier will also be required to install the same test equipment.

38.1.2 Related Tasks

Inspection tasks that provide similar requirements are to be found in industries involving:

1. Machined (i.e., milled, turned and ground) metal surfaces
2. Paper
3. Plastic film
4. Cork floor tiles

38.1.3 Image Formation

From the long experience gained by the clothing retailer with the traditional method of assessing creasing, it is known that grazing illumination is optimal, or very nearly so. ➋ *Figure 38.1* shows a sample image, obtained from a plain white linen handkerchief. To obtain this image, a non-collimated beam was projected from the left. Notice that the "back-ground" intensity varies across the picture, although this can be eliminated effectively, using spatially invariant lighting. Local intensity variations, due to creasing, indicate the varying angle of incidence. Shadows within the "valleys" were avoided by making sure that the light source was not too low.

38.2 Inspection Algorithms

The inspection algorithm consists of three parts:

1. Filtering and thresholding to generate a textured binary image
2. Generating a multi-variate description of that binary image
3. Associating that description with the creasing factor

38.2.1 Stage 1: Filtering

Six methods of processing this image to generate a textured binary image are illustrated below. Any of these might reasonably be expected to form the basis of an effective solution and could be combined with any of the binary texture analysis functions, Stage 2. Asterisks (*) indicate lines

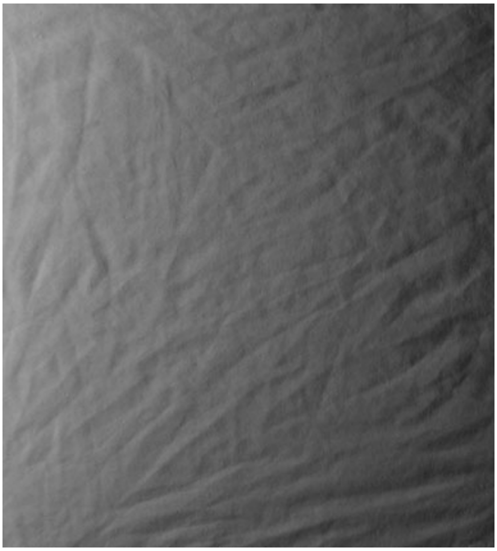


Fig. 38.1
Sample image, creased fabric. No attempt was made to optimise the image quality; better lighting would make the "back-ground" more nearly constant. This was done deliberately, so that the effectiveness of the image filtering algorithms could be demonstrated better

where parameter values might be adjusted to optimise results for different levels of magnification. This is best performed experimentally, which is a role for which QT is well suited.

It is not possible to explain, how each of the scripts listed below evolved while using QT. Human intelligence plays a key part in identifying branches on the search tree that will lead eventually to a good solution. How such decisions are made is a mystery!

38.2.1.1 Method 1: Difference of Low-Pass Filters

The following QT command sequence runs moderately fast and produced the image in *Fig. 38.2*.

```
enc             % Enhance contrast
mdf(7)          % Median filter noise removal*
caf(11)         % Low-pass filter*
caf(17)         % Second low-pass filter*
sub             % Subtract results
heq             % Histogram equalise
thr             % Threshold at mid grey. This command following heq
                  produces an image in
                % which approximately 50% of the pixels are white
maj             % Majority voting filter
edg(8,0)        % Mask edge effects
```

38.2.1.2 Method 2: Local-Area Histogram Equalisation

The following is a rather slow QT command sequence and produced the image in *Fig. 38.3*.

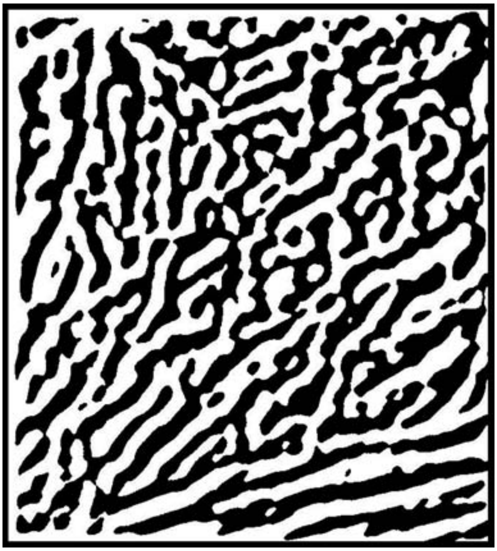


Fig. 38.2
Filtering and thresholding, Method 1

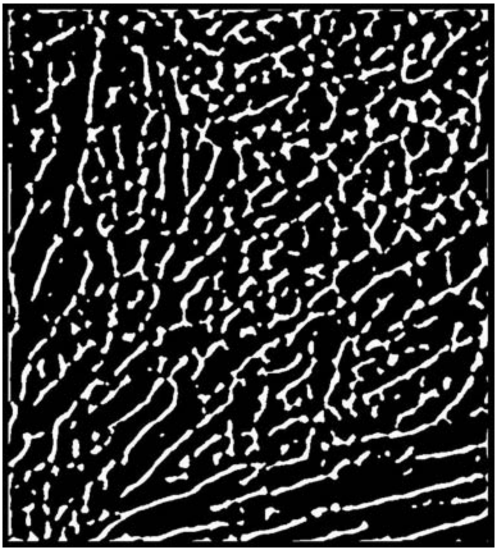


Fig. 38.3
Filtering and thresholding, Method 2

```
enc               % Enhance contrast
mdf(7)            % Median filter noise removal*
caf(7)            % Low-pass filter (averaging over a circular region)*
lhq(15,15)        % Local-area histogram equalisation (This is the slow part.)
thr(128)          % Threshold*
maj               % Majority voting filter
edg(8,0)          % Mask edge effects
```

38.2.1.3 Method 3: Direction of Brightest Neighbour

The following is a moderately fast QT command sequence and produced the image in *Fig. 38.4*.

```
enc                                 % Enhance contrast
mdf(7)                              % Noise removal*
caf(7)                              % Low-pass filter*
dbn                                 % Brightest neighbour
thr(4,4)                            % Select level 4 only*
maj                                 % Majority voting filter
edg(8,0)                            % Mask edge effects
```

38.2.1.4 Method 4: Unsharp Masking

The following is a relatively fast script and produced the image in *Fig. 38.5*. It is akin to Method 1.

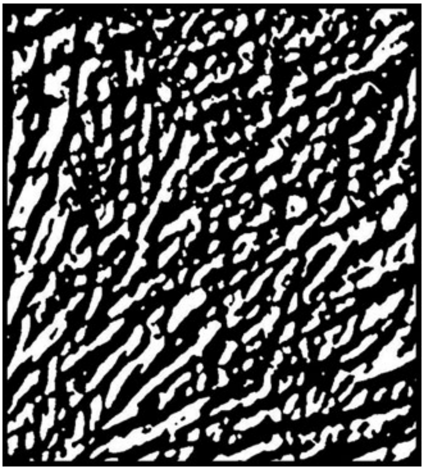


Fig. 38.4
Filtering and thresholding, Method 3

```
enc          % Enhance contrast
mdf(7)       % Noise removal*
caf(15)      % Low-pass filter*
sub          % Subtract
heq          % Histogram equalisation
thr          % Majority voting filter
edg(8,0)     % Mask edge effects
```

38.2.1.5 Method 5: Grey-Scale Opening

The following is a relatively fast script and produced the image in ➋ *Fig. 38.6*.

```
enc          % Enhance contrast
mdf(7)       % Noise removal
opn(15)      % Opening*
sub          % Subtract
heq          % Histogram equalisation
thr          % Threshold
maj          % Majority voting filter
edg(8,0)     % Mask edge effects
```

38.2.1.6 Method 6: Subtracting a Reference Image Created with *rox*

The following is a relatively fast script and produced the image in ➋ *Fig. 38.7*

```
enc          % Enhance contrast
mdf(7)       % Noise removal*
wri          % Save preprocessed image
neg          % Negate
rox          % Row maximum
neg          % Negate. This and two previous commands perform row minimum
wri(1)       % Intermediate result
rei          % Read pre-processed image
lrt          % Invert horizontal axis
rox          % Row maximum (runs from left to right)
lrt          % Invert horizontal axis. This and two previous commands
               perform row maximum
             % running from right to left
rei(1)       % Intermediate result
adi          % Add images
rei          % Read pre-processed image
sub          % Subtract
heq          % Histogram equalisation
thr          % Threshold
```

```
maj          % Majority voting filter
edg(8,0)     % Mask edge effects
```

38.2.2 Stage 2: Analysing Binary Texture

Several methods are available for analysing binary texture; again, just a few of the many possibilities are described below. The illustrations are based on the image generated by Method 6.

(a) Count "zero crossings" (black–white or white–black transitions) in preferred directions: horizontal, vertical, ±45°.

Example: count all zero crossings:

```
bed          % Binary edge
cwp          % Count white points
```

Another example, count zero crossings along the horizontal axis

```
con([0,0,0; −1,1,0; 0,0,0])    % Intensity difference between horizontal
                                 neighbours
thr(200)                       % Keep black-white transitions
x = cwp                        % Count white points
```

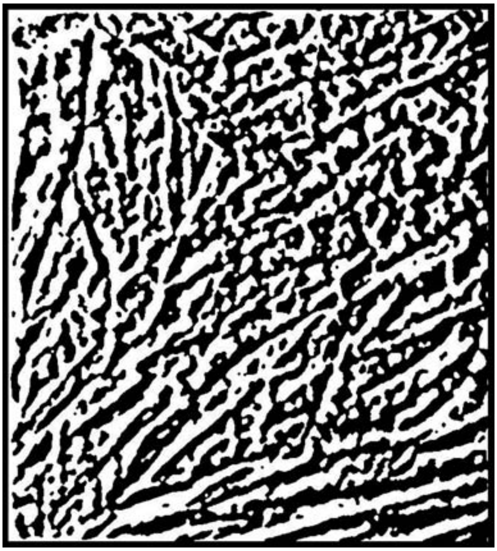
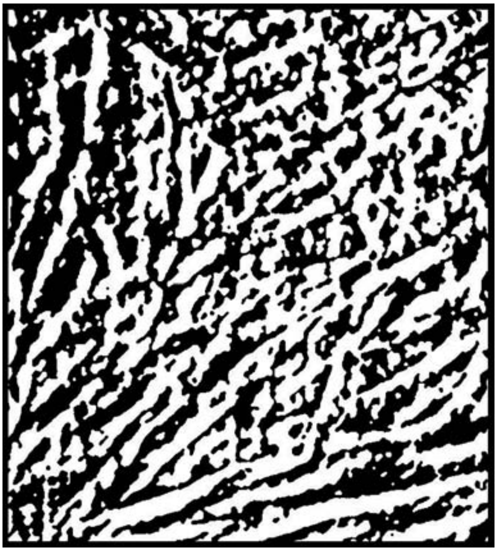
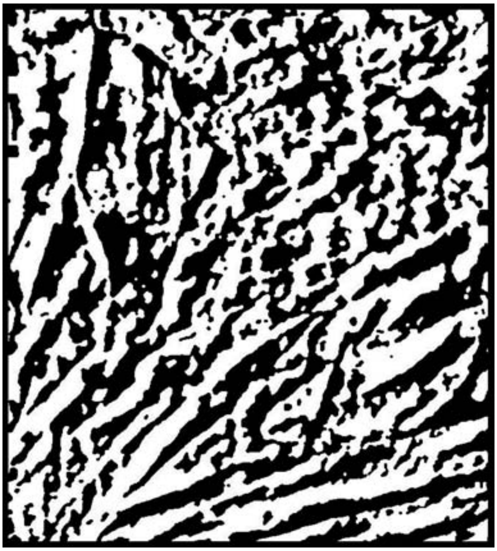


Fig. 38.5
Filtering and thresholding, Method 4



■ **Fig. 38.6**
Filtering and thresholding, Method 5



■ **Fig. 38.7**
Filtering and thresholding, Method 6

(b) Size distribution of the blobs and spaces between blobs using *gft*:

```
gft          % Grass-fire transform measures sizes of
               white objects
wri          % Save image
swi          % Switch images
neg          % Negate
gft          % Grass-fire transform
rei          % Read image saved earlier
sub          % Subtract
hpi          % Display graph plotted in Fig. 38.8
x = dev      % Standard dev. sensitive to blob size
```

The graph shown in *Fig. 38.8* was obtained by applying this QT command sequence to the image in *Fig. 38.7* (*Method 6*)

(c) Horizontal run-lengths of the blobs and spaces between blobs using *rlr*. (Use *rlc* for measuring vertical run-lengths.)

```
rlr          % Run-length along each row
wri          % Save image
swi          % Switch images
neg          % Negate
rlr          % Run-length along each row
rei          % Read image saved earlier
sub          % Subtract
hpi          % Display graph – similar to that above
x = dev      % Standard dev. sensitive to blob size
```

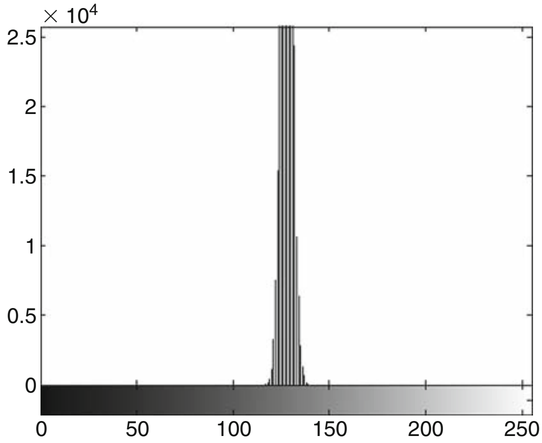


Fig. 38.8

Texture analysis. The width of the peak is sensitive to the sizes of the blobs and the spaces between them, which are dependent on the amount of creasing

This produces a graph similar to that shown in *Fig. 38.8*. Again, by measuring the width of the peak, it is possible to estimate the creasing factor.

(d) Morphology with directional structuring elements. The following table shows the effect of applying the erosion operator, with linear structuring elements inclined at ±45°, to the image shown for Method 6. In this case, the area (number of white pixels, function cwp) is more stable than the blob count (cbl). Further measurements can be obtained by using structuring elements at other angles (e.g., 0° and 90°) and by using dilation. A combination of erosion in one direction and erosion in the orthogonal direction could also be used to good effect.

Length of structuring element	Structuring element angle = + 45°		Structuring element angle = −45°	
	No. of blobs	Area	No. of blobs	Area
11	149	141,914	207	95,129
21	186	79,936	207	25,400
31	153	45,437	83	6,267
41	117	26,994	27	1,607
51	76	16,958	10	300

(e) Morphology with variable-size structuring elements. This is a variant of the method of analysing binary texture just described. It uses cross-correlation to measure the effects of (morphological) opening, with variable-size structuring elements. Several values are computed and are then used to derive a polynomial equation. In this way a statistical measure, estimating blob size and inter-blob spacing in the binary image, is calculated. Here is a function to do this:

```
function p1 = analyse_binary_texture(n,r)
% analyse texture in a binary image by applying opening with variable-size
structuring elements (1/2)
global current
global alternate
original = current;                    % Keep image
for i = 1:n                            % n is size of biggest. structur-
                                         ing element
current = original;                    % Recover original image
opn(i),                                % Opening – see notes below
alternate = original;
a(i) = icc;                            % Correlation coefficient
end
figure(2)                              % Will plot graph in figure no. 2
plot(a,‘xk’)                           % Plot points just found
title(‘‘Correlation coefficient’’)     % Title for graph
hold on                                % Keep what we have plotted
p1 = polyfit(1:n,a,r)                  % r-th order polynomial fits a(1:n)
```

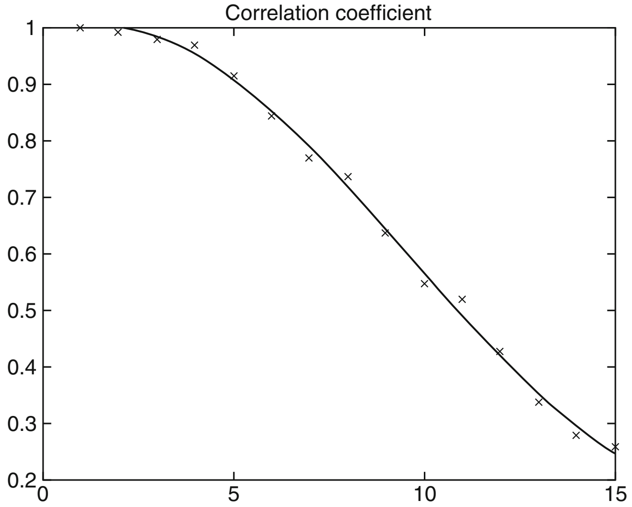


Fig. 38.9
Creasing is measured by finding the position along the abscissa where the correlation coefficient is equal to 0.5

```
q1 = polyval(p1,1:0.1:n);               % Evaluate polynomial
plot(1:0.1:n,q1,'-k')                   % Plotting polynomial
hold off                                % Finished plotting in figure no. 2
figure(1)                               % Revert to figure no. 1
swi                                     % Switch images
current = original;                     % Recover original image yet again
subplot(1,2,1), imshow(current)         % Draw current image
subplot(1,2,2), imshow(alternate)       % Draw alternate image
```

By evaluating the function

$$p = \text{analyse\_binary\_texture}(15, 3)$$

the coefficients (contained in the output vector p) of the following third order polynomial were obtained

$$y = 0.0004x^3 - 0.0118x^2 + 0.0361x + 0.9705.$$

A graph was then plotted (Fig. 38.9). To estimate the creasing factor, we might conveniently solve this equation, either numerically or by inspecting the graph, to find x (i.e., structuring element size) such that $y(x) = 0.5$.

38.3 Concluding Remarks

Naturally occurring texture cannot be defined precisely in mathematical terms. In any project involving texture analysis, human judgement must be the final arbiter, so the idea that there exists a unique *optimal* solution is non-sensical. For this reason, it is necessary to resort to experimental methods for designing heuristics for recognising textures. In this chapter, we have presented results for several ideas, in order to demonstrate that many *reasonable* solutions exist. While it is easy to suggest new texture filtering methods, it is not obvious, which if any of the six heuristics illustrated for Stage 1 could for the basis for a satisfactory solution. Evaluating them in combination with an appropriate Stage 2 method requires a considerable amount of further experimental work. This is problem specific and no amount of theoretical analysis can replace it! Several additional points should be noted:

- Low-angle (grazing) illumination was used to emphasise the texture of plain white fabric, creased simply by hand. In practice, a carefully controlled creasing protocol should be used. Printed fabric and material that has a patterned weave have not been considered and would almost certainly complicate matters.
- Several methods for Stages 1 and 2 are presented without making recommendations. To so would prejudge the experimental design needed for Stage 3. The large number of possible combinations of methods for Stages 1 and 2 makes it imperative that an appropriate set of computational tools is available.
- A primary aim of this chapter is to demonstrate that it is easy to suggest a multitude of "reasonable" solutions but these can only be judged subjectively. By using an interactive image processing system, like QT, a vision engineer can test and evaluate ideas quickly and easily. Many of these may simply be guesses, inspired by prior experience. The strength of systems like QT lies in the ease with which new ideas can be either dismissed as wild fantasy, or accepted as valuable new knowledge. An interactive image processor is well suited to "fine tuning" Stage 1 procedures, by adjusting their control parameters and seeing the results displayed almost immediately.
- The QT command *esr* facilitates the batch processing of a large number of images. So,

  $$[A, B, C] = esr(X, Y, Z)$$

  performs the operation X on all JPEG/TIFF image files in folder Y and leaves the processed images in folder Z. (Defaults are defined for Y and Z.) Any non-image files in folder Y are ignored. The output variable C is an array of A rows, each containing B measurement values.

39 Environmental, Social and Ethical Issues

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_39,

39.1 Introduction

J. H. Lemelson anticipated the subject now known as Machine Vision as early as 1954 [1]. (He filed 76 relevant patents but their lien on major Machine Vision manufacturers was released by a land-mark ruling in the U.S. Court of Appeals for the Federal Circuit in 2005. The judgement was made on legal, not technical, grounds.) However, it was not until the 1970s that serious work began. The 1980s began optimistically but growth slowed later in the decade, triggered by a lack of trust in the technology among US automobile manufacturers. The 1990s witnessed a great acceleration in technical progress and the rate of installation of new systems. Growth continued erratically in the first decade of the new century; world-wide sales of Machine Vision hardware are expected to reach $2,100 million by 2012 [2]. Major market sectors at the time of writing include automobile, aerospace, electronics, glass and plastic containers, printing and food. Non-industrial applications using similar technology include reading automobile number plates, recognition of faces, eyes (iris) and fingerprints. The latter enjoy the widest public awareness, being mentioned occasionally in televison news bulletins and popular drama productions.

It is an auspicious time to take stock of the situation and to raise awareness of both the dangers and potential benefits of Machine Vision technology. It has been said that the purpose of history is to avoid making the same mistakes again. What then are the lessons that we can learn from this relatively short period of development of our subject? The main one is that the expectations and hyperbole of the mid-1980s lead to a rapid growth of interest and the subsequent demise of a large number of companies involved in developing and selling Machine Vision products. The situation was very serious in Europe but in the United States the effect was catastrophic. Even now, we cannot build a Machine Vision system that can work properly in uncontrolled lighting, as some vendors have allowed their customers to believe. However, the market has recovered. The events of the mid-1980s should have been anticipated, since exactly the same effect had happened in the early 1970s to Pattern Recognition research, which the author witnessed at first hand. Many younger researchers in Pattern Recognition are unaware of the real reasons for the decline in research funding and interest then. Contrary to popular belief it was not the discovery of the limitations of multi-layer perceptrons but the realisation that the subject had been oversold. The technical achievements in both Machine Vision, in the 1980s, and Pattern recognition a decade and a half earlier, were judged by funding agencies as being not commensurate with the investment of time and money that had been made. It is important to understand why this happens. It is, in the author's opinion, an almost certain consequence of the process of obtaining research funds. Even young and innately enthusiastic researchers are encouraged to "oversell" their particular branch of technology, in order to convince the guardians of the research funds that their work should be sponsored. The author has observed that many people in the latter category expect "good" results in a far shorter period of time than is realistic. It is imperative that older, more experienced workers, who witnessed the events of earlier years, should tell their younger colleagues what really happened. As we shall see, "overselling" is one of the most pernicious errors to have afflicted Machine Vision and Pattern Recognition research. Older colleagues will recall that this happened to Cybernetics as well. It is interesting to note that subject labels, such as "Pattern Recognition" and "Cybernetics" have a period of popularity and then fade, to be replaced later by other "more modern" terms. The name "Automated Visual Inspection" has been replaced to a large extent by the more general term "Machine Vision", but this is due in part to the broadening scope of the subject, which now includes inspection, grading, sorting, counting, measuring, process monitoring, locating, and guiding robots.

An industrial Machine Vision system is usually purchased with one or more of the following claims by the vendor in mind:

(a) It will reduce product cost
(b) It will improve product reliability and/or safety
(c) It will improve process efficiency
(d) It will improve operator safety

The author suspects that (a) and (b) usually influence the decision to buy (or not buy) a Machine Vision system most heavily. Their justification for this is based on personal observation and anecdotal evidence received from his colleagues.

Appendix C presents a set of points in the form of "proverbs", intended to guide designers of Machine Vision systems. In the same manner, we shall summarise our main conclusions here in the form of "bullet points". The major topics to be considered in this chapter are:

(a) The impact of Machine Visions Systems on the environment.
(b) The implications of Machine Vision for product and factory safety, the health and well-being of employees.
(c) The importance of intellectual integrity in a field requiring a careful balance of advanced ideas and technologies.
(d) Commercial and managerial integrity.
(e) The impact of Machine Visions technology on employment prospects, particularly for people with low skill levels.

There are, of course, many general ethical, legal and moral issues which relate to all aspects of selling and procuring high-technology equipment. In this chapter, we shall concentrate upon those that refer specifically to Machine Vision.

39.2 Environment

Machine Vision systems are not particularly demanding in the level of resources they require. Of course, they are never likely to form a very high volume product. They include electronic circuits but, apart from that, do not require the use of particularly unpleasant materials in their construction. The real impact of Machine Vision systems on the environment is likely to be very positive. Certainly, the potential exists for this to happen. There several ways in which Machine Vision systems could provide very beneficial effects upon the environment.

- Machine Vision can reduce scrap by sorting "good" and "defective" items

  In 1979, the author was involved in a project which attempted to reduce scrap by sorting a "polluted" batch of fasteners into "good" ones, which could be re-used, and a much smaller number of "defective" items, which were truly worthless, except as scrap metal. At that time, the company involved estimated that, at a single factory, they were discarding about £700–£800 of *good* fasteners in polluted batches, *every week*. (This figure should be increased by multiplying by a factor of about ten, to estimate the value in present-day US dollars.) The implication is that, by sorting the polluted batches, the company would be able to meet the same production levels with a reduced demand on both raw materials and energy.

- Machine Vision can reduce the need for high quality raw materials

  On a more general level, we note that it should be possible to employ Machine Vision systems more effectively than at present by reducing, or at least limiting the demand for high-quality raw materials. For example, vegetables and fruit might be sorted/cut under the control of a vision system. This should allow a food-processing system to make good use of raw materials which have localised defects, such as bird pecks, insect holes, bruising, etc. Automatic cutting of leather (for shoes, bags, car seats, etc.) has been studied using an intelligent vision system to identify point and small-area defects. Similarly, more efficient use of timber might be possible using a cutter guided by a vision system. Since a vision system is potentially able to increase the efficiency of sorting, grading, packing, identifying and locating defects, on a wide range of materials and products, there is the potential for a significant benefit to the environment.

- Machine Vision can sort waste for recycling

  Automatic sorting of waste, prior to recycling, is one direct way in which Machine Vision could benefit the environment. There is considerable scope in this general area, particularly if we know what to expect the vision system to see. Until the 1970s, it was common practice to recycle milk, beer and soft-drink bottles, etc., by returning them to the supplier for washing and refilling. (In the past, many a young person has raised a handsome sum of money by returning empty bottles and collecting the deposit.) This practice has, of course, fallen out of favour. This is due, in part, to the difficulty of guaranteeing the quality of the recycled material. Vision systems should be able to help, for example, by sorting small spent batteries from consumer electronics.

- Packaging could be coded to aid identification of scrap by a vision system

  The task of sorting/verifying that the correct type of recycled container is present during refilling would probably be made much easier if scrap-identification codes were to be placed on all such products. Consider for example, the potential benefits that might be obtained if the major suppliers of soft drinks were to use vision systems to assist in the re-use of their bottles and cans. An effective Machine Vision system is likely to be necessary if we are ever to achieve efficient sorting of bottles rather than regarding them merely as cullet. It is folly to throw good glass-ware into a bottle-bank, since the energy cost of reheating cullet is the same as the cost of heating the raw materials used in making glass. (Heating is a major component in the cost of making a bottle.) By "rescuing" certain popular types of bottle, before they are smashed in the bottle-bank the energy savings could be significant. There are certain types of container for which recycling would not be safe but for many non-consumable products the dangers would be minimal. A vision system could, for example, inspect returned bottles of toiletries, after washing, to check for residual contaminants and structural integrity.

- Autonomous machines could clear litter in factories and streets

  We are all aware of the growing problem of litter in our streets and other public. Research into autonomous vehicles is progressing apace. The author suggests that developing self-guiding machines that hunt and gather litter would be an ideal application of Machine Vision. Of course, such a system would need to use vision to search for paper, plastics, cans, bottles, etc., Some common types of litter, e.g., soft drinks cans, fast-food wrapping could perhaps be made more be clearly identifiable by making it fluorescent. Industry needs similar systems that are able to perform tasks such as cleaning toxic, carcinogenic, biologically active and radio-active waste from factory sites and plant. An autonomous vacuum cleaner, guided by vision, would be very useful and would readily find application in a wide variety of industrial applications.

39.3 Product and Factory Safety

In many hazardous, or simply unpleasant, situations in which people are currently employed, there exists the possibility of using a machine to improve safety and/or avoid health hazards.

- Vision systems can protect workers from dangerous machines.

  Vision systems can be used to provide a "safety curtain" to protect workers who venture too near dangerous machines with fast moving parts, such as robots, power presses, stamping machines, metal-cutting machines, etc.
- Cost is no longer a barrier to installing a vision system to improve safety.

  Vision systems are able to perform well in a variety of relatively undemanding tasks that hitherto would have been regarded as insufficiently important to justify their use. For example, simple and inexpensive vision systems might be used to perform the following:
  - Monitor stacking of cartons
  - Monitor leaks, spills, etc.
  - Ensure machine safety guards are in place
  - Make sure that oven doors are closed
  - Check the floor for spills, obstructions in gangways, etc.
  - Warn workers of approaching fork-lift trucks, etc.
  - Look for oil/chemical leaks
  - Monitor coolant flow in metal cutting machines

  It should be remembered that it is possible to buy quite versatile, low-cost, stand-alone vision units ("smart cameras") and more sophisticated multi-camera systems. In both cases, the *incremental cost* of adding an extra camera can be quite small. The additional hardware needed comprises only low-cost items such as: lamps, fixing brackets, cables, lens, image sensor and its power supply unit. The incremental cost of the software is also small. (Designing and testing a new algorithm with a system such as QT or NeatVision usually takes just a few hours.)
- A person should not do hazardous work if a visually guided machine can do it.

  Given appropriate protection for the equipment and regular maintenance, a Machine Vision system can often reduce, or even eliminate, the need for people to work in conditions that expose them to hazards such as:
  - Very hot air
  - High levels of radiation: laser, strobed lighting, infra-red , ultra-violet, x-ray, and gamma radiation
  - Dust, fumes, smoke
  - Chemicals. These might be in liquid, aerosol or gaseous form. The hazard may be direct, caused by handling dangerous materials, or simply present as a pollutant in the broader working environment
  - Audible noise, ultra-sound and infra-sound

- Vibration
- Heavy or repeated lifting/manipulation

  Robots are already in use for cleaning and repair work inside nuclear reactors. Robots are also used to handle suspected bombs. A robot for ladling molten metal was developed some years ago but did not then use visual guidance. A robot has been developed for shearing sheep. When a human being performs this task, he adopts a poor posture and is therefore susceptible to back injury.

- Operator well-being should be a primary factor when commissioning a vision system.

  An example of an *indirect* benefit to operator well-being was encountered by the author some years ago, in a factory where forged ferrous automobile components were being made. At that time, the forged parts were being inspected for cracks using a magnetic-particle indicator method, which required a person to sit in a very small darkened cubicle, illuminated only by an ultra-violet lamp. The working conditions for the inspector were very unpleasant: the operator was in a tiny, dark depressing enclosure, exposed to the kerosene-based magnetic-fluorescent "ink" which is used to highlight the cracks. This environment is also very noisy. The prime motivation for the project, to build a vision system to inspect these forged components, was partly economic and partly motivated by the need to ensure higher product reliability. The benefit to the factory workers was viewed as being incidental but welcome.

- Legislators, employers and labour unions should all be made aware of the potential advantages of machine vision in avoiding exposing workers to unhealthy/unpleasant working conditions.

  The concept of "reasonable means" to define the limits of what is required of an employer in protecting his workers is enshrined in law. Engineers need to educate legislators, employers and labour unions that this includes the use of vision systems for inspection, process monitoring and control. Of course, the legal expectation of what is "reasonable" in terms of factory safety changes as technology advances. The education of thes groups is therefore ongoing.

- Legislators, employers and labour unions should all be made aware of the possibility of using machine vision to improve product safety.

  Again, the limit of what is regarded as "taking reasonable care" should be redefined in the light of recent advances in Machine Vision. It is only a few years since one of the author encountered several dangerous defects in everyday items of glassware. All of these could have been detected in the factory by a vision system. Techniques for inspecting bottles, jars and tumblers had been published 10 years prior to that. The author does not accept that it is a valid defence to say that no machine was available commercially, since the techniques for detecting these particular types of defects had been validated many years before.

39.4 Intellectual Integrity

It is clear to anyone who has worked in Machine Vision system development or research that the devices are complex, because they combine many different types of technology, and the algorithmic/heuristic procedures they use rely upon subtleties that are not obvious to the casual observer. It is especially important to approach working in Machine Vision with an appreciation that the real difficulties lie in the systems integration.

- Generalisation without proper experimental evidence is always dangerous

  Unfortunately, it is a common experience to find that a vision systems engineer, working for a manager who does not fully appreciate the technical nuances, experiences a lot of pressure to compromise on design principles. For example, a non-technical manager might well fail to appreciate that a vision system designed to inspect *clear* glass bottles might not work properly on clear glass jars, green or brown glass bottles, or on clear plastic bottles. (The side walls of the last mentioned are usually thinner and the refractive indices of glass and plastic are different.)
- What may seem like similar tasks to us may be quite different to a machine

  For sound commercial reasons, some vision companies focus on niche markets. For example, a company might produce a machine for inspecting bare printed circuit boards. It is all too easy to assume that the same machine can be modified slightly to inspect populated PCBs. In fact, it may be much easier to adapt that machine to inspect some completely different product, such as cloth, CDs, printing, etc. A would-be customer should not therefore expect the supplier of a machine vision system, capable of tackling one task, to be able to develop reliable and robust techniques for use in the same general market area.
- Do not assume that you are an expert on vision, simply because you can see

  One of the most common errors which causes significant problems is that a very experienced practitioner in Machine Vision is told by non-experts how to solve a problem. (The author has suffered this many times!) However confident or well educated the speaker, when a non-expert says "Of course, I can see at a glance how to solve this problem", a vision engineer is about to be told *erroneously* how to do his job. This mistake is particularly awkward if the speaker is the supervisor of the hapless vision engineer. Undue pressure from a non-expert manager can result in two undesirable events:

  1. The engineer is expected to achieve things that are physically impossible, or simply very difficult to achieve and will render the final solution unreliable.
  2. The technology is oversold to the would-be customer, who is not sufficiently well versed in the technology to be discriminating enough to sense its dangers.

  Customers do this as well! Indeed, one wonders sometimes why they bother to consult the vision company at all. The problem is caused by one commonly held fallacy: that introspection is able to reveal how the inner mind works. Quite simply it does not and no self-respecting vision engineer should ever listen to such foolish ideas.
- It may be impossible to collect a representative set of test samples

  The reason is that a manufacturing system is expensive to build and operate, so that it is only ever allowed to make "good" products for any significant length of time. ➔ *Figure 39.1* As soon as a cluster of defective products is detected, the operating parameters of the manufacturing system are adjusted, so that it quickly reverts to making "good" products again. The result is that very large numbers of "good" products are made but very few "defective" ones. Hence, it is possible that insufficient numbers of the latter group will be available to satisfy the requirements of traditional Neural Network training methods, which need large numbers of samples from two (or more) classes. Learning procedures that are able to use a single class of patterns are required in this situation. Even then, there are difficulties in evaluating the learning process and demonstrating to the customer that it has been successful. It is simple intellectual dishonesty to disregard the difficulties just described and to persist in using techniques which are manifestly unsuitable for the

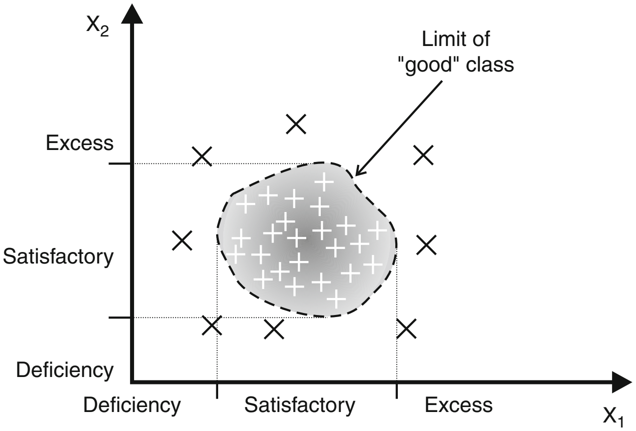


Fig. 39.1

Collecting enough representative data can be difficult. It is possible to collect very large numbers of "good" widgets (*vertical crosses*). However, manufacturers do not like making "faulty" widgets (*diagonal crosses*), because they cannot sell them. Except in the simplest cases, it is impractical to collect a representative sample of all types of "faulty" widget. When there are two parameters measuring the "quality" of an object, we need at least 8 (= $2 \times 2 + 2^2$) samples of the "faulty" class to defines the surface separating them from the "good" class. (This is the absolute minimum; in practice many times this number is required.) When there are N parameters, this number rises exponentially, as ($2N + 2^N$). Notice that it is probably impossible to adjust the manufacturing plant to make all types of defect

particular task in hand. A similar problem arises when trying to evaluate rule-based decision-making processes.

- Do not be economical with test samples

  It is very easy to misrepresent the truth about the performance of a vision system by presenting it with a few carefully chosen examples, during a demonstration. Machine Vision is particularly prone to this, since the factors affecting system performance are very subtle. For example, an inspection algorithm may work very well if the products are clean and dry. However, any appreciable amount of oil, dirt or moisture on them may confuse a vision system so badly that the system will be useless in the factory.

- Machines do not generalise as well as humans do

  A particular colour on a printed package may change in quite a subtle way, when the ink is supplied by a different company, or the carton is interchanged with one of a slightly different shade. A colour recognition system will not necessarily be able to cope with such changes, even though they are quite insignificant and unimportant to a human being.

- Exaggerated claims discredit the whole industry

  It is widely accepted that hyperbole is an integral part of selling. However, there is an important distinction to be made between extolling the virtues of one machine relative to another and knowingly making claims that are factually incorrect. As we have already emphasised, Machine Vision is characterised by having a large number of very subtle features that can seriously affect the performance of a given system. This makes the subject particularly sensitive to unrealistic claims. Moreover, customers do not always have the ability to discriminate between hyperbole and untruth. Eventually, of course, the untruths and exaggerated claims bring discredit to reputable manufacturers and equipment suppliers. We cannot emphasise enough that danger lurks in the subtle nuances. Every systems engineer knows this. The problem for vision engineers is, perhaps, as acute as it is in any other technology.

- For a vision system, appearance is everything

  The distinction between *composition* (e.g., made of brass) and *finish* (e.g., painted bright blue) is one classic point in Machine Vision circles. It is tempting to say that objects (of a given shape) made from any metal may be inspected by a certain system, However this is unlikely to be true unless, they have all been coated in the same way. Dull lead, polished brass or copper, oxidised steel or galvanised tinplate are so different that it is very unlikely that a single machine vision system will be able to cope with them all -unless, of course, they have all been painted bright blue!

- We must be seen to be up to date with technology!

  There is a seemingly natural desire, among some people, to employ automation, even in those situations where a simpler, more reliable technology might work just as well, possibly even better *and* be cheaper. Straightforward solutions exist for an enormous range of industrial inspection and measurement tasks. It is good practice, therefore, to question whether vision is really appropriate *in every application*. Employing an overcomplicated and inordinately expensive technology, when a simpler one will suffice, is simply poor engineering practice. This must be balanced against the enormous versatility of vision-based sensors.

- The machine may not be able to see a feature even though a person can

  Place a freshly laundered white handkerchief on a piece of white paper. Can you see it? In most cases a person can see it quite easily but it is doubtful whether a vision system will be able to do so, accurately and reliably. Human beings are extremely clever at discerning very fine differences in shade, colour and texture. Present-day vision systems are not nearly so adept. This is partly due to the fact that they use sensors that cannot match the performance of the eye, and partly because existing vision algorithms are very clumsy, compared to human beings. Since a machine may not be able to discern fine differences, we have to begin our design exercise with an open mind about what a machine will be able to see. Neither the vision engineer's manager, nor the customer, should try to exercise undue pressure to achieve what they erroneously "know to be possible". Expressed in simple terms, the maxim is "Trust the vision system, not your own eyes".

- All results should be reproducible but is the data?

  One well known supplier of vision systems produces a wall chart containing the following maxim:

  ► The results of all system performance demonstrations should be reproducible, statistically characterised as to be repeatable and be statistically verifiable.

  When inspecting "hard" engineering products, such as spark plugs, electrical switch contacts, carburetor components, etc., the advice given above is undoubtedly very well worth following. In practice, however, the situation is often far too complicated for this to be useful. To understand why, consider a bakery. Loaves and cakes cannot be preserved indefinitely, even by freezing, so any results we obtain with them are not reproducible. We cannot make replicas of cakes that are totally realistic. (Replicas of food items are made for the film and television industries and, for this purpose, they are perfectly adequate. However, they do not fool a human being upon close inspection.) The concept of "statistically verifiable" is not valid either, particularly when applied to a complex manufacturing plant, such as a bakery, where there are numerous independent variables. Even if all of the plant parameters and measurable material quantities (e.g., oven temperature, dough moisture content, etc.), could be measured, there remains an enormous number of unmeasurable and uncontrollable variables (e.g., bran and gluten content of the flour, sugar grain size, etc.), which render the concept of statistical stability invalid. Any operation such as baking, which relies upon natural materials is prone to variations of material characteristics which makes this piece of "good advice" unworkable.

- Vested interests often influence judgement

  A well-known and well respected journal once refused to publish an article written by the author. One of the referees, who was described by the editor of the journal as a "venerable and very well-respected figure", referred to certain results quoted in the article as "having been faked". (Another journal subsequently accepted that article. Similar work has also been published several times elsewhere. To prove his professional integrity in this matter, the author even offered to stage a public demonstration of his work. That offer was declined!) Why should an eminent person with a fine reputation make such an accusation? One possible explanation is that the referee had a certain predisposition in the matter and did not wish his/her academic standing to be diminished. Perhaps the problem was caused simply by an error of judgment but the fact remains that an idea, which has since gained respectability by independent refereeing, was nearly denied to the waiting world! In a subject with as many subtleties as Machine Vision possesses, there is no room for blinkered thinking. There exists a set of guidelines outlining the rôle of editors, referees and authors, who all have a responsibility to each other and to the wider technical community. There is much anecdotal evidence about prejudice within funding review committees and much could be written on this vexed question. However, these broader issues are not our main concern here, where are concentrating on question of integrity relating specifically to Machine Vision.

39.5 Commercial and Managerial Integrity

It is often claimed that Machine Vision is a very flexible technology. While there is no doubt that this is technically true, there is a practical difficulty which has, so far, made this an empty claim: there are simply not enough properly trained vision engineers to realise the full potential.

- We must not claim versatility for Machine Vision when there is not enough engineering effort available to realise its technical potential.

  There is no truly general-purpose industrial vision system on the market today. To explain the importance of this fact, let us consider just one well-known company, which has over 50,000 products. The company operates a policy for innovation and invention which ensures that, at least, 25% of sales are based on products that are less than 5 years old. Simple analysis shows that ten new products are being introduced by that single company each working day. We simply do not have sufficient skilled labour to be able to design vision systems to inspect or control the manufacture of all of them. Of course, this situation is repeated many times over, in every industrialised country. If Machine Vision is truly a versatile technology, we ought to be able to build equipment that can inspect a significant proportion of new products. Due to man-power limitations, we cannot do this for more than a tiny proportion of all potential applications. Hence the claim that Machine Vision is a versatile technology is an empty one.

The purpose of this book is, of course, to make more people aware of the potential and subtleties of Machine Vision technology. In this way, the contributors to this book hope to train more vision engineers, so that the bottleneck can be relieved and more successful applications achieved.

- Do not use vision when a simpler technology will suffice

  Since it is potentially very versatile, it is all too easy to rely upon Machine Vision technology, when a simpler one will suffice. In its usual modern English translation, Occam's Razor states

  ► It is vain to do with more what can be done with less.

  Many alternatives exist to Machine Vision, which for many people is beguiling, simply because it is perceived as an advanced technology. Less sophisticated technologies are often more appropriate on both technical and commercial grounds. In this situation, it is regrettable that many vision systems suppliers still persist in trying to sell their products. Unfortunately, this brings discredit on the whole vision industry.

- A vision engineer should be present when demonstrating the system.

  As we indicated earlier, Machine Vision contains many subtleties that are not obvious to a person with normal sight but who lacks comprehensive training in the technology. It is all too easy to provide a misleading answer to a seemingly straightforward question from a customer. Many questions relate to the ability of the vision system to generalise as the following question shows:

  ► You have successfully shown that the system can inspect brown widgets. Can it inspect green ones as well?

  It is absolutely essential that the answers to such questions are only given after appropriate and detailed experimentation. Offering a bland affirmative answer, without proper experimental evidence is totally unjustified and constitutes improper professional behaviour. Experienced vision engineers would not dare to answer without experimental evidence!

- False impressions may be given on purpose, accidentally or by neglect.

  An example of this was evident some years ago in a series of presentations by a well known vision equipment manufacturer, which persistently showed unprotected cameras hanging from a gantry in a steel-works. Since no verbal reference was made to this, there was a subtle implication that camera protection is unimportant and can safely be ignored. Every professional engineer sees the misleading impressions that glossy demonstrations at exhibitions and sales literature can bring. References to terms such as *real time*, *user friendly*, *industry standard*, etc., are technically meaningless, since they have no formal definition. The first of these, for example, implies quite a different time scale in relation to an automated baking plant (minutes) and power press (milliseconds). "Ambient light" is another ill-defined and often confusing term. Operating a vision system within a factory is quite a different matter from doing so in an open field, where the variations in light level are very much greater.

- Prove the system before selling it

  We have repeatedly emphasised that it is impossible to design a vision system while sitting at a desk, or drawing board. For the same reason, it is impossible to build a general-purpose system that is not programmed or adapted in some way to the specific task in hand. It is therefore very unwise to purchase a system that is claimed to be "general purpose", without testing it on the specific application.
- Training and post-sales support both form an integral part of supplying a vision system

  A proper appreciation of the factors that influence a vision system is essential, if the maximum benefit is to be obtained. Prior to installation of the system, the user should be alerted to the importance of such issues as following a regular maintenance procedure, surface finish of the product, stray light entering the camera, and local atmospheric pollution, due to fumes, spray, smoke and dust. After commissioning the system, there will often be occasions when the system is not working properly because some such lesson has been forgotten. The equipment supplier should anticipate this and budget accordingly.
- Avoid built-in dependency upon one supplier

  Contrary to what has just been said about the need for providing good after-sales support, there is a danger that the customer will become dependent upon that one company and therefore be unable to make the mental switch necessary to refer to a new company, or a consultant. Customer dependence on one supplier has obvious dangers for the long-term development of the subject.
- The terminology and concepts of Machine Vision are not unique to any one company

  It does a company no credit to obscure the operation of a machine by using words and phrases that are not well understood in the academic community. The supplier of Machine Vision equipment should avoid the use of jargon of his own invention; standard, well-understood engineering and scientific terms should be used, so that an independent consultant can evaluate the machine.
- Finance is not the only justification for installing a vision system

  All too often, economic considerations are assumed to form the only significant factor influencing the decision to purchase a vision system. Product and factory safety should be given a comparable weighting to that of finance. A professional person must not be "one dimensional" in his thinking. The equipment supplier should emphasise all of the potential benefits of installing a Machine Vision system. The customer should attempt to quantify all of the advantages, bearing in mind that finance is only one of them.
- Design new products with inspection in mind

  The benefits of retrofitting a vision system to an existing manufacturing plant may not be realized. On the other hand, by designing new products with automated visual inspection in mind may be very beneficial. Low-cost fluorescent materials can easily be added to a variety of plastic goods, packaging materials, food-stuffs, toiletries, pharmaceutical products, paints, inks and lacquers, to improve image contrast. This will often make visual inspection easier and more reliable than it might otherwise be. Injection mouldings can often be provided with a tiny protuberance, or an array of dots to enhance vision system performance to identify, locate, or orientate the product. The cost of such modifications may be very small but the effects, in terms of improved system performance may be enormous. Modifying the product slightly may make automated visual inspection both cheaper and more reliable.

39.6 Employment

- Do not attempt to automate jobs done by people with learning difficulties

  A few years ago, the author visited a bakery on a very hot summer day. One employee was working very close to one of the ovens, which, of course, made the poor man even hotter. His task was one which could easily been automated using a visually guided robot. However, he was described by the line manager as being "one of their best workers, who loves his job", even though he has learning difficulties. This was a man who, despite the working conditions, greatly valued his work. Automating that job would have made that man unemployed and possibly even unemployable. Vision gives us the opportunity to automate a lot of low-skill jobs but we should consider seriously the social and humanitarian consequences of doing that.
- Be careful when using vision to monitor staff activity

  A vision system is available for monitoring the movements of people in a shop or factory. One of its main uses is to observe what causes people to stop and look at the products in a supermarket. The appeal of supposedly eye-catching displays can be assessed. The same machine could be used in a factory for watching people as they are working (or supposed to be). Many people will be happy to accept this use of technology but others will, no doubt, liken its use to the excesses of Orwell's nightmare world of "1984". Watching people talking, when their employer thinks they should be working may eventually be judged to be a legitimate use of Machine Vision technology but we do need to debate such issues before it is too late to halt them.
- Automation causes unemployment - but only a little!

  Here, at last, is the one issue which nearly every reader will expect to see in this chapter. The author, like many other engineers concerned with automation and information technology has been taken to task on this topic, by many well-meaning individuals who are concerned about and, in many cases, fear the spread of computer technology. Many times they have been forced to defend their rôle in this development. We are all aware of the pernicious problem of unemployment in Western society. Most people seem to believe that automation is a major factor in creating unemployment. In fact, this is not so, as [3] suggests. This came as a complete surprise to the author, who began writing this essay with a different attitude from the one with which he completed it. While we must not be complacent about the rôle of Machine Vision technology in causing unemployment, this surprising fact does give some encouragement for us to develop our technology further, without undue fear of causing major social problems.

References

1. Lemelson JH (2009) URL http://en.wikipedia.org/wiki/Jerome_H._Lemelson. Accessed 2 October 2009
2. Machine Vision Market (2009) Weathering the storm, Photonics. http://www.photonics.com/Article.aspx?AID=36908. Accessed 22 February 2010. *Machine Vision Market Set to Grow*, URL http://www.imveurope.com/news/news_story.php?news_id=192. Accessed 2 October 2009
3. Causes of Unemployment, Economy Watch (1995) URL: http://www.economywatch.com/unemployment/causes.html. Accessed 22 February 2010. Sasha Nemecek, "*Employment Blues: Nothing to do with Being Green*", Scientific American, June 1995, p 25

40 Lighting-Viewing Methods

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_40,

Guide: This chapter provides qualitative descriptions of a variety of lighting and viewing methods (LVMs). These are presented in alphabetical order, since there is no natural structure for such a list (➲ *Table 40.1*). There are usually five elements within an image acquisition sub-system: light source, illumination optics, image-enhancement optics, main lens and sensor. The positions of these relative to each other and the object being observed must be carefully selected, to obtain a digital image that is easy to process electronically. The physical layout of the optical sub-system and the choice of appropriate components are of equal importance. ➲ Chapter 7 describes the light sources available, while optical components are discussed in ➲ Chaps. 5 and ➲ 6. Scanners and cameras and are described in ➲ Chaps. 9 and ➲ 10 respectively. ➲ Chapter 8 explains some of the concepts necessary to appreciate the methods described here. Of particular note is that of an illumination diagram. This explains, in graphical form, the shape and position and of the light source in relation to the widget and camera. Notice that the diagrams included in this chapter are not drawn to scale; they merely suggest a design for the lighting-viewing sub-system in broad conceptual terms. A multitude of combinations of the methods described below is possible.

Throughout this chapter, the word "*widget*" is used to denote the object that is to be examined.

When designing the image acquisition sub-system, a vision engineer should normally follows the following steps:

(i) Look at the widget
  (a) In a darkened room, move a "point" light source round the widget. Rotate the widget as well. Watch the way that its appearance changes
  (b) Repeat (a) using a broad light source (e.g., stand in front of a window). Ensure that no light falls on the widget from any other direction
(ii) Bearing in mind the observations in step (1), consult the LVM Catalogue. Choose one or a few possible lighting-viewing methods
(iii) Build a prototype lighting-viewing sub-system based on the idea(s) suggested by the catalogue
(iv) Acquire some sample images
(v) Process these images using an interactive image processing system, such as QT (➲ Chaps. 21 and ➲ 41) or Neat Vision (➲ Chap. 22)
(vi) Adjust the lighting-viewing sub-system and repeat steps (4) to (5), until satisfactory results are obtained
(vii) Design and build the target system based on the prototype

Step (1) is very simple, yet it is always in danger of being omitted. The phrase that summarises this design process is "*Experiment, look and observe*". Over a period of many years, involving numerous applications, the author has found this procedure to be very effective indeed. *Never attempt to design an image acquisition system without moving from the desk!*.

Table 40.1
Alphabetical list of lighting-viewing methods

Title	Method number
3D shape analysis by projecting Moiré patterns	1
All round view of a cylindrical object	2
Anamorphic optics	3
Arbitrary lighting pattern	4
Automatic gain control and auto-iris lens	5
Back illumination of shiny objects (array camera)	6
Back lighting of matt opaque objects (array camera)	7
Back lighting of matt opaque objects (line-scan camera)	8
Calculate the field of view – I	9
Calculate the field of view – II	10
Calibration: calculating spatial resolution	11
Circle of light projecting radially outwards	12
Circular features	13
Circularly polarising filter suppresses glinting	14
Coaxial diffuse illumination	15
Coherent flexible fibre optic bundle	16
Collapse image into line	17
Collecting natural light/sun light	18
Colour filtering	19
Combined back and front illumination	20
Combining several views	21
Conical mirrors view an annulus	22
Continuously moving web	23
Crack detection (ferrous and non-ferrous)	24
Crossed linear polarisers reduce glinting	25
Dark field illumination (array camera)	26
Dark field illumination (line-scan Camera)	27
Detect pimples, pits and changes of refractive index	28
Detecting particles in a swirling liquid	29
Diffuse coaxial illumination	30
Diffuse front illumination	31
Dual orthogonal-beam X-ray imaging	32
Dual-wavelength height measurement	33
Endoscope for inspecting holes	34
Estimating range during image capture	35
Examine the inside of a pipe	36
Fast electronic shutter	37
Fibre optic image conduit/converter	38

Table 40.1 (Continued)

Title	Method number
Fish–eye lens views inside hole	39
Flexible inspection cell/robot vision cell	40
Fluorescence and phosphorescence	41
Focussed annulus of light	42
Forced thermal emission	43
Generating light with arbitrary spectrum	44
Granular material (pellets) in free flight	45
Grazing illumination	46
Hand-held sensor for fixed-range close-up view	47
Hardness testing	48
Height analysis by projecting patterns	49
High resolution view of web	50
Highly variable ambient lighting	51
Homogenising illumination	52
Hypercentric (pericentric) lens	53
Illuminate top surface only	54
Illuminating a wide moving web	55
Illumination within a hollow cylinder	56
Injecting light into glassware	57
Inspecting a flat moving web using a laser scanner	58
Inspecting a rotating object	59
Internal analysis of an aerosol spray or dust cloud	60
Internal features on transparent objects	61
Laser bore scanner	62
Laser pattern projector	63
Locate object with zero contrast	64
Long focus optical system using mirrors	65
Low-definition range information	66
Magnify one image axis	67
Mapping by projecting multiple stripes	68
Mapping range with polychromatic light	69
Measuring range by projecting patterns	70
Measuring thickness of a thin film	71
Measuring thickness of a thin film	72
Micro-louvres suppress ambient light	73
Microscopic fourier analysis of transparent film	74
Motorised filter wheel	75
Multi-arm image conduit	76
Multi-camera specular and near specular illumination	77

Table 40.1 (Continued)

Title	Method number
Multi-camera structured lighting	78
Multiple views of a widget	79
Multi-spectral image sensing	80
Multi-view stereo imaging	81
Normal viewing of spherical surface	82
Objects on a conveyor	83
Observing bubbles in clear liquid	84
Omni-directional illumination	85
Optically tooled bowl feeder	86
Passive de-speckling of laser	87
Pipe or hollow cone of light	88
Polar-coordinate scan of annulus	89
Projected array of spots	90
Protecting the camera using a periscope	91
Protecting the camera with fibre optics	92
Protecting the camera	93
Range measurement using talbot fringes	94
Reading printing in photo-chromic ink	95
Real-time optical filtering (high-pass)	96
Ring of light projecting inwards	97
Rotate image using a dove prism	98
Silhouette from multiple shadows	99
Silhouette in restricted space	100
Size measurement in free fall	101
Split field viewing	102
Stabilised lighting	103
Steerable image sensor	104
Steering a laser beam in two directions	105
Stroboscopic illumination	106
Structured lighting	107
Switched-colour light source	108
Telecentric lens	109
Using mirrors to view front and back	110
Very large membrane mirror	111
Vibration analysis: speckle pattern interferometry	112
View annulus, ignoring central disc	113
View flat surface without glinting	114
View orthogonally polarised images	115
Viewing a bore	116

Table 40.1 (Continued)

Title	Method number
Viewing aerosol spray	117
Viewing inside an irregular transparent object	118
Viewing small objects (short focus lens attachment)	119
Viewing small objects/features	120
Viewing small objects	121
Viewing stress in a transparent sheet	122
Viewing through a small aperture	123
Visual sensor mounted on robot	124
Visualise changes in refractive index	125
Visualising heat distribution	126
Wireless image acquisition	127
X-ray imaging	128
Zooming by movement	129

40.1 Method 1: 3D Shape Analysis by Projecting Moiré Patterns

OBJECTIVE To obtain low-precision height information quickly about a smooth surface.
TYPICAL APPLICATIONS Measuring machined components, castings, forgings, mouldings, extrusions, etc.
EQUIPMENT Ruggedised projector, fitted with a fine grating. The camera views the object through a similar grating. The gratings may have patterns consisting of a set of parallel lines, arrays of spots, circles, or other curves.
CAMERA Array.
MOTION Static.
GEOMETRY The widget should not possess any step-like features, since they produce "broken" Moiré-pattern curves that are difficult to interpret (This requirement can be relaxed if verification, not measurement is to be performed).
MATERIALS The surface should be of nearly uniform reflectivity.
REMARKS Moiré patterns are created by overlaying one set of parallel lines/arcs, or a grid, onto another similar, but slightly different set. ➋ *Figure 40.1a* shows the optical layout, while ➋ *Fig. 40.1b–j* demonstrate the sensitivity of this method. Moiré patterns do not provide absolute measurements of surface height, since they do not indicate the direction of its slope. If the surface has a known shape, it may be advantageous to project a set of curves, instead of lines. If a set of projected lines/curves has a resolution similar to that of the camera's photo-detector array (taking the lens magnification into account), additional moiré stripes can be created (➋ *Fig. 40.1e–g*). Hence, care has to be taken, to avoid adding a secondary moiré pattern, due to the camera's finite resolution. Alternatively, it is possible to exploit this effect, by modifying the optical arrangement: we simply remove the grating near the camera. The projected line spacing should be equal to that of the camera's photo-detector array, which acts as the second sampling grid. Care must always be taken to avoid creating diffraction effects, due to the use of very narrow lines on the grating.
LAYOUT DIAGRAM AND EXAMPLES ➋ *Figure 40.1*

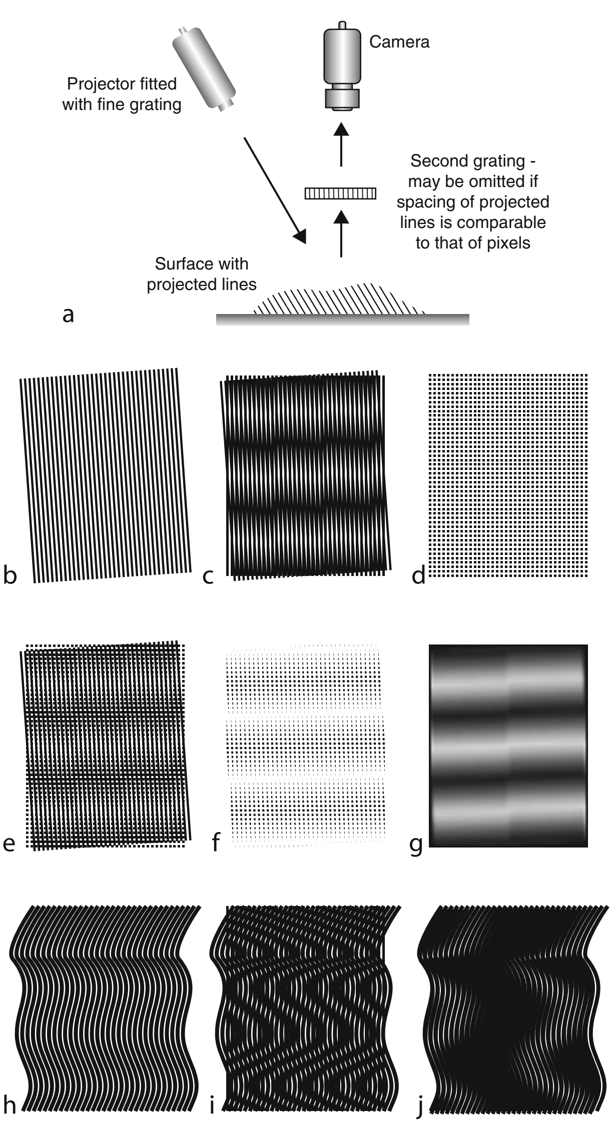

Camera
Projector fitted with fine grating
Second grating - may be omitted if spacing of projected lines is comparable to that of pixels
Surface with projected lines
a
b
c
d
e
f
g
h
i
j

Fig. 40.1

Moiré patterns. (a) Optical layout. (b) Parallel lines, inclined at 4°. (c) Superimposing the previous image onto an identical set of lines, inclined at 0°. (d) Grid, representing a video camera's photo-detector array. (e) Superimposition of image (b) and the same image rotated by a small angle. (f) Intersection of the binary images in (b) and (d). (The outputs from the camera's photo-detector array are proportional to the spot sizes here.) (g) Applying a severe blurring (low-pass) filter eliminates the fine detail. (h) Set of parallel arcs. (i) Moiré pattern, produced by the previous image and a set of vertical parallel lines. (j) Superimposing image (h) onto a similar set of arcs that have been rescaled slightly along the horizontal axis

40.2 Method 2: All Round View of a Cylindrical Object

OBJECTIVE To obtain an all-round view of cylindrical objects: 360° in the horizontal plane (The top and bottom of the cylinder are not viewed).
TYPICAL APPLICATIONS Inspecting corks, helical springs, various electrical and mechanical components.
EQUIPMENT Two ring lights.
CAMERA Three or more array cameras.
MOTION Static.
GEOMETRY Cylindrical.
MATERIALS The objects must not scratch, score or stain the feed tube.
REMARKS The widgets are fed and viewed while they are held in position within a glass tube. A wire cage might be used instead, although this might obstruct the view of the widget surface. Care must be taken to design the feed channel in such a way that the widgets do not jam. Three, four or more cameras can be used and would normally be spaced evenly around the widget. The feed mechanism can be modified to examine extruded products, such as confectionary bars and other food products), With the arrangement show in ➲ *Fig. 40.2*, lighting is uniform all the way round the widget but can vary in the vertical direction. By placing light sources above and below the viewing cameras, it is possible to minimise these variations.

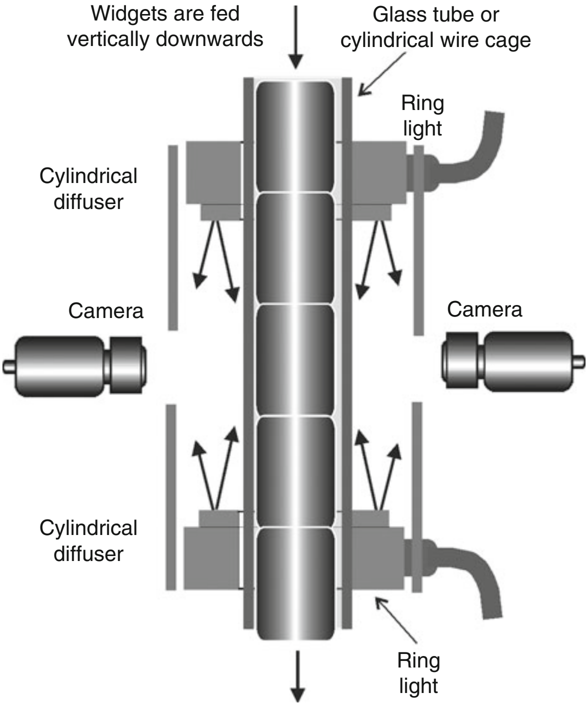


Fig. 40.2
Obtaining an all-round view of a cylindrical object

REFERENCE Santos JC, Aurelio J, Padilha AJ, Batchelor BG, Automatic grading of bottle corks. In: Proceedings of the 5th international conference on automated inspection and product control. IFS, Bedford

LAYOUT DIAGRAM ➔ *Figure 40.2*

40.3 Method 3: Anamorphic Optics

OBJECTIVE To obtain controlled geometric warping of an image, to simplify, or speed up, the image processing operations.

EQUIPMENT Mirror or lens, specially made to achieve the desired effect. Anamorphic mirrors may be concave, convex, or both. Surfaces are often non-spherical.

MAJOR TYPES AND THEIR USES

Mirror form	Function/application
Convex cone	(a) Examine inside surface of a pipe/tube
	(b) Inspect female screw threads
Convex sphere:	(a) Inspect inside surface of a pipe/tube
	(b) Increase the resolution in the central part of an image compared to its periphery
Concave ellipsoid	Normal view of part of a spherical surface. Camera with a wide-angle lens and widget are placed at the two foci
Cylindrical lens	(a) Adjust image aspect ratio
	(b) Pair provides parallelogram warping of image
Concave toroidal lens	Magnify annular region radially
Double wedge	Image splitter
Axicon (conical lens)	Remove central portion of field of view
Parabolic mirror (used with fish-eye lens)	Provides infinite depth of field
Fish-eye lens	Provides field of view in excess of 2π steradians

REMARKS Many of the lighting-viewing methods described elsewhere in this collection are based on anamorphic optics.

Anamorphic optics, producing image distortion, are abundant in our everyday lives, since polished curved surfaces, (e.g., car body, chrome kettle, bowl of a silver spoon, Christmas-tree bauble, reflective "silver" wrapping paper, etc.) produce warping. The "Hall of Mirrors" at an amusement fair creates comical distortions of the human body, using mirrors that are not quite flat. Reflections in large windows are similarly distorted. After the 1745 rebellion in Scotland, images showing the likeness of the Young Pretender (Bonnie Prince Charlie) were outlawed. To overcome this restriction, his supporters often kept paintings of him that could only be viewed properly by observing the reflection in a brightly polished metal cylinder. The human brain quickly learns how to correct the anamorphic warping produced by variable-focus spectacle lenses. Fish-eye lenses produce anamorphic distortion, while providing an all-round view. Using this principle, certain CCTV lenses, intended for surveillance applications, are deliberately made to enlarge the central portion of the scene, while providing low-resolution peripheral vision. *Cinemascope* lenses are used to “stretch” a nearly square image, on the film, into an elongated rectangular image, projected onto the cinema screen. Special-purpose “slant lenses” are made that can correct for oblique viewing of a plane surface (A trapezium is mapped into a square). Driving mirrors are often anamorphic, since they must provide a good all round view of the road.

Anamorphic optics can be reflective (concave or convex mirrors), or refractive. If they are carefully designed and manufactured, anamorphic optical systems can achieve some valuable effects for Machine Vision applications. For example, simple curves of known geometric form can be “straightened”, thereby making image analysis easier. A particularly useful function of anamorphic optics is to provide high resolution at the centre of the camera’s field of view with a lower resolution near the periphery.

Designing and manufacturing optical-quality anamorphic surfaces is not as difficult or expensive as might be imagined. Using CAD ray-tracing programs and CNC machining, optical quality non-planar, non-spherical surfaces can be made with relative ease. These can then be replicated cheaply in plastic, using standard moulding techniques. (Variable-focus spectacle lenses are made to order in this way.) Anamorphic mirrors can be made by evaporation coating of such surfaces, using aluminium, silver or gold. Low-grade anamorphic mirrors can be made using metal-coated Mylar film, reflective wrapping paper or *Silverlux*™, stretched over a suitable former.

REFERENCES

- *Silverlux*™ is a product of the 3 M Company
- Ray SF (1988) Applied photographic optics. Focal Press, London/Boston, p 267. ISBN 0-240-51226-X
- Batchelor BG, Hill DA, Hodgson DC (eds) (1985) Automated visual inspection. IFS, Bedford. ISBN 0-903608-68-5. Also published by North Holland, Amsterdam, 1985, ISBN 0-44487577-98, pp 96, 115
- Thomas DE (1980) Mirror images. *Scientific American*, December 1980, pp 158–171
- Stork D (1992) Anamorphic art photography: deliberate distortions that can easily be undone. Opt Photonics News 3(11):8–12
- Falk D (1986) Seeing the light: optics in nature, photography, color vision and holography. Wiley, New York
- OpTex DV anamorphic attachment (converts a 4:3 aspect ratio to 16:9), ZGC, Mountain Lakes. http://www.zgc.com. Accessed 2 Feb 2011

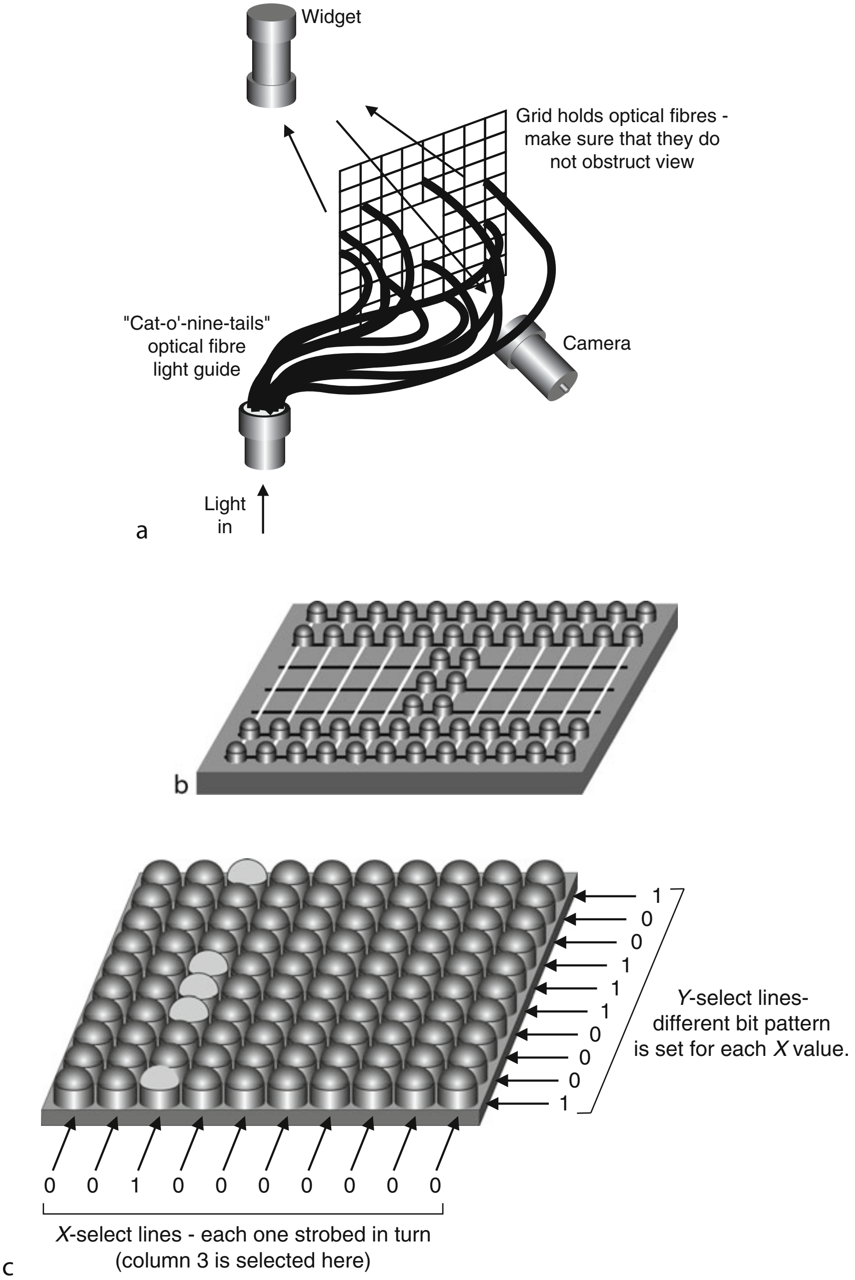
40.4 Method 4: Arbitrary Lighting Pattern

OBJECTIVE To generate a complex pattern of light that can be adjusted to suit individual applications.

TYPICAL APPLICATIONS

(a) Inspecting
- Complex objects: castings, mouldings, assemblies, etc. which may have multiple crevices and holes
- Glass-ware
- Curved shiny surfaces

(b) Verifying holograms (➋ *Fig. 40.3f*)


Widget
Grid holds optical fibres - make sure that they do not obstruct view
"Cat-o'-nine-tails" optical fibre light guide
Camera
Light in
a
b
1
0
0
1
1
1
0
0
0
1
Y-select lines- different bit pattern is set for each *X* value.
0 0 1 0 0 0 0 0 0 0
X-select lines - each one strobed in turn (column 3 is selected here)
c

◘ Fig. 40.3
(Continued)

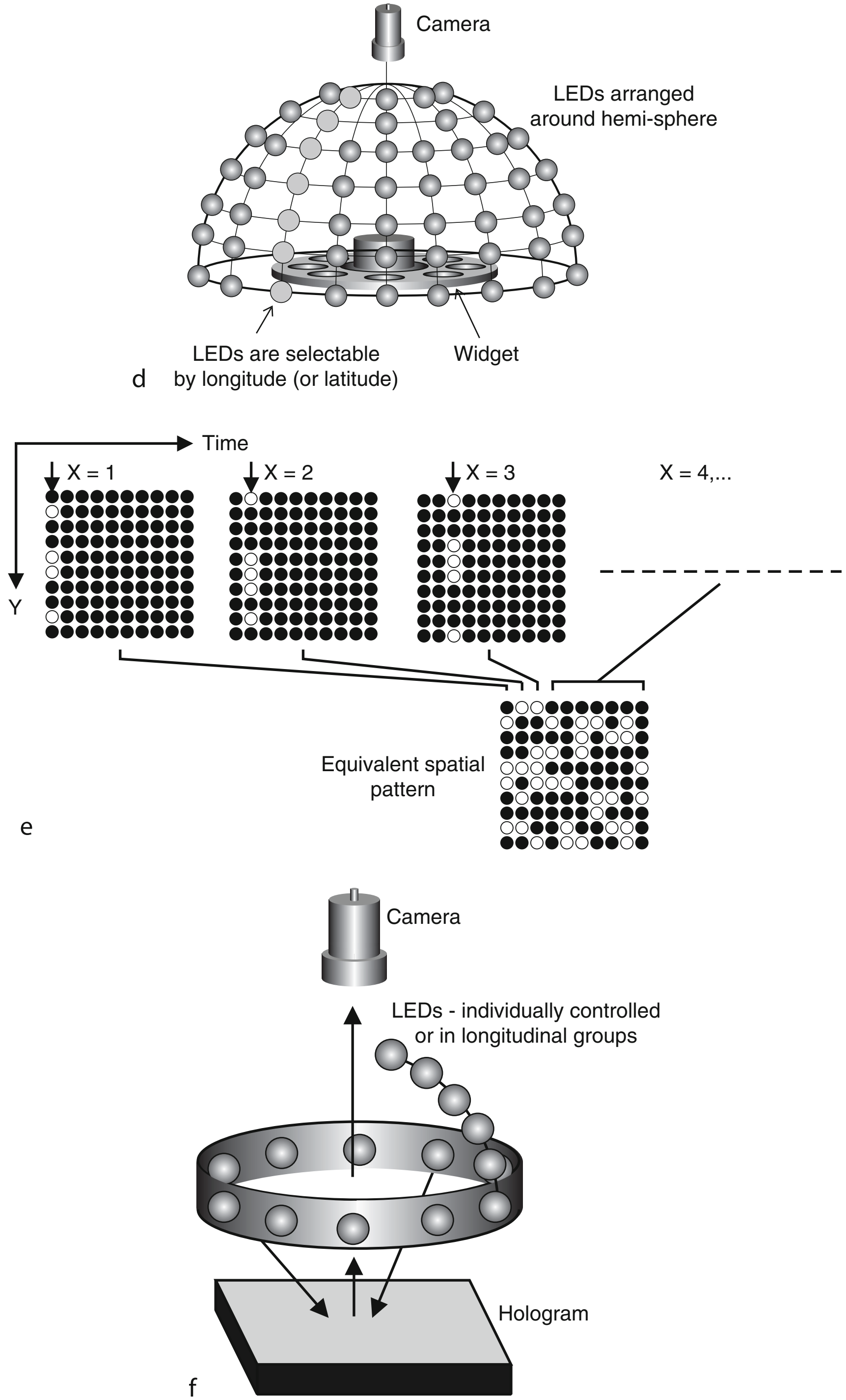


Fig. 40.3
(Continued)

EQUIPMENT One of the following:

(a) "Cat-o'-nine-tails" fibre optic bundle. The "loose" fibre ends are held in a grid to create the desired lighting pattern (➲ *Fig. 40.3a*)

(b) Printed circuit board, designed as a "bread board" with a large number of empty spaces. Single LEDs are then inserted as needed to define the desired pattern (➲ *Fig. 40.3b*)

(c) Back-light screen fitted with a suitable mask to create the desired lighting pattern

(d) Overhead or 50 mm slide projector, fitted with a suitable transparency to create the desired lighting pattern

(e) Array of LEDs, which can be strobed either individually or in groups. The array may be flat rectangular (➲ *Fig. 40.3c*), flat circular, folded in concertina fashion, cylindrical or hemispherical (➲ *Fig. 40.3d*)

(f) Computer with micro-mirror or LCD display projector. The computer "displays" the appropriate image onto the widget (A flying-spot projector cannot be used since this would create strobing effects.)

REMARKS (a) to (d) in the list above create fixed lighting patterns, while (e) and (f) provide a dynamic illumination effect in which the lighting pattern might be changed between successive video frames. While it would be useful to be able to control each LED in an array separately, this may not be convenient, particularly if it contains many diodes. A technique will therefore be described in which groups of LEDs (e.g., rows/columns in ➲ *Fig. 40.3c*, or lines of latitude/longitude in ➲ *Fig. 40.3d*) are switched together. For the sake of simplicity, we shall consider the rectangular array (➲ *Fig. 40.3e*). Contrary to intuition, this is not used to generate a 2D lighting pattern directly but instead creates a succession of "linear" patterns. There are *N* stages in this process, where *N* is the number of *X*-select lines. Each of these "linear" light patterns is obtained by illuminating the appropriate sub-set of the LEDs in one column of the array. While light cannot be stored, the charge patterns that this sequence generates in the camera can. The camera performs temporal integration of the light falling on its photo-detector array over (part of) one frame period, thereby producing a charge pattern that is the same as that would be produced by the equivalent static 2D pattern. Thus, at any given moment, one *X*-select line and several *Y*-select lines are set *ON* (➲ *Fig. 40.3c* and *e*). A different *X*-select line is then switched *ON* and a new bit-pattern is applied to the *Y*-select lines (The original *X*-select

Fig. 40.3

Creating arbitrary lighting patterns. (a) "Cat-o'-nine-tails" fibre optic bundle. (b) "Bread board" consists of a bare printed circuit board. (Current-limiting resistors are provided for each socket.) LEDs are inserted into the empty sockets, as needed. (c) Rectangular array of LEDs, Each column in the array is activated in turn, to define a "linear" light pattern, (d) LED array covering an hemisphere (For clarity, only the LEDs at the front of the hemisphere are shown here). For control purposes, this may be regarded as a folded version of the flat array. The LEDs may be grouped together along lines of latitude or longitude. (e) Sequence diagram for the fully populated rectangular array. (f) Verifying a hologram using a programmable light source. For clarity only a simple ring light is shown here.) When viewed in monochrome light, the intensity of a given point on a hologram varies with the angle of illumination. In white light, the colour changes.) Hence, by varying the lighting, a series of images can be obtained, which then be analysed easily, by template matching, to verify the identity of the hologram

line is, is of course, switched *OFF* first). This process is repeated by pulsing each *X*-select line in turn. Clearly, the camera integration time must be greater than *N.T*, where *T* is the flash duration (Notice that *T* also affects the magnitude of the charge pattern created in the camera and hence controls the image brightness). In this way, each LED can be controlled individually, even though several LEDs may be illuminated at the same time. However, only simple *ON/OFF* control is achieved. This arrangement is well suited for use in a prototyping system, or for applications in which the lighting pattern must be changed rapidly, perhaps between successive video frames. The LED array and fibre-optic methods are significant in the fact that the light pattern need not be flat. Methods (d) and (f) in the "*Equipment*" list effectively project a pattern of light emanating from a single, non-isotropic point source.
LAYOUT DIAGRAMS ➋ *Figure 40.3*

40.5 Method 5: Automatic Gain Control and Auto-Iris Lens

OBJECTIVE To compensate for varying light levels, when working in ambient lighting.
TYPICAL APPLICATIONS

- Surveillance and other security applications
- Monitoring the factory environment

EQUIPMENT Either of the following:

(a) Auto-iris lens. The output from the camera is fed back into the lens, which adjusts its aperture in an attempt to keep the video level constant
(b) Camera with automatic gain control (AGC). The camera adjusts its light-integration period in order to maintain a constant output level

OPTICAL ENVIRONMENT Slowly changing light levels can be accommodated by either method. It normally takes an auto-iris lens, or a camera AGC circuit, a few seconds to recover from a large step change in the light level.
REMARKS An auto-iris lens can operate either by computing the average or peak level from the video signal generated by the photo-sensor array. The former is more robust, since it is less susceptible to problems caused by glinting, although it does render the camera susceptible to saturation on highlights. Averaging is usually computed over the whole field of view but this need not be so; a small region in the centre of the field is often used instead. These two methods cannot be combined. The human eye has a dynamic range of about 10^{14}:1. Neither of these methods can match this; a high-grade video camera has a dynamic range of about 10^{9}:1. Notice that altering the iris size changes the depth of field and may increase aberrations for large apertures (small f-numbers). Raising the electronic gain increases camera noise.

40.6 Method 6: Back Illumination of Shiny Objects (Array Camera)

OBJECTIVES To view the silhouette of a shiny object.
TYPICAL APPLICATIONS

- Metrology and other applications where the edge contour must be located accurately
- Dimensional measurement of extrusions and lathe-turned objects

EQUIPMENT Collimator and telecentric lens. Both of these must be larger than the widget.
CAMERA Array.

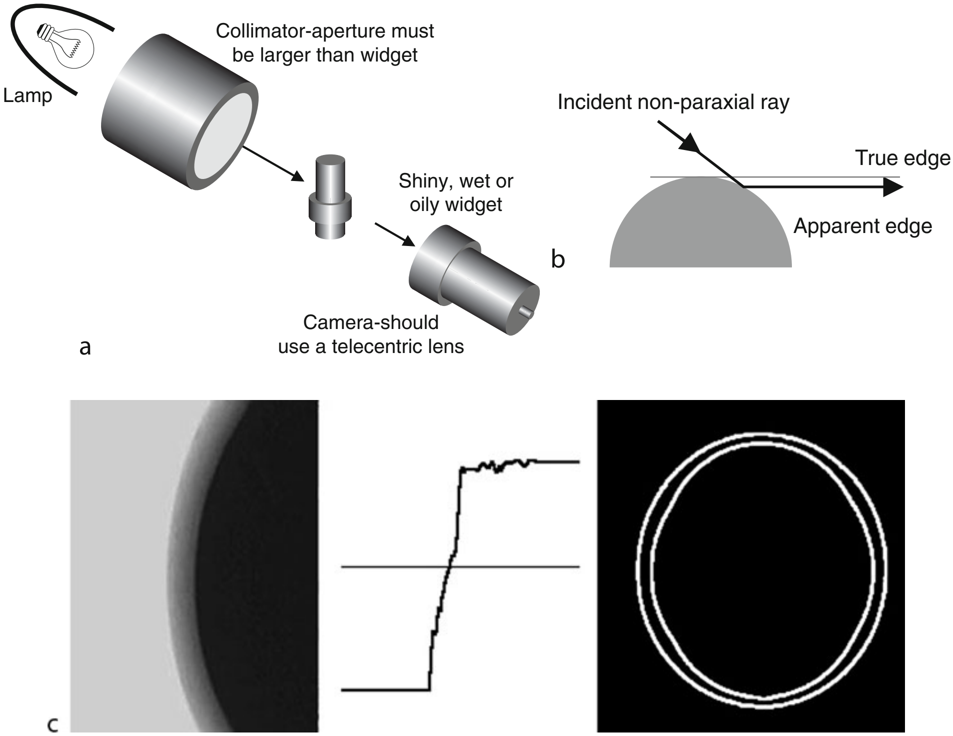
MOTION Static.

GEOMETRY Particularly suitable for objects with spherical, cylindrical and conical surfaces. Appropriate for any elongated object whose outer edge is gently curved along the optical axis. Flat objects and those with sharp outer edges do not need the collimator, since the edge is well defined by the simpler technique (Method 7).

MATERIALS Any shiny/glossy material (e.g., polished metal, smooth plastic, rubber, glass, ceramic), painted, lacquered, wet, oily and waxy surfaces.

REMARKS When viewing a shiny widget using the simpler technique (Method 7), we underestimate the size of the widget dimensions (See ➋ *Fig. 40.4b*). The collimator prevents reflections of non-paraxial rays occurring at points inside the outer edge. The illumination diagram is not an appropriate way to represent this lighting arrangement, since the rays are not all coincident at a single point.

DIAGRAMS AND SAMPLE IMAGES ➋ *Figure 40.4*


Lamp
Collimator-aperture must be larger than widget
Shiny, wet or oily widget
Camera-should use a telecentric lens
a
Incident non-paraxial ray
True edge
Apparent edge
b
c

▪ Fig. 40.4

Back lighting using a collimator to avoid reflections of non-paraxial rays. (a) Optical arrangement. (b) The effect of non-paraxial rays on simple back-illumination, when the collimator is not used. (c) *Left:* Sample image, derived from a brightly polished metal sphere in front of a rectangular "light box." Ordinary back-illumination (Method 17) produces a light grey annulus around a dark grey disc. The grey band is caused by reflections of non-paraxial rays. *Centre.* Intensity profile shows the gradual change of intensity within this grey band. *Right.* True contour (outer curve) and the apparent edge contour (inner curve), obtained when a collimator is *not* used. Notice that the errors depend on the radius of the light source (The errors are greatest opposite the corners of the light source: at 45°, 135°, 225° and 315°.)

40.7 Method 7: Back Lighting of Matt Opaque Objects (Array Camera)

OBJECTIVE To view silhouettes of non-shiny matt opaque objects.

TYPICAL APPLICATIONS

- Dimensional measurement and inspection of laminate objects, such as flat metal stampings, pressings, plastic mouldings, leather cuttings, etc.
- Dimensional measurement and inspection of a very wide range of 3D objects.

EQUIPMENT Proprietry back-light unit, or light box of the type used in hospitals for viewing X-rays.

CAMERA Array.

MOTION Static.

GEOMETRY Thin laminate, or any 3D object in which the relevant measurements may be obtained from its silhouette.

MATERIALS Opaque, non-reflective.

REMARKS Back-lighting is ideal for shape analysis and high-precision metrology. It operates on the same principle as the industrial shadowgraph and hence is readily accepted by industrial engineers. It is one of the oldest and most important lighting-viewing methods used in Machine Vision, since it produces high-contrast images that are easy to process. It is important that the back-ground be almost uniform (Do not rely on the human eye to judge this; use an image processor). In this case, the intensity histogram will resemble that shown in ➜ *Fig. 40.5c* and it is sensible to threshold the image at the bottom of the "valley" (point P). In many instances, a good estimate for the threshold parameter is at the centre of the intensity range actually found in the image. For example, we can easily estimate the minimum and maximum intensities (A and B) and find the average value: P = (A + B)/2. More generally, we may relate A and B to the statistical properties of the image. For example, let us suppose that F_A percent of the pixels are darker than level A and that F_B percent of the pixels are darker than level B (Typically, we might take F_A to be 5% and F_B to be 95%). Then, the threshold (P) might be computed using the more general formula: $P = [K.A + (1 - K).B]$, where $0 < K < 1$. In this way, it is possible to make the thresholding process independent of the brightness of the back-light source.

DIAGRAMS ➜ *Figure 40.5*

40.8 Method 8: Back Lighting of Matt Opaque Objects (Line-Scan Camera)

OBJECTIVE To view silhouettes of non-shiny matt opaque objects.

TYPICAL APPLICATIONS

- Dimensional measurement and inspection of laminate objects, such as flat metal stampings, pressings, plastic mouldings, leather cuttings, etc.
- Dimensional measurement and inspection of a very wide range of 3D objects
- Measuring width and position of extrusions, cables, etc. Examining edge integrity of same

EQUIPMENT Line scan camera, plus one of the following:

- Masked back-lighting unit
- Fibre-optic line illuminator

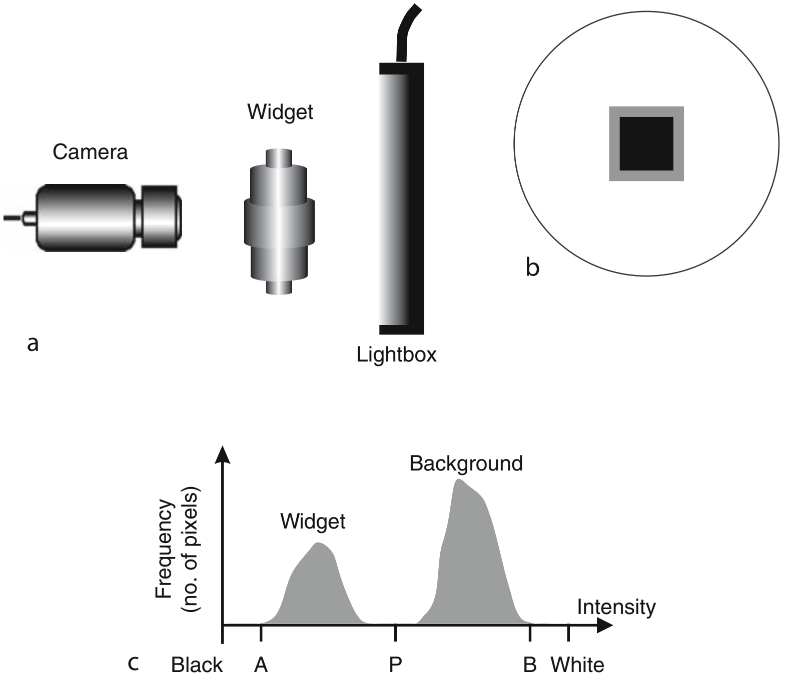


Fig. 40.5
Back lighting of matt opaque objects. (a) Optical layout. (b) Illumination diagram. (c) Intensity histogram for an idealised case

- Linear array of LEDs with a diffusing screen
- Light box of the type used in hospitals for viewing X-rays

CAMERA Line scan.
MOTION Smooth or indexed linear movement.
GEOMETRY Thin laminate, or any 3D object in which the relevant measurements may be obtained from its silhouette.
MATERIALS Opaque, non-reflective.
REMARKS See notes for Method 7. Several manufacturers make specialist linear back-lighting units.
DIAGRAMS *Figure 40.6*

40.9 Method 9: Calculate the Field of View: I

OBJECTIVE To calculate the field of view (FOV) for a flat (2-dimensional) widget that is always presented for viewing in the same orientation. The same analysis can be applied to the silhouette of a 3-dimensional object.
GIVEN PARAMETERS, NOTATION AND ASSUMPTIONS Orientation of the widget is fixed.
X – Width of the widget, measured along the x-axis.
Y – Height of the widget, measured along the y-axis.

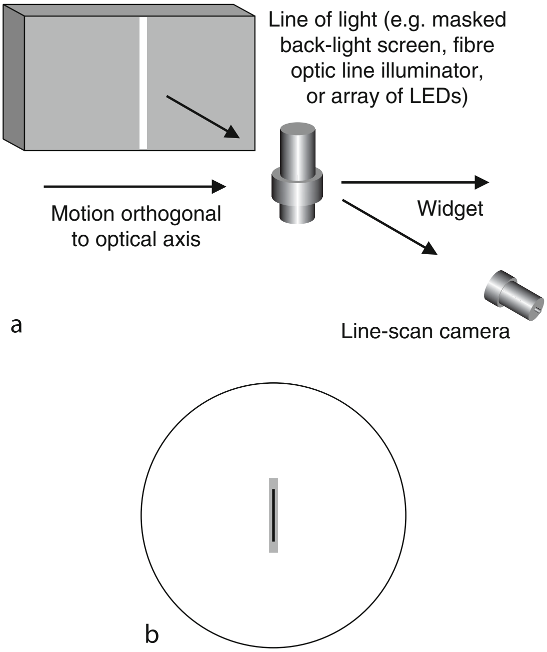

Line of light (e.g. masked back-light screen, fibre optic line illuminator, or array of LEDs)
Motion orthogonal to optical axis
Widget
Line-scan camera
a
b

Fig. 40.6
Back Lighting of matt opaque objects using a line-scan camera. (a) Optical layout. (b) Illumination diagram

Dx – Positional tolerance of the widget, measured along the x-axis.
Dy – Positional tolerance of the widget, measured along the y-axis.
Camera format:
  Array type
  Square image
  Pixels are square
  Resolution N^2 pixels
  Photo-detector spacing S

CALCULATED PARAMETERS

1. Size of the field of view (F = length of one side)

$$F = MAX((X + 2.Dx), (Y + 2.Dy))$$

   (In practice, the image projected by the lens onto the photo-detector array should be slightly larger than that array, to avoid “losing” edge pixels.)

2. Pixel size (length of one side)

$$F/N = MAX((X + 2.Dx), (Y + 2.Dy))/N$$

3. Minification required

$$M = F/[N.S] = MAX((X + 2.Dx), (Y + 2.Dy))/N.S$$

4. Processing efficiency (Compares the number of pixels in the minimum-enclosing-rectangle surrounding the widget, compared to the total number of pixels within the camera's field of view. We cannot estimate the "real" efficiency, since we have no idea how compact the widget is.)

$$10^2.X.Y/[MAX((X + 2.Dx), (Y + 2.Dy))]^2 \text{ percent.}$$

Assuming that Dx and Dy are small, this is approximately equal to

$$10^2.X.Y/[MAX(X, Y)]^2\text{percent}$$

If this figure is small, it may be worthwhile limiting the image processing to include only those pixels contained within a region of interest (ROI) of size X.Y. This contains $\{X.Y/[M.S]^2$ pixels.

DIAGRAM ➲ *Figure 40.7*

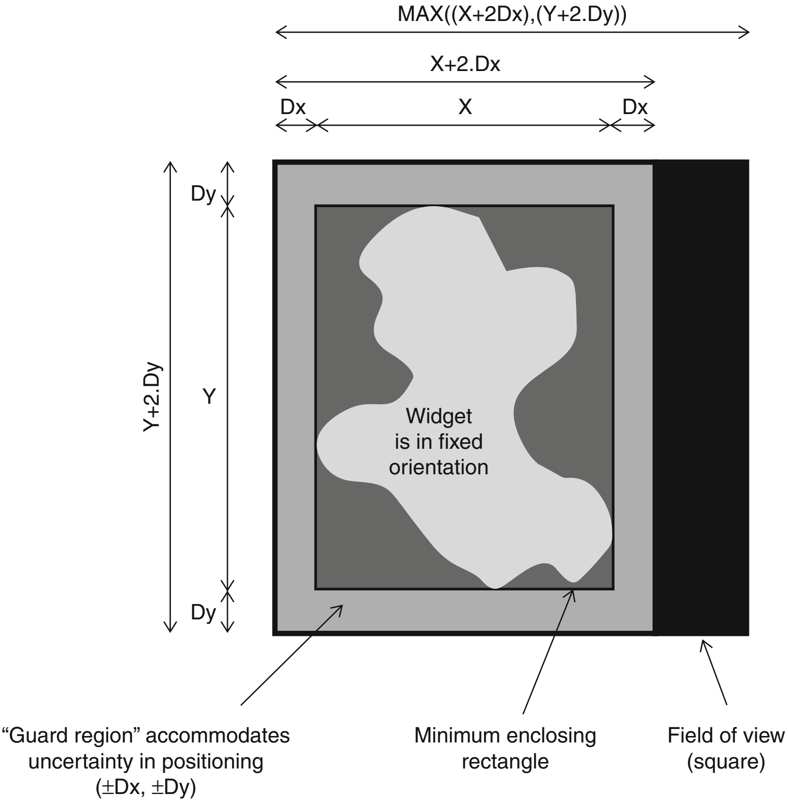

MAX((X+2Dx),(Y+2.Dy))
X+2.Dx
Dx
X
Dx
Dy
Y+2.Dy
Y
Dy
Widget is in fixed orientation
"Guard region" accommodates uncertainty in positioning (±Dx, ±Dy)
Minimum enclosing rectangle
Field of view (square)

Fig. 40.7

Calculating the FOV for a widget in fixed orientation. The field of view must be large enough to include the minimum-enclosing-rectangle, plus a "guard region," which accommodates the uncertainty in position of the widget

40.10 Method 10: Calculate the Field of View: II

OBJECTIVE To calculate the field of view (FOV) for a flat (2-dimensional) widget that is always presented for viewing in any orientation. The same analysis can be applied to the silhouette of a 3-dimensional object.

GIVEN PARAMETERS, NOTATION AND ASSUMPTIONS

X – Diameter of the minimum-enclosing-circle.

Dx – Positional tolerance of the widget, measured along the x-axis.

Dy – Positional tolerance of the widget, measured along the y-axis.

Camera format:

Array type

Square image

Pixels are square

Resolution N^2 pixels

Photo-detector spacing S

CALCULATED PARAMETERS

1. Size of the field of view (F = length of one side)

$$F = X + 2.MAX(Dx, Dy)$$

(In practice, the image projected by the lens onto the photo-detector array should be slightly larger than that array, to avoid "losing" edge pixels).

2. Pixel size (length of one side)

$$F/N = [X + 2.MAX(Dx, Dy)]/N$$

3. Minification required

$$M = F/[N.S] = [X + 2.MAX(Dx, Dy)]/[N.S]$$

4. Processing efficiency (See Method 1.)

$$10^2.X^2/[X + 2.MAX(Dx, Dy)]^2 \text{ percent.}$$

REMARKS The minimum-enclosing-circle can be computed by a straightforward image processing operator (QT operator *mbc*. The FOV calculation for a widget that is allowed to vary in orientation by a finite amount ($\pm\theta$, where $\theta < 180°$) is more complicated. This requires the computation of the dimensions of the largest minimum-enclosing-rectangle for a range of angles of rotation, $[-\theta, +\theta]$. The formula given above for F is safe but slightly pessimistic. The QT operator *fov* is described in ➋ Chap. 41.

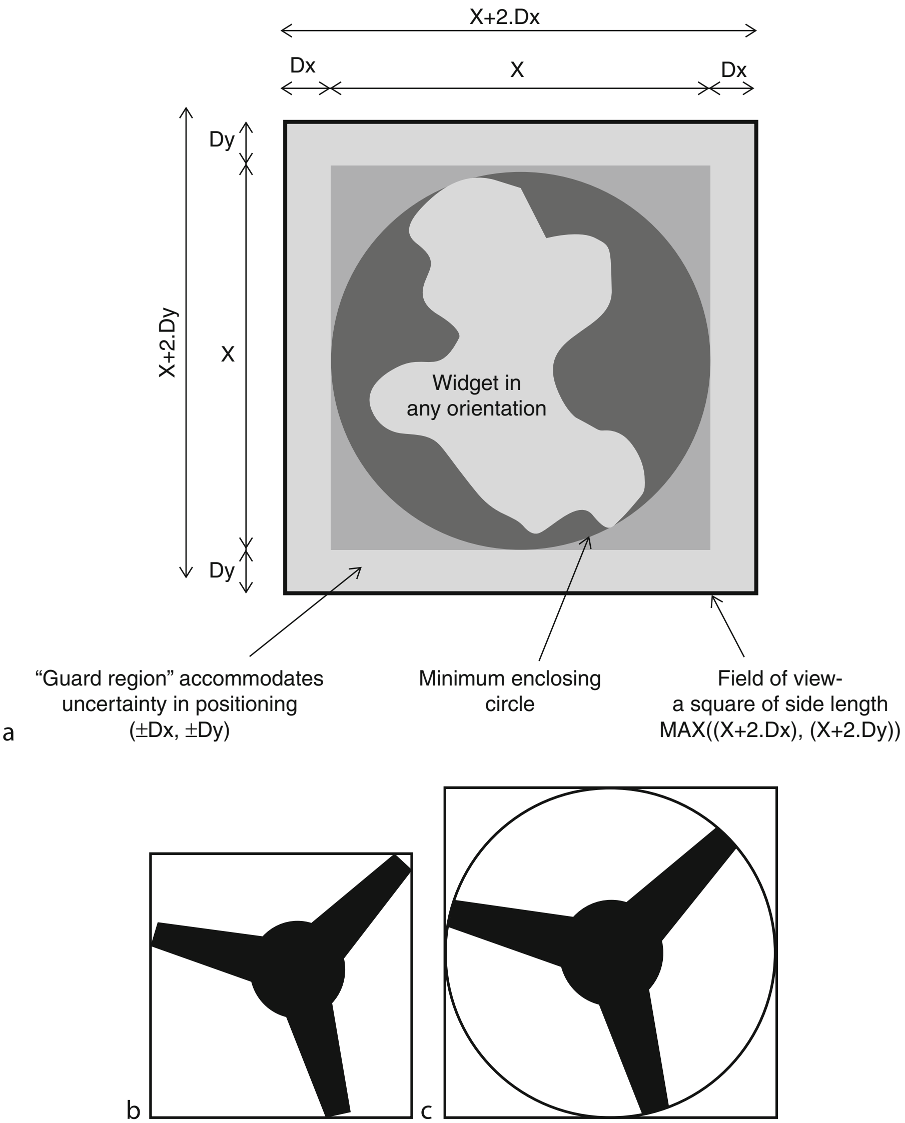
DIAGRAMS ➋ *Figure 40.8*

40.11 Method 11: Calibration: Calculating Spatial Resolution

OBJECTIVE To calculate the effective size of the pixels in a digital image.

ASSUMPTIONS AND NOTATION

- An array camera uses N^2 square photo-detectors of dimension D^2
- The photo-sensor array is based on a square grid of pitch S^2. Since there is a "dead space" between adjacent photo-detectors, D is less than S



◘ Fig. 40.8

Calculating the field of view for a widget in any orientation. (a) The FOV must be large enough to include the minimum-enclosing-circle, plus the ‘‘guard region.’’ X is the diameter of the minimum-enclosing-circle. (b) The FOV for the variable-orientation case *(right)* is usually larger than that needed for Method 1 *(left)*. In this example, 34% more pixels are needed

- The minification of the optical system is M. This is the ratio of the size of a feature as it appears in the image plane (i.e., on the photo-sensor array) to the size of the same feature within the original scene Minification is the reciprocal of magnification

CALCULATED PARAMETERS

1. Size of the field of view (F = length of one side)

$$F = N.S.M$$

2. Size of the virtual pixels

$$V = M.S$$

3. Size of the photo-detector array

$$P = N.S$$

4. Photo-sensor light-collecting efficiency (This is a function of the photo-sensor geometry and cannot be changed except by using a different camera.)

$$E = 100.D^{2/}S^2$$

   This means that E% of the photons projected onto each photo-detector in the array contribute towards its output signal.

5. Minimum-size feature (spot diameter/arc width) that can be seen (Nyquist limit)

$$2.M.S.$$

   In practice, a larger value is needed to obtain reliable measurements; a more realistic limit is about 2–3 times this value:

$$4.M.S \text{ or } 6.M.S.$$

   The maximum number of line-pairs that can be seen reliably is therefore about

$$N/8 \text{ or } N/12.$$

REMARKS Imagine looking into the lens towards the sensor array, with your eye located in the object plane. The sensor array appears *magnified* by a factor M. Hence, each of the photo-sensors appears to be of size $[M.D]^2$, located on a grid of pitch $[M.S]^2$. The integral of the light flux emanating from the square "covered" by the magnified photo-detector determines the intensity of the corresponding pixel within the digital image. It is not correct to say that the average intensity within the virtual pixel determines the pixel intensity, although this is a reasonably accurate approximation in regions of low spatial frequency. M can be determined experimentally using the following procedure.

CALIBRATION PROCEDURE

1. View a matt black disc of known diameter (φ), using back-illumination (Method 7). This disc is placed in the object plane. A high-contrast printed sheet with front lighting (Method 114) might suffice instead
2. Segment (threshold) the resulting digital image to isolate the image of the disc
3. Measure the height (H) and width (W) the image of the disc, in terms of pixels counted along the vertical and horizontal axes
4. Compute the parameters $M_h = \varphi/H$ and $M_v = \varphi/W$, which should be very nearly equal (If not, the camera's photo-detector array is not based on a square grid, the lens provides

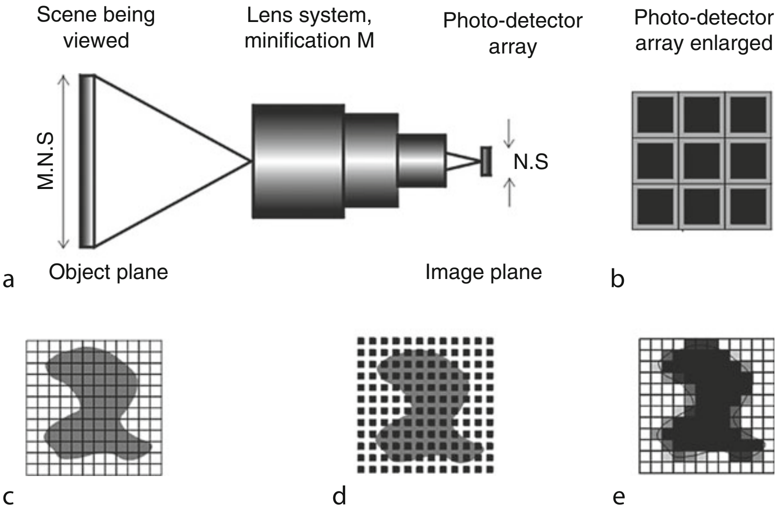


Fig. 40.9
Forming a digital image by sampling the intensities within a scene lying on a plane. (a) Optical system. (b) The optical system projects an image onto an array of non-contiguous photo-sensors (*black*). Since each one of these is smaller than the pitch of the array, there is a dead area around each one. (*grey*) (c) Binary object, with the virtual pixels superimposed. (d) The photo-detector array samples the intensity distribution within the scene. (e) Digital image with the outline of the binary object superimposed

different levels of minification along the vertical and horizontal axes, the test object is not normal to the optical axis, or the frame-grabber hardware is set up to generate non-square pixels)

5. Assuming that M_h and M_v are very nearly equal, take the minification to be the average:

$$M = (M_h + M_v)/2.$$

Another, more accurate measurement is given by

$$M = \varphi.\sqrt{(\pi/A)}$$

where A is the area of the image of the disc (i.e., number of pixels in this "blob").

40.12 Method 12: Circle of Light Projecting Radially Outwards

OBJECTIVE To project a continuous circle of light around the surface of a cylindrical bore.
TYPICAL APPLICATIONS

- Inspecting the bores of pipes, tubes, drains, sewers, etc.
- Examining holes in castings, mouldings and machined piece parts.

EQUIPMENT Any one of the following:

- Optical probe consisting of two axicons and a conical mirror (An axicon is a cone of transparent material. See Method 88).

- Laser fitted with diffractive optics to generate a continuous ring of light, and a conical mirror
- Standard ring light (LED or fibre optic) and a conical mirror
- Custom array of LEDs, arranged in a circle and projecting their beams outwards
- Custom fibre-optic ring light, with the fibres arranged to project light outwards, rather than forwards

CAMERA Array scan. Alternatively, a "wheel-spoke" or circular-scan camera would be useful, as this may simplify the image processing.

MOTION Indexed or continuous linear movement along the axis of the bore.

GEOMETRY Medium-large diameter bore (The probe must be able to enter the bore, leaving a sufficiently large gap to enable the camera to view its surface).

REMARKS Ideally, this method would produce the same effect as a point source of light that emits light uniformly in all directions within a plane. No rays are emitted except in that plane. If light from such a source were projected normally onto the internal surface of a cylinder, a bright circle of light would be created. In a practical optical system, a circle of finite diameter replaces the point source; all light rays then travel radially outwards from that circle. A circular light source can be created in a variety of ways:

- Using a circular array of "point" sources (e.g., LEDs or optical fibres)
- Diverting a hollow cylinder of light using a conical mirror. Such a light cylinder may be generated using a laser and a diffractive optics lens, or axicons

To reduce aberrations, a lens with a high f-number (i.e., small aperture) should be fitted to the camera.

LAYOUT DIAGRAMS ➲ *Figure 40.10*

40.13 Method 13: Circular Features

OBJECTIVE To sample a scene using circular, or "wheel-spoke" features.

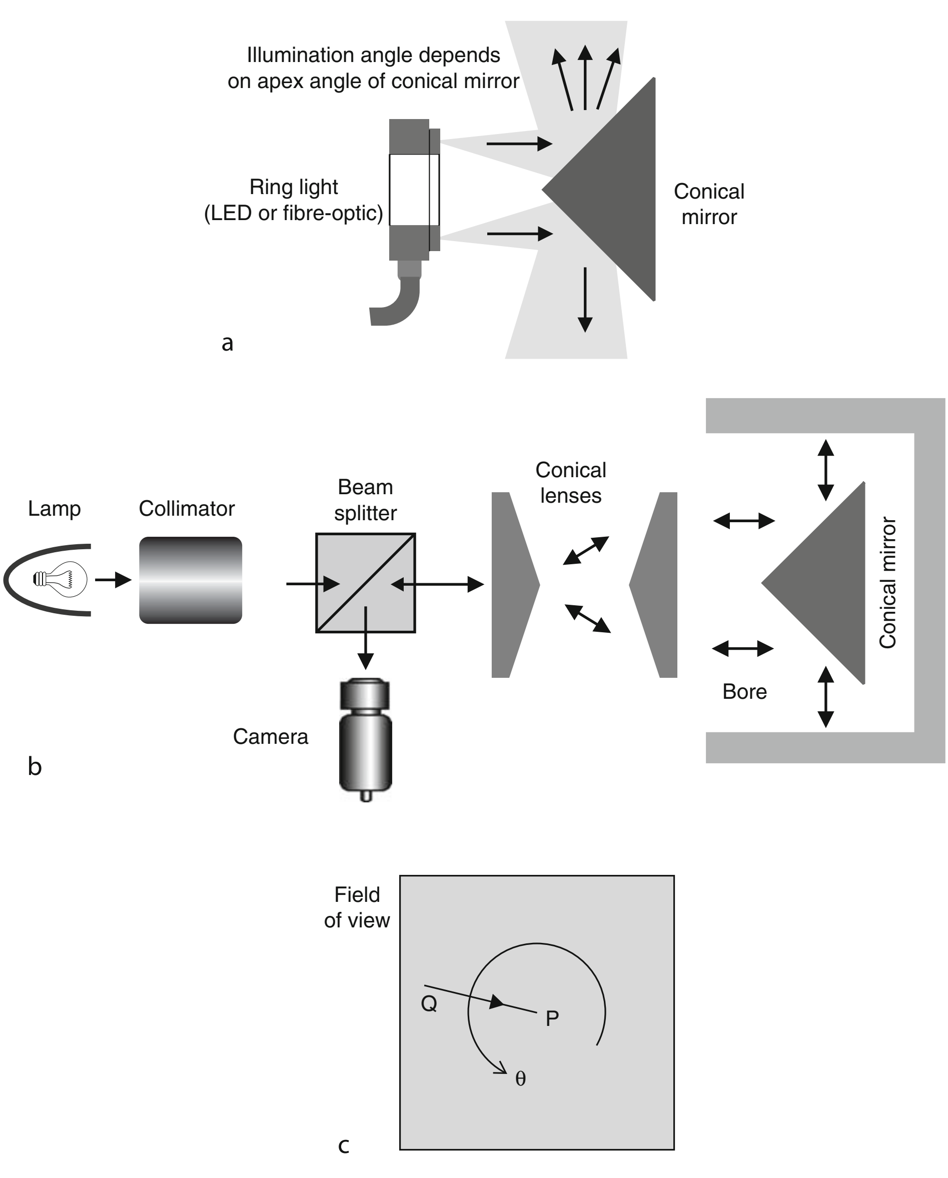
TYPICAL APPLICATIONS Inspecting tops of bottles, jars, tumblers, cans, circular holes, gears, bore surfaces: tubes, pipes, drains, etc.

EQUIPMENT Solid-state or CRT camera with a circular-scan, multi-ring, ring-and-wedge, spiral-scan or radial-scan pattern.

OPTICS Conventional lens. These cameras can also be used in conjunction with various lighting-viewing arrangements.

POSTURE The circular/radial features should be concentric with the camera's field of view.

REMARKS The sole reason for using non-standard scan patterns like those shown in figure is to avoid storing and manipulating "unwanted" pixels and to structure the image data in such a way that processing is made easier. Similar effects can be obtained by digitising the signal from a high-resolution array camera and then sampling the resultant digital image using software, or special purpose hardware. Although it is possible to build a variety of non-standard solid-sate photo-sensor arrays, tube cameras are generally easier and cheaper to use. The reason is that their scan patterns are more versatile and can be modified at will. For example, the pitch of a spiral raster can be adjusted by modifying the drive signals applied to the beam-deflection coils/plates. On the other hand, the array geometry of a solid-state photo-sensor is fixed during manufacture and it is very expensive to produce special-purpose devices to order. Electrically, a circular-scan camera (➲ *Fig. 40.11a*) is similar to a line-scan camera; its photo-detectors are



◘ Fig. 40.10

Circle of light projecting radially outwards. (a) Using a standard ring light and a conical mirror. (b) Using two axicons and a conical mirror. The beam-splitter enables coaxial illumination and viewing. (c) Camera's field of view. As the probe advances into the bore, surface features move along radial lines, such as PQ, disappearing from view when they reach the centre of the image, P. θ is the angular position

arranged around a circle. The ring-wedge detector (➔ *Fig. 40.11b*) is intended specifically for analysing Fourier transforms produced optically (Method 74). This array geometry makes use of the symmetry of Fourier transforms (i.e., $F(A, \omega) = F(-A, -\omega)$). The scan pattern shown in ➔ *Fig. 40.11c* produces a similar digitisation effect to that achieved by Method 98 (Rotating Dove Prism) A spiral scan (➔ *Fig. 40.11d*), produced by a modified vidicon, has been used to inspect bottle tops. The "wheel spoke" pattern in➔ *Fig. 40.11e* is similar to that used in rotating radar scanners. Possible uses of these cameras is evident elsewhere: Methods 116, 97, 66 and 59.

SCAN PATTERNS ➔ *Figure 40.11*

40.14 Method 14: Circularly Polarising Filter Suppresses Glinting

OBJECTIVE To suppress highlights due to glinting, when viewing bright shiny reflective objects.

TYPICAL APPLICATIONS Inspecting metal machine parts.

EQUIPMENT Polarising filter, IR-blocking filter.

CAMERA Array or line scan.

MOTION Static or linear motion.

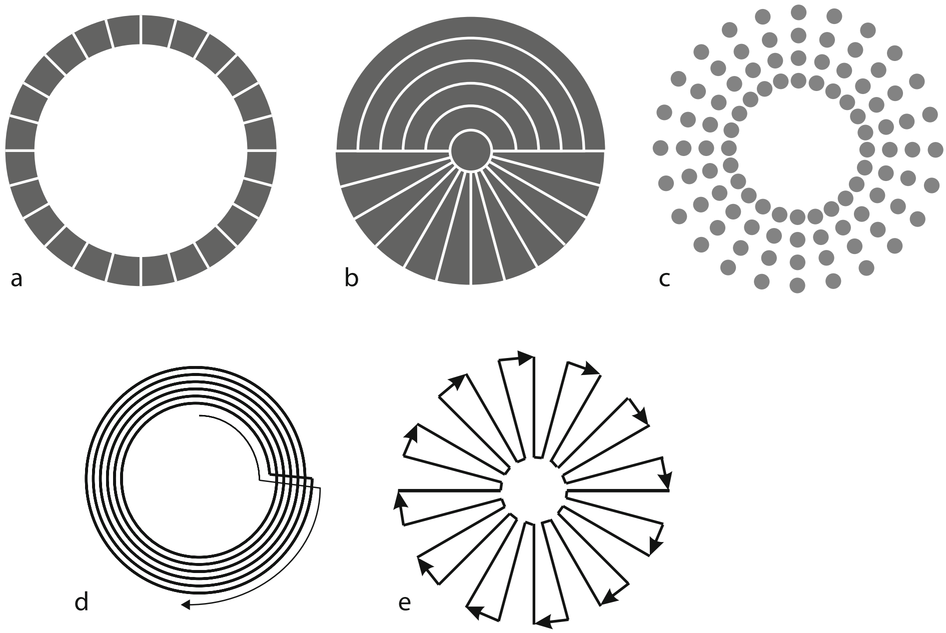


Fig. 40.11

Non-standard scan patterns. (a) Solid-state circular-scan array, consists of a single ring of photo-detectors. (b) Solid-state ring-and-wedge detector, designed to analyse optical Fourier transform patterns. (c) Solid-state array of photo-detectors arranged around a series of concentric rings. Notice that the photo-detectors are all the same size, to ensure that the sensitivity is constant throughout the whole field of view. (d) Spiral-scan (relies on a CRT camera). (e) Radial-scan pattern (relies on a CRT camera)

MATERIALS Bright metal.
REMARKS The outward (illumination) and return (i.e., after reflection) beams must both pass through the polariser. Some polarising filters do not function effectively in the near infrared wave-band. Moreover, incandescent lamps emit strongly in the NIR and solid-state image sensors are sensitive in this part of the spectrum. So, it may be necessary to place an IR-blocking filter between the widget and camera. Circular polarising filters must be inserted in the correct orientation in the optical path (They cannot be "flipped over"). Polarising material in the form of a rigid plastic sheet is available from several major optical manufacturers. When using a low-cost sheet polariser, highlights due to glinting appear as deep violet spots, which can be further attenuated by using a long-pass optical filter. To determine the correct orientation for a circular polariser, place it on a shiny metallic surface and observe the surface with the room lights on. Now, flip the polariser and try again. In the proper orientation, the metal surface looks dark and usually blue. This method can be used to make diffuse reflecting material more prominent.
LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.12*

40.15 Method 15: Coaxial Diffuse Illumination

OBJECTIVE To provide diffuse illumination without causing glinting.
TYPICAL APPLICATIONS Viewing small bright shiny objects, such as coins, electrical connectors, printed circuit boards.
EQUIPMENT Small incandescent filament lamp, LED array or fibre-optic illuminator, diffusing plate and beam-splitter.
GEOMETRY

- Mirror-like surfaces
- Complex, embossed, etc.

MATERIALS

- Bright metal surfaces
- Reflective plastic
- Lacquered/painted surfaces

REMARKS Excellent images can be obtained from mirror-like surfaces, which are often difficult to view using other methods. It appears as though the widget is illuminated by light emanating from a small area, corresponding to the front of the camera lens. Since this produces a much narrower illumination beam than Method 31, it is very sensitive to the geometry of the widget surface; it can readily detect embossing, engraving, stippling creasing, dimples, pimples, scratches and pits. This method relies on producing specular reflection. Hence, the widget must have a well-defined shape and be held in fixed orientation, to ensure that glinting is visible to the camera.
DIAGRAMS AND SAMPLE IMAGE ➋ *Figure 40.13*

40.16 Method 16: Coherent Flexible Fibre Optic Bundle

OBJECTIVES

- To transmit an image through a tortuous path
- To acquire an image from the end of a robotic arm

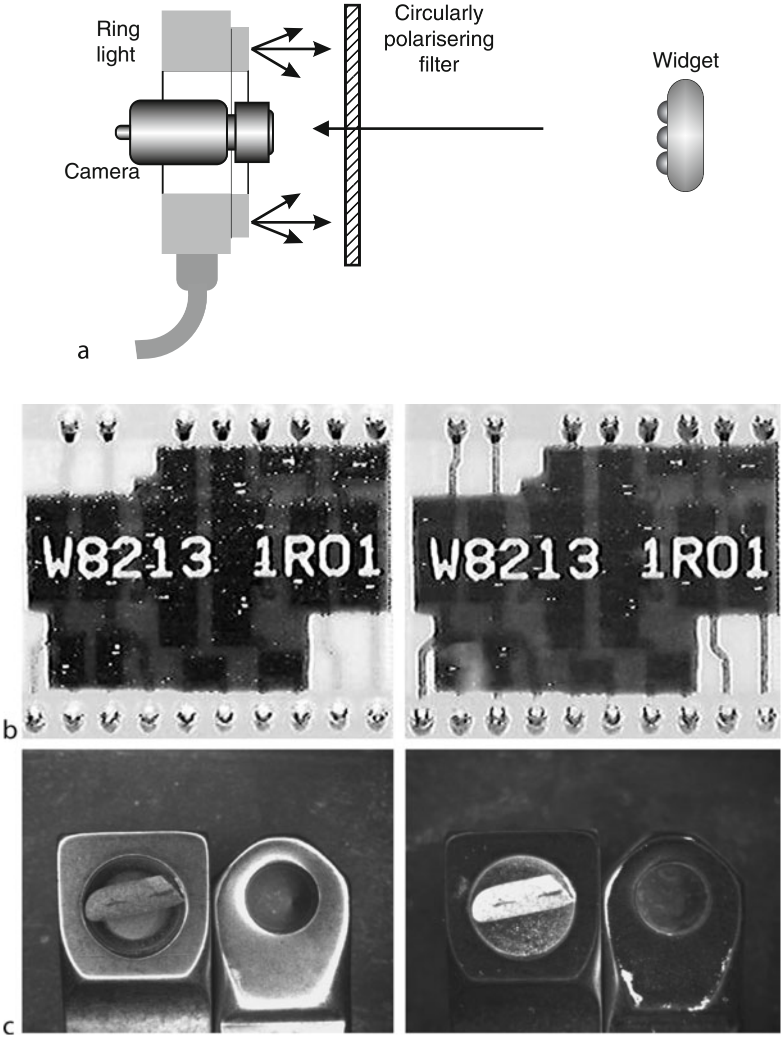


Fig. 40.12
Circularly polarising filter. (a) Optical layout. (b) Sample image: Thin-film circuit. *Left:* With no polariser present. *Right:* With polariser in place. Notice the enhanced visibility of the thin conductor tracks. (c) Making a diffuse reflecting material more prominent Shiny metal object with stone. *Left:* ring light, no polariser. *Right*: Ring light and circular polariser. [Images kindly supplied by Spencer Luster, Light Works, Inc., URL http://www.lw4u.com]

TYPICAL APPLICATIONS

- Internal inspection of pipes, tubes, drains, sewers, air ducts, etc.
- In situ inspection of aero-engines, turbines, etc.

EQUIPMENT Flexible fibre optic (FO) bundle.
CAMERA Array scan.
LENS Lens is fitted to the end of the fibre optic bundle.
MOTION Static.

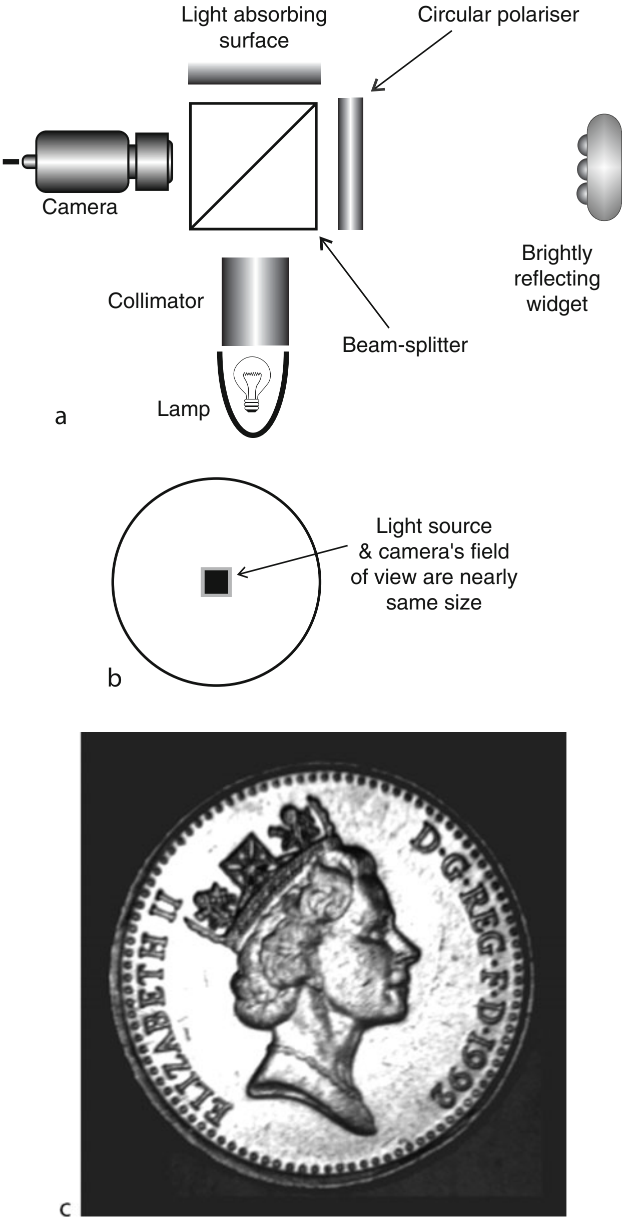


Fig. 40.13

Coaxial diffuse illumination. (a) Optical layout. The circular polariser is one option for providing lighting and is only needed when glinting occurs. (b) Illumination Diagram. (c) Sample image: Cupro-nickel coin. There is some wear evident on the flat surface (called the ''table''), above and below the queen's head

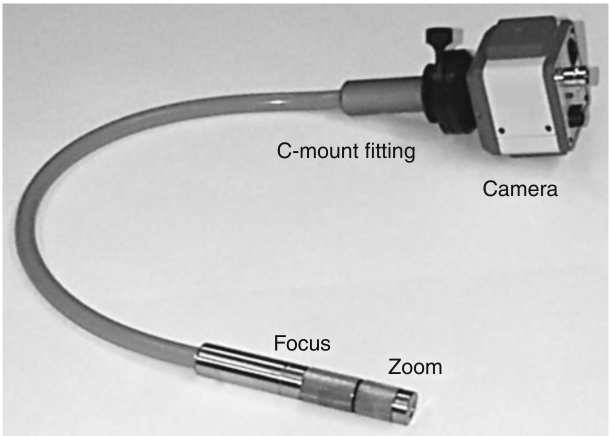


Fig. 40.14
Flexible fibre-optic imaging bundle. Much longer bundles (over 3 m) are available but are expensive

GEOMETRY Objects with holes.
REMARKS Flexible coherent fibre optic bundles for image transmission (as distinct from non-coherent lighting bundles) are more expensive than image conduit but are more versatile. Some very sophisticated FO bundles are used in medicine and are fitted with a variety of tools (surgical, lighting, air, vacuum, etc.). All FO bundles superimpose a fine mesh-like pattern onto an image (See Method 38.). Flexible fibre bundles can be constructed using glass, quartz (for viewing UV images), and plastic. However, they can be damaged relatively easily, by bending through too small a radius, heat (damages plastic bundles), radiation (ordinary glass turns brown) and continued flexing over a prolonged period. There is a gradual degradation of image quality, as the bundle is flexed repeatedly; individual fibres break, causing dark spots to appear in the image. For this reason, FO bundles are not recommended nowadays for use on robots; miniature cameras are normally preferred, instead.
PHOTOGRAPH ➋ *Figure 40.14*

40.17 Method 17: Collapse Image into Line

OBJECTIVE To detect linear features at a known angle, by integrating the intensity along one axis, to produce a 1-dimensional light pattern.
TYPICAL APPLICATIONS
- Counting slab- or disc-like objects, such as biscuits, cookies, tea bags, sweets, etc. stacked in a linear array
- Analysing the texture of cloth
- Detecting the level of a liquid in a glass bottle or jar
- Identifying blocks of text on a printed page, as a prelude to measuring line spacing and determining character size

EQUIPMENT Converging cylindrical (i.e., biconvex, or plano-convex) lens.
CAMERA Line scan.

MOTION Static.

REMARKS Consider *Fig. 40.15a*. For light rays travelling in a horizontal plane (Y is constant), the normal lens formulae apply. A light ray travelling in a vertical plane (X is constant) as it enters the cylindrical lens, emerges from it having undergone a simple vertical shift; the direction of travel is unchanged. The deflection is proportional to the thickness of the lens at the point of entry. To understand how this method can be applied in practice, consider a stack of flat disc-like objects of variable thickness (e.g., cookies). Looking sideways at the stack, we see a stripe pattern of no fixed period; each stripe corresponds to one cookie in the stack. Since the edges of cookies are rough and irregular, a simple intensity profile would not yield an accurate way to count them. However, by integrating the intensity, whether by image

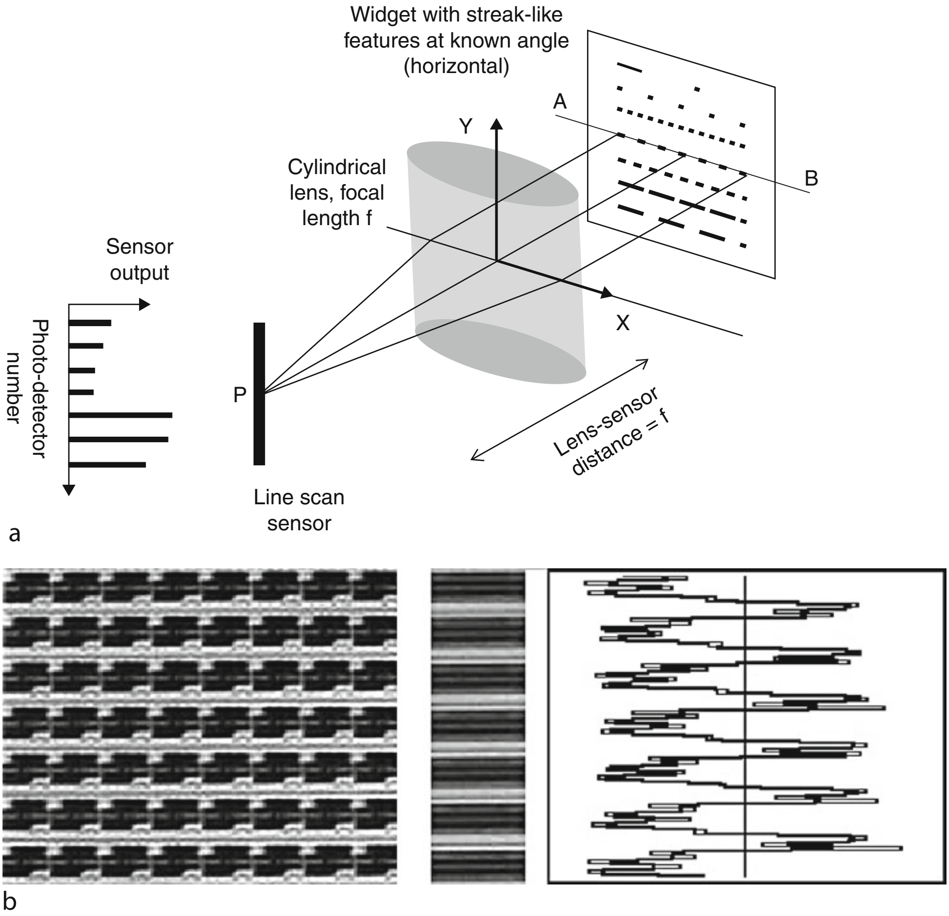

Widget with streak-like features at known angle (horizontal)
A
B
Y
X
Cylindrical lens, focal length f
Sensor output
Photo-detector number
P
Line scan sensor
Lens-sensor distance = f
a
b

Fig. 40.15

Integrating the intensity along a defined axis, using a cylindrical lens. (a) Optical layout. Paraxial rays emanating from all points along the line AB are focussed onto point P and hence contribute to the output from the line-scan camera. (b) Sample image. Array of thin-film circuits. *Left*: Original image. *Centre:* View through a cylindrical mirror, aligned to integrate horizontally. *Right:* Output of the line-scan camera (simulated)

processing, or optically as described here, a much more accurate estimate can be obtained. A cylindrical lens combined with Method 98 can provide an opto-mechanical implementation of the Radon Transform. (This is a useful image processing operator, which is used to detect linear structures in a scene.)
LAYOUT DIAGRAM AND SAMPLE IMAGE ➜ *Figure 40.15*

40.18 Method 18: Collecting Natural Light/Sun-Light

OBJECTIVE To illuminate the widget, or a colour calibration target, using natural light.
TYPICAL APPLICATIONS Calibration of colour vision systems, used for inspecting printing, paint-work and fabrics, etc., where precise colour matching is essential.
EQUIPMENT A light-pipe formed from a roll of *Silverlux*™.
REMARKS The light pipe is a straight, vertical tube, formed from a roll of *Silverlux*™ (This is a thin, flexible sheet of silvered aluminium.) The tube may be several metres long, typically with a diameter of about 250 mm. The top of this tube can be arranged to project through the factory roof and can be fitted with a variety of light-collecting optics (➜ *Fig. 40.16b*). The light-entry window can be masked to avoid direct illumination by sun-light, or deliberately arranged to collect as much light as possible, whatever the time of day and year. Among the options are:

- Hemi-spherical dome (collects maximum amount of light, from all parts of the sky)
- Cone (apex pointing down-wards, captures sun-light)
- Vertical tube with a black, rifled bore (captures diffused light, blocks sun-light except in the Tropics)
- Masks, louvres, etc. (block/admit sun-light for specific values of elevation and azimuth)

The spectrum of the light transmitted by the light pipe is a very close approximation to that existing in a natural environment and includes VIS, IR and UV components. Of course, care must be taken to prevent light from artificial sources from entering the pipe and to keep the light-collector clean.
REFERENCES

- *Skypipe*™, Kennies P, Heating and air conditioning. http://www.kennies.com/skypipe.htm. Accessed July 2002
- *Silverlux*™, 3M Company, St. Paul. www.3m.com. Accessed Sep 2011

LAYOUT DIAGRAM ➜ *Figure 40.16*

40.19 Method 19: Colour Filtering

OBJECTIVE To provide sharp discrimination between different optical wavelengths in near-UV, VIS and near-IR bands (Separates colours in VIS.).
TYPICAL APPLICATIONS

- Inspecting printing, examining labels on packaging, etc.
- Inspecting woven textiles and printed fabrics
- Reading resistor colour codes
- Used in conjunction with coloured lighting (e.g., switched LEDs), for inspecting holograms, analysing
- Analysing fluorescence (Methods 24 and 41)

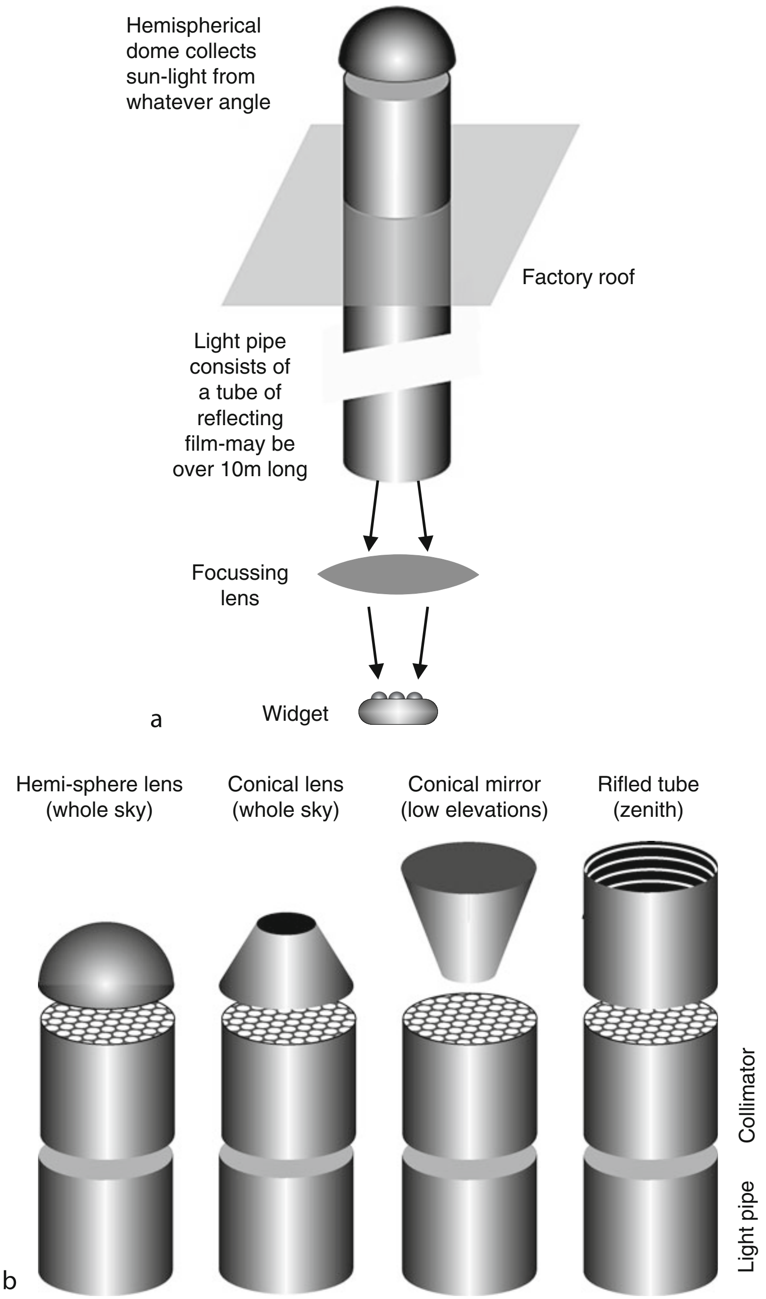


Fig. 40.16

Light pipe to produce natural lighting over a small area in the middle of a large building. (a) Optical arrangement (b) Selecting light rays from preferred directions. The collimator is optional

EQUIPMENT Coloured glass or gelatine filter, interference filter, dichroic filter.
REMARKS Optical filters are available in several forms. Filters approximating human colour detectors (intended for colour television applications) are available. Interference filters provide a sharper cut on/off characteristic than absorptive (i.e., coloured glass/gelatine) filters. Interference filters are more expensive and are susceptible to damage from moisture and heat. Interference filters change colour (i.e., cut-off wavelength changes) when they are tilted relative to the optical path. This allows fine tuning of the cut-on/cut-off wavelength (➲ *Fig. 40.17a*). Several factors must be borne in mind when choosing a filter:

(a) The camera's spectral response
(b) The emission spectrum of the light source
(c) The reflectance characteristics (i.e., colour variation) of the widget
(d) The absorption characteristics of the lenses and other elements in the optical system

Interference filters are available from all major optical manufacturers. A long-pass dichroic filter for which $\lambda_1 \approx 0.7$ μm is called a *cold mirror*; reflects VIS and transmits IR. A long-pass dichroic filter with $\lambda_1 \approx 0.7$ μm is called a *hot mirror*, since it reflects IR and transmits VIS. The latter is particularly useful for protecting the camera when inspecting very hot objects (e.g., red-hot steel).
FILTER CHARACTERISTICS ➲ *Figure 40.17*

40.20 Method 20: Combined Back and Front Illumination

OBJECTIVE To provide a method for viewing opaque objects which have piercings, so that edge and internal/surface features can be seen simultaneously.
TYPICAL APPLICATION Inspecting printed circuit boards, to verify that the holes and pads are aligned.
EQUIPMENT Back-lighting unit and flood lamps. Colour filters (optional).
CAMERA Colour or monochrome array camera.
MOTION Static.
GEOMETRY Has holes/piercings.
MATERIALS Opaque, non-shiny.
REMARKS The colour filters allow the images from the silhouette and front face to be separated or combined, as appropriate. Another simpler optical arrangement is possible, by simply omitting the colour filters. In this case, a monochrome camera can be used and the intensity of the front and back illumination sources must be carefully balanced, to ensure that the surface features and silhouette can both be seen clearly.
LAYOUT DIAGRAM ➲ *Figure 40.18*

40.21 Method 21: Field Splitter Combines Several Views

OBJECTIVE To select several views of an object and to combine them for a single camera.
TYPICAL APPLICATIONS

- Viewing the head and point of a long thin object (e.g., nail)
- Inspecting complex objects
- Inspecting a printed circuit board by comparing it with a "golden sample."

- Width measurement/positioning of web products (The camera views opposite edges of the web)
- Inspecting the pattern on a printed web, by comparing it with a golden sample, or another pattern displaced across or down web
- All round inspection of a widget
- Taking two orthogonal diameter measurements simultaneously to give better accuracy when assessing size and/or position

EQUIPMENT 45°/45°/90° prisms, or front-coated plane mirrors.

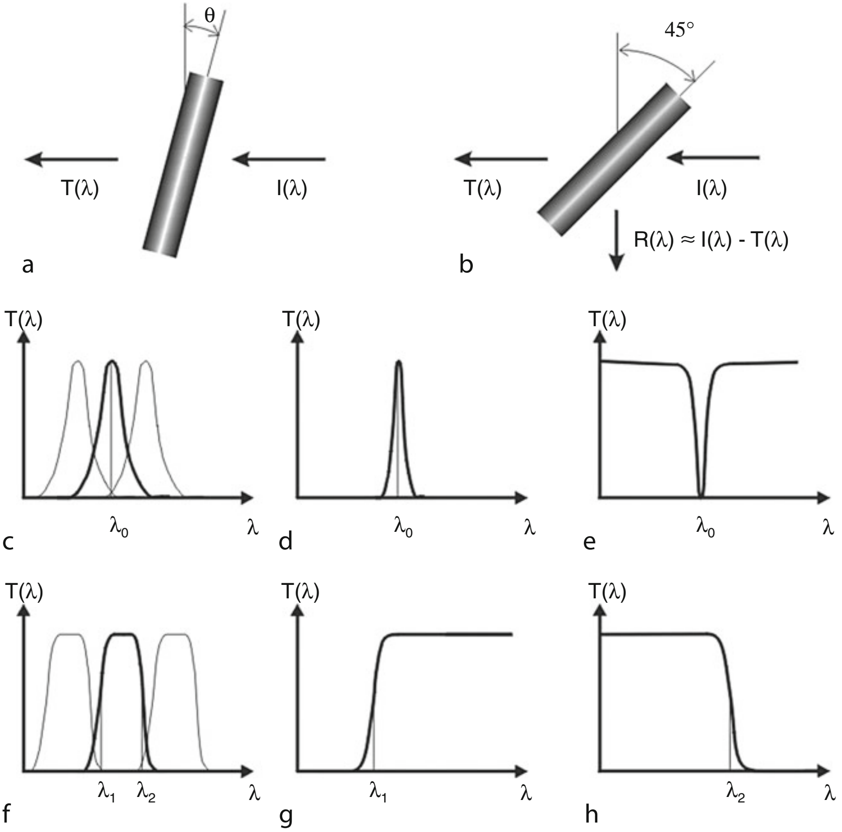
CAMERA Array.

LENS Standard lens.

MOTION Static.

GEOMETRY Long thin object.

REMARKS This method is an adaptation of the periscope. The optical path length may vary considerably in different parts of the camera's field of view, in which case the sub-images may be magnified by different amounts. Since miniature cameras have become available, this optical set-up is not nearly as important as it once was. The same effect can be achieved using low-cost video camera, with electronic switching of the video signals.



■ **Fig. 40.17**
(Continued)

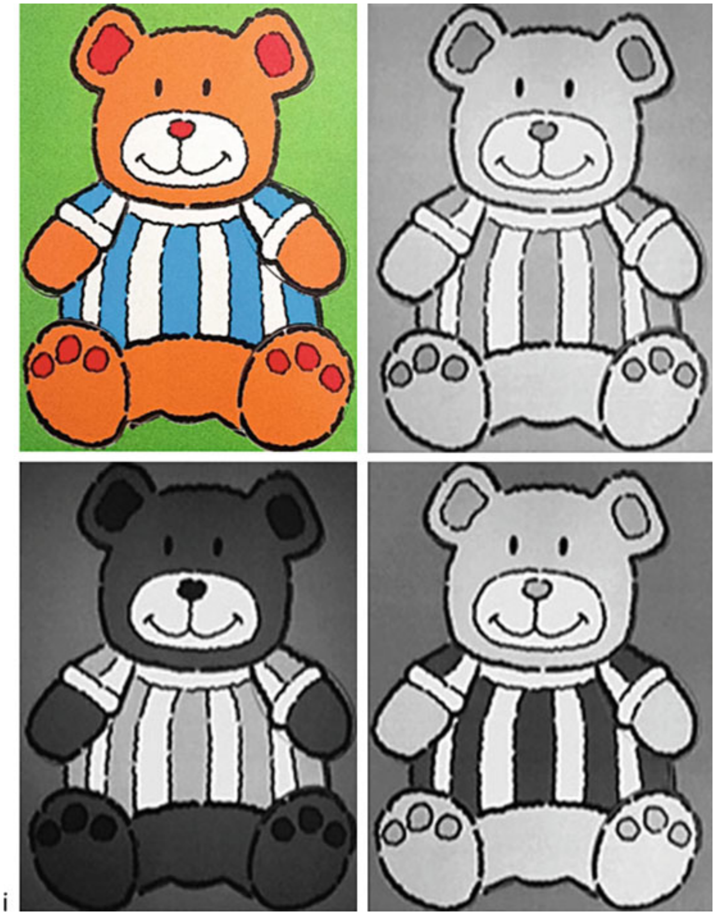


Fig. 40.17

Colour filtering (a) Interference filters, but not coloured glass/gelatine filters, can be tuned (i.e., λ_0 is changed slightly), by altering the tilt angle, θ. (b) Dichroic filters. The angle of incidence is fixed (45°). The incident beam is split into two components (transmitted and reflected beams), with complementary spectra. (c) Narrow band-pass filter. A family of filters is available, spanning the whole near-UV, VIS and near-IR spectrum. (d) Line-pass filter, transmits light within a very narrow range of wavelengths (monochromatic light, from a laser, sodium lamp, etc.), while blocking others. Such a filter can be used in conjunction with monochromatic lighting, in a system operating in bright ambient lighting. (e) Line-block filter, used to attenuate monochromatic light within a panchromatic spectrum. (f) Wide band-pass filter. Again, a range of filters covers the whole near-UV, VIS and near-IR spectrum. (g) Long-pass filter. The cut-off frequency, λ_1 may lie anywhere in the near-UV, VIS and near-IR spectrum. A UV-blocking filter has $\lambda_1 \approx 0.4\ \mu$m. If $\lambda_1 \approx 0.7\ \mu$m, the filter passes IR but not VIS. (h) Short-pass filter. An IR-blocking filter has $\lambda_2 \approx 0.7\ \mu$m. If $\lambda_2 \approx 0.4\ \mu$m, the filter passes UV but not VIS. A long-pass and short-pass filter can be combined, to create a band-pass filter which accepts a wider range of wavelengths than the fixed band-pass filters described in (f). (i) Example *Top left:* Sample images: child's toy. *Top right*: No filter. *Bottom left*: Short-pass filter ($\lambda_2 = 0.55\ \mu$m). *Bottom right*: Long-pass filter ($\lambda_1 = 0.55\ \mu$m)

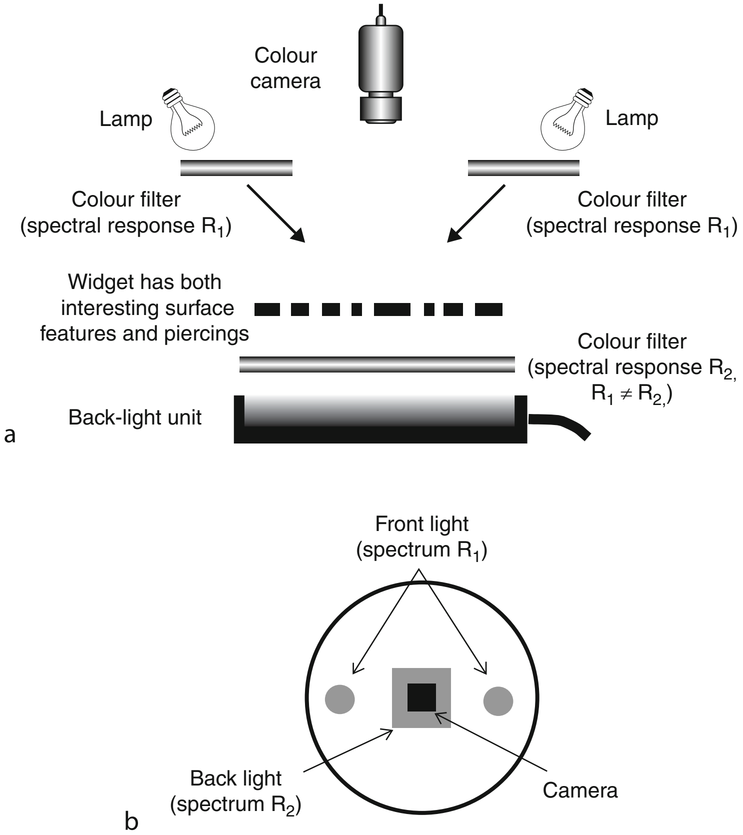

Colour camera
Lamp
Lamp
Colour filter (spectral response R_1)
Colour filter (spectral response R_1)
Widget has both interesting surface features and piercings
Colour filter (spectral response $R_{2,}$ $R_1 \neq R_{2,}$)
Back-light unit
a
Front light (spectrum R_1)
Back light (spectrum R_2)
Camera
b

Fig. 40.18

Combining front and back illumination. (a) Optical layout. (b) Illumination diagram. This is an unusual form of the diagram, since it describes both front and back illumination

REFERENCES Adjustable Field Splitter, a multi-arm articulated periscope. Light Works, Toledo. US Patent 5,975,710. http://www.lw4u.com

DIAGRAMS AND SAMPLE IMAGE ➔ *Figure 40.19*

40.22 Method 22: Conical Mirrors View an Annulus

OBJECTIVE To view an annulus; the central disc is removed from view by the use of anamorphic optics.

TYPICAL APPLICATIONS Inspecting all kinds of circular components/features, e.g., tops/rims of bottles, jars, cans, drinking glasses, circular holes, tyres, etc.

EQUIPMENT Custom optical component in the form of a frustrum, with a conical hollow in its base. The cross-section is shown in ➔ *Fig. 40.20a*.

CAMERA Array.

LENS Standard.

MOTION Static.

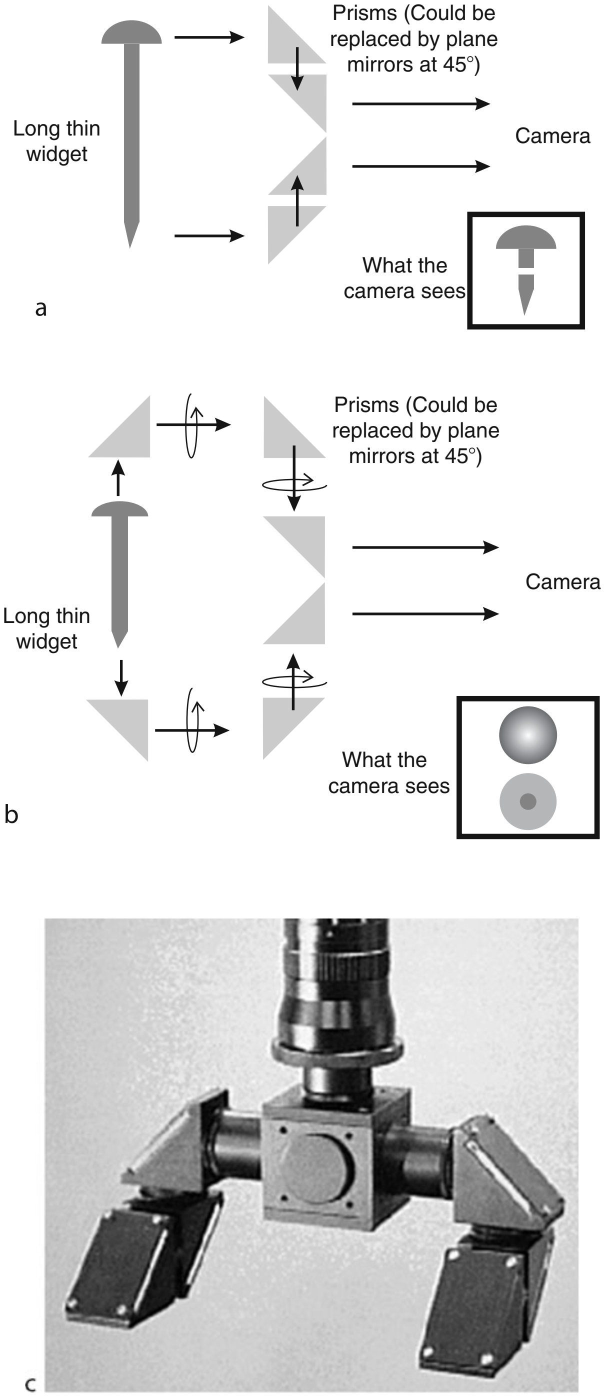

Prisms (Could be replaced by plane mirrors at 45°)
Long thin widget
Camera
What the camera sees
a
Prisms (Could be replaced by plane mirrors at 45°)
Long thin widget
Camera
What the camera sees
b
c

Fig. 40.19 (Continued)

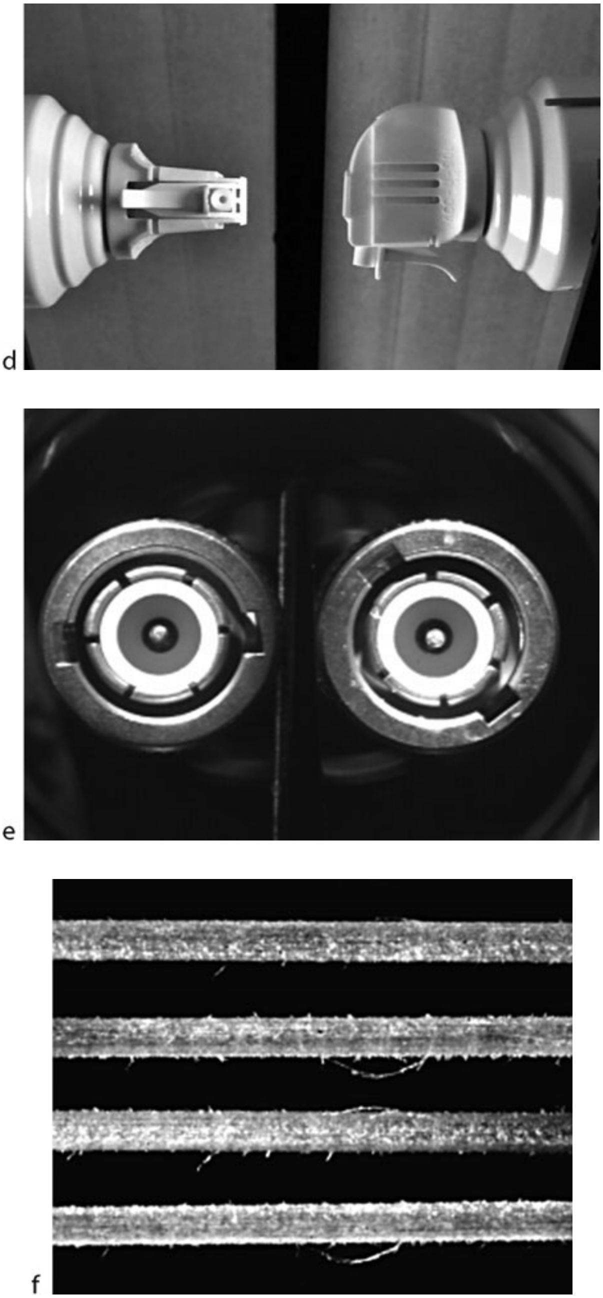


Fig. 40.19

Field splitter. (a) Using four prisms to ‘‘remove’’ the centre from the image of a long thin widget. (b) Viewing along orthogonal optical axes. (c) Adjustable Field Splitter, a proprietry product of Light Works, LLC. (d) Orthogonal views of a plastic bottle. (e) BNC connector, viewed simultaneously at opposite ends. (f) Four views taken around a fine fibre. [Images kindly supplied by Spencer Luster, Light Works, Inc., http://www.lw4u.com]

GEOMETRY Annulus.

REMARKS The optical layout and notation are explained in ➋ *Fig. 40.20*. The camera should be fitted with a lens with a large *f*-number, to reduce aberrations. To understand this method, we use polar co-ordinates. The point (r, θ) ($r \geq R$) in the object plane, maps to the point (r-R, θ) in the image plane. Points in the object plane that lie within the central disc of radius R are all "removed" by the conical lens. This process may be pictured in the following way: Imagine that the scene being viewed is painted on a series of threads laid radially in the object plane. Now, imagine that each thread is pulled inwards a distance R towards the centre point (The thread collecting at the centre is cut off and ignored.). The new image formed by the paint on the remaining portions of the threads is exactly the same as that seen by the camera. This method is based in the same principle as the periscope. For a cone whose semi-angle is 45°, the number of pixels is reduced by 25%. If a thin annulus whose internal and external radii are R and (R + δ) ($\delta < < R$) is to be examined, the field of view of the camera can be reduced to a square of size $[2\ W]^2$. The number of pixels to be digitised and processed is then reduced by $100.[1 - (\delta/(R + \delta))^2]$ percent, which represents a very considerable saving. By reversing this optical set-up, we can generate an annulus of light from a collimated beam.

LAYOUT DIAGRAM ➋ *Figure 40.20*

40.23 Method 23: Continuously Moving Web

OBJECTIVE To examine a continuously moving web for creases, scratches, splits, cuts, stains, embedded foreign bodies, etc. The web may be opaque or transparent. In the latter case, there may be limited rear access, making it impossible to place even a miniature camera or lights behind the web.

TYPICAL APPLICATIONS In-process inspection of a wide variety of web products. The web may be plain or printed. Similar lighting-viewing methods can be used on flat rigid panels, such as metal sheet, plaster-board, printed circuit boards, etc.

EQUIPMENT Depending on the web's optical properties (transparency/opacity, surface gloss, presence/absence of piercing, embossing, etc.), various items of equipment are needed:

(a) For coaxial illumination and viewing of opaque film: diffusing filter and beam-splitter
(b) For diffuse front illumination of opaque film: diffusing filter, beam-splitter, diffusing screen
(c) For combined front- and back-illumination of transparent film, or web with piercings: retroreflector
(d) For dark-field or back-illumination of transparent film, or opaque web with piercings: broad-face light source

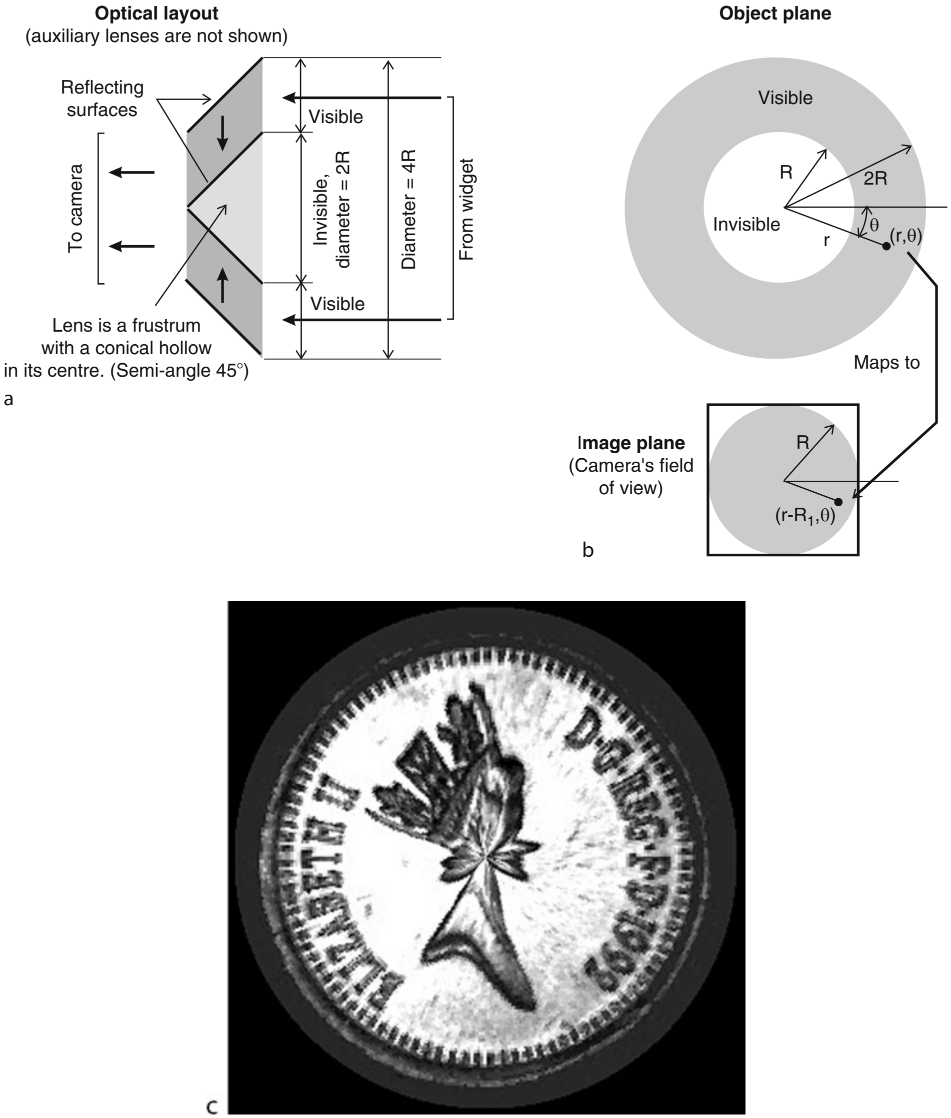
CAMERA Line-scan or array-scan.

MOTION Continuous linear motion.

GEOMETRY Flexible web held taut between rollers, or flat rigid material which may be in continuous form or discrete sheets.

MATERIALS Paper, card, metal film, clear or opaque plastic, glass, multi-layer laminate sheeting, knitted or woven materials. These may be transparent or opaque, shiny, matt or have a sheen.

REMARKS Several different lighting-viewing methods are illustrated in figure. Rolled, semi-fluid dough, clay, etc. may be inspected in a similar way. Retro-reflective sheet/ribbon can also be purchased from automobile spares retailers.



■ **Fig. 40.20**

Inspecting an annulus, without storing and processing unwanted pixels in the central region. (a) Optical arrangement, explaining how the image is formed. The white disc in the object plane, labelled "Invisible" vanishes. Each point (r, θ), within the grey annulus maps into a point (r-R, θ) in the image plane. (b) Sample image (simulated using QT, Chap. 21). *Left:* Original image of a coin. *Right:* Central region removed

REFERENCES Scotchlite (retro-reflective film. 3M Company, St. Paul. Wikipedia, http://en.wikipedia.org/wiki/Scotchlite. Accessed 4 Feb 2011
LAYOUT DIAGRAMS ➔ *Figure 40.21*

40.24 Method 24: Crack Detection (Ferrous and Non-ferrous)

OBJECTIVE To view cracks in ferrous or non-ferrous components.
TYPICAL APPLICATIONS Detecting cracks in metal forgings, castings, plastic mouldings, ceramics, glassware, etc.
EQUIPMENT UV lamps, UV- and VIS-blocking filters and fluorescent "ink". See below for details.
MATERIALS Ferrous or non-ferrous.
SAMPLE PREPARATION There are three distinct methods in use:

Ferrous widgets (Magnetic-Fluorescent Particle Method)

1. Magnetise the widget. This might be achieved in a solenoid, or simply by passing a large electric current (e.g., 1,000 Amp) through the widget
2. Immerse the widget in a suspension containing very small particles that are both magnetic and fluorescent (Often referred to as "ink.") The ink particles adhere around the crack, where there is a high remnant field
3. Rinse the component gently (This removes most ink particles from the widget surface but does not dislodge them from around the crack, where there is a large remnant magnetic field)
4. Illuminate the widget using ultra-violet light
5. Observe the fluorescence in the visible waveband (It is important that we prevent ultra-violet light from entering the camera and spoiling the image, by using a UV-blocking filter). Cracks are visible as bright streaks

Non-ferrous widgets (Fluorescent Dye Penetrant Method)

1. Immerse the widget in low-viscosity dye. This should be chosen so that it can penetrate into the crack quickly, by capillary action
2. Dry the widget
3. Illuminate the widget using ultra-violet light
4. Coat the component in fine white powder, chosen so that it will "draw" the dye out of the cracks by capillary action
5. Illuminate the coated widget using ultra-violet light
6. Observe the fluorescence in the visible (VIS) waveband

In a slightly safer variant of this method, blue illumination is provided and fluorescence is observed in the orange-red part of the VIS spectrum.

Non-ferrous widgets (Visible Dye Penetrant Method)

As above, except that visible light (VIS) is used and a vividly coloured dye.

REMARKS Safety is a major concern when people are involved in either the magnetic-particle or dye penetrant methods. Prolonged exposure to even modest levels of UV radiation can induce skin cancer and damage eyes, by causing cataracts. Special attention must be paid to protecting the hands, arms, neck, chest and face. In addition, the ink may be based on noxious or inflammable chemicals, such as kerosene. It contains minute particles of iron oxide, which

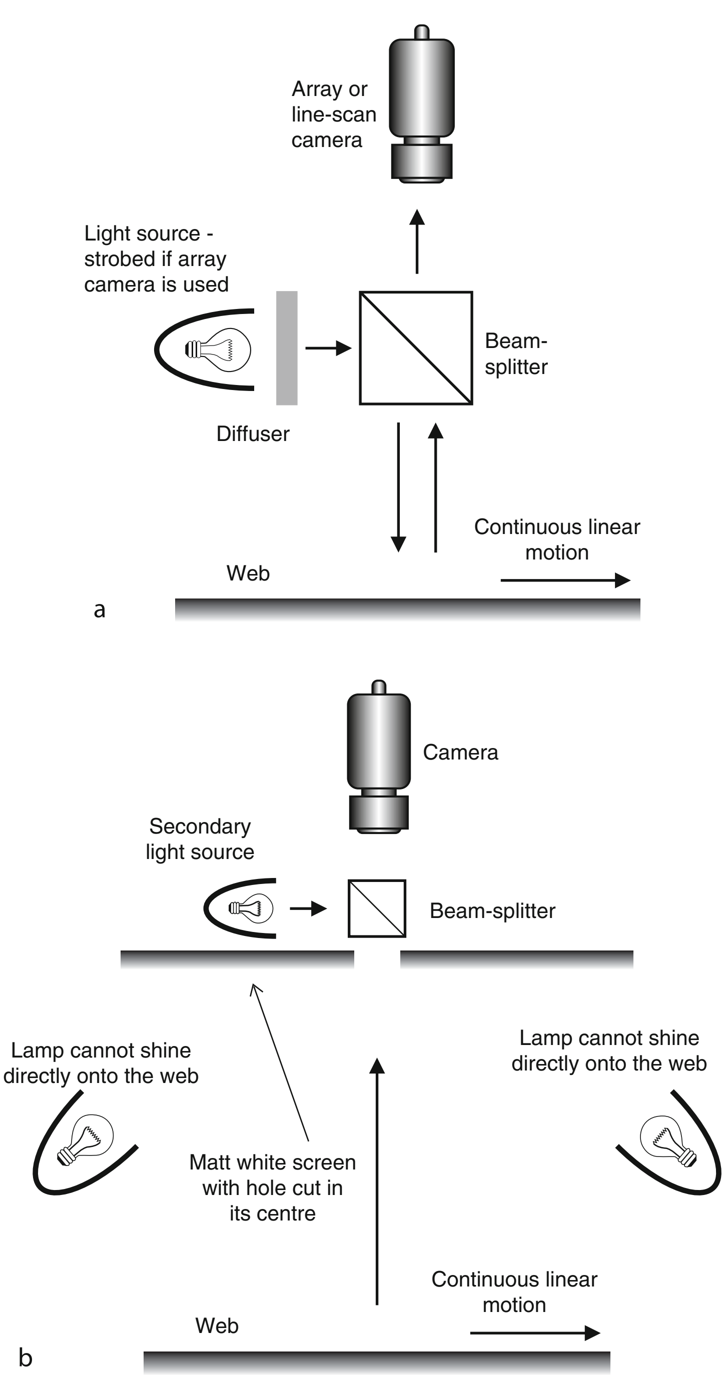


Fig. 40.21
(Continued)

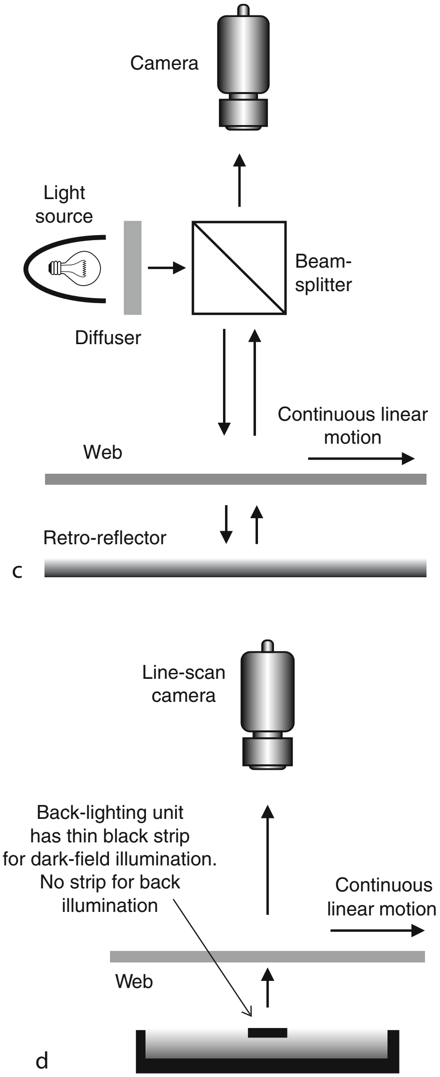


Fig. 40.21

Inspecting continuously moving web materials. (a) Optical layout for coaxial illumination and viewing of an opaque web. It may be necessary to offset the optical system slightly, so that reflections from the surface of the film can be avoided. This method is highly sensitive to undulations in the film surface and is therefore able to obtain good images of embossed patterns, as well as changes in surface reflectance. (b) Optical arrangement for diffuse front illumination of an opaque web. This is less sensitive to embossing and other surface undulations than the previous method. (c) Back illumination when there is limited rear access to the web. (d) Optical layout for dark-field and back-illumination of a transparent film

can cause skin irritation. Since only very low light levels are produced by fluorescence, a human inspector has be dark adapted. This may necessitate the inspector working in a small darkened and noisy booth, which is detrimental to his mental health. The health risks and unreliable performance achieved by human inspectors, together provide powerful motivation for automating this task, Image processing algorithms based on grey-scale morphology are often appropriate.

REFERENCES

- Batchelor BG, Hill DA, Hodgson DC (eds) (1985) Automated visual inspection. IFS, Bedford. ISBN 0-903608-68-5. Also published by North Holland, Amsterdam, 1985, ISBN 0-44487577-98, p 159
- Batchelor BG, Cotter SM (1983) Detection of cracks using image processing algorithms implemented in hardware. Image and Vision Computing 1(1):21–29

40.25 Method 25: Crossed Linear Polarisers Reduce Glinting

OBJECTIVE To reduce the effects of glinting, when examining shiny objects.

TYPICAL APPLICATIONS Examining painted and lacquered objects.

EQUIPMENT Two linear polarisers. Their easy axes are oriented at 90° to each other (Often referred to as *crossed polarisers.*).

CAMERA This method is well suited for use with cameras that suffer from optical overload. It is useful for both CRT and solid-state cameras. The latter may be line- or array-scan types.

GEOMETRY Suitable for objects with a complex shape that are difficult to accommodate by adjusting the geometry of the light source.

MATERIALS Bright shiny surfaces, such as polished metal, smooth plastic, glass, glazed ceramics, painted/lacquered surfaces, wet and oily objects.

REMARKS This method is based on the fact that specular reflection (produces glinting on a shiny surface) retains linear polarisation, whereas diffuse reflection does not. A shiny surface illuminated with light polarised at θ° reflects light that is also polarised at θ°. On the other hand, a diffuse reflector emits light that is almost uniformly polarised through all angles: 0–180°. Glinting is a pernicious problem in Machine Vision, as it frequently spoils image quality. Highlights due to glinting can, for example, give a false impression about the position of an object's outer edge. Glinting can also cause local optical overload of the camera, leading to:

(a) *Burn-in* (This is a problem in some older CRT cameras but is less important in solid-state cameras.)
(b) *Comet-tail effect* (Again, this is a problem for some CRT cameras but not with solid-state cameras)
(c) *Blooming* (This occurs in some solid-state and CRT sensors)

Some polarising filters do not function well in the near infra-red wave-band (Recall that the peak in the emission-spectrum of incandescent lamps and the peak in the sensitivity of solid-state image sensors are both in the NIR band). For this reason, it may be necessary to add an IR-blocking filter, to optimise image contrast. Low-cost plastic-sheet polarisers render bright highlights as spots of deep colour (e.g., violet, or red). To improve image quality still further, it may be advantageous to insert an appropriate optical filter between the polariser and the camera.

LAYOUT DIAGRAM AND SAMPLE IMAGES ➋ *Figure 40.22*

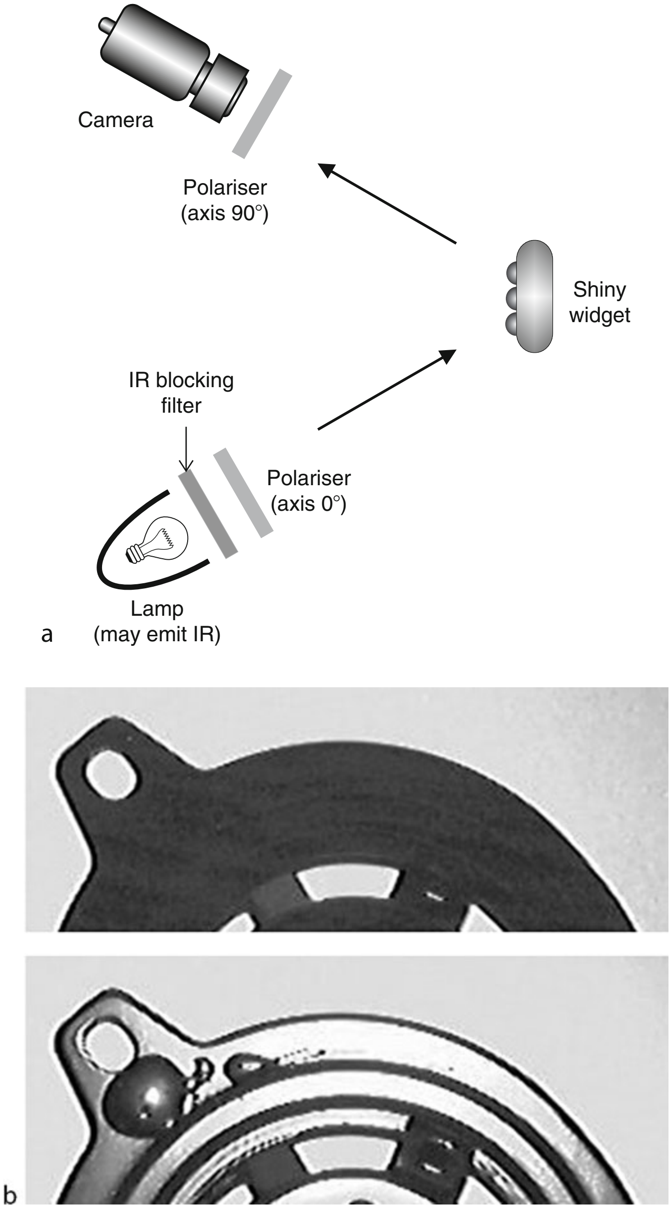


Fig. 40.22

Crossed polarisers eliminate glinting. (a) Optical layout. (b) Sample images: painted metal pressing. The paint is black, smooth and shiny. The back-ground (*white copier paper*) provides a good diffusing surface. *Top:* With polarisers in place, glinting is suppressed. *Bottom:* With no polarisers present, glinting occurs. Notice the highlights on the ''dollops'' of adhesive (*black, smooth and shiny*), near the round hole

40.26 Method 26: Dark Field Illumination (Array Camera)

OBJECTIVE To view silhouettes of transparent objects.
TYPICAL APPLICATIONS Shape analysis of glass and plastic bottles, jars, tumblers, etc.
EQUIPMENT Light box, with opaque black mask.
MATERIALS Transparent/clear glass, plastic. Transluscent materials.
CAMERA Array.
MOTION Static.
MATERIALS Transparent/clear glass, plastic. Transluscent materials.
REMARKS This method produces far better results than back-lighting on some transparent/transluscent objects, such as glass and plastic containers. When there is no object present, the camera receives no light at all, as it is facing the mask. A bottle, or jar, placed in the optical path receives light, which is diverted within and around the side walls, by refraction and internal reflection. Light therefore emerges to reach the camera, from those parts that are not illuminated directly from behind. Vertical edges and sudden variations in wall thickness are high-lighted particularly well. Hence, embossing, cracks and features such as "bird-swings" (filament of glass hanging between the side walls, across the inside of a bottle) and "spikes" ("stalagmite" of glass protruding inwards from the base or side-wall of a jar) are all highlighted by this method.
LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.23*

40.27 Method 27: Dark Field Illumination (Line-Scan Camera)

OBJECTIVE To view silhouettes of transparent objects.
TYPICAL APPLICATIONS Shape analysis of glass and plastic bottles, jars, tumblers, etc.
EQUIPMENT Light box, with thin black strip placed across its centre.
CAMERA Line scan.
MOTION Linear movement.
MATERIALS Transparent/clear glass, plastic. Transluscent materials.
REMARKS This method produces far better results than back-lighting on some transparent/transluscent objects, such as glass and plastic containers. When there is no object present, the line-scan camera receives no light at all, as it is facing the mask. A bottle, or jar, placed in the optical path receives light, which is diverted within and around the side walls, by refraction and internal reflection. Light therefore emerges to reach the camera, from those parts that are not illuminated directly from behind. Vertical edges and sudden variations in wall thickness are high-lighted particularly well. Hence, embossing, cracks and features such as "bird-swings" (filament of glass hanging between the side walls, across the inside of a bottle) and "spikes" ("stalagmite" of glass protruding inwards from the base or side-wall of a jar) are all highlighted by this method.
LAYOUT AND ILLUMINATION DIAGRAMS ➋ *Figure 40.24*

40.28 Method 28: Detect Pimples, Pits and Changes of Refractive Index

OBJECTIVE

- To detect and count liquid, or solidified, drops of a clear liquid lying on a clear transparent film or plate
- To detect sudden changes in refractive index in a smooth flat, or nearly flat, plate

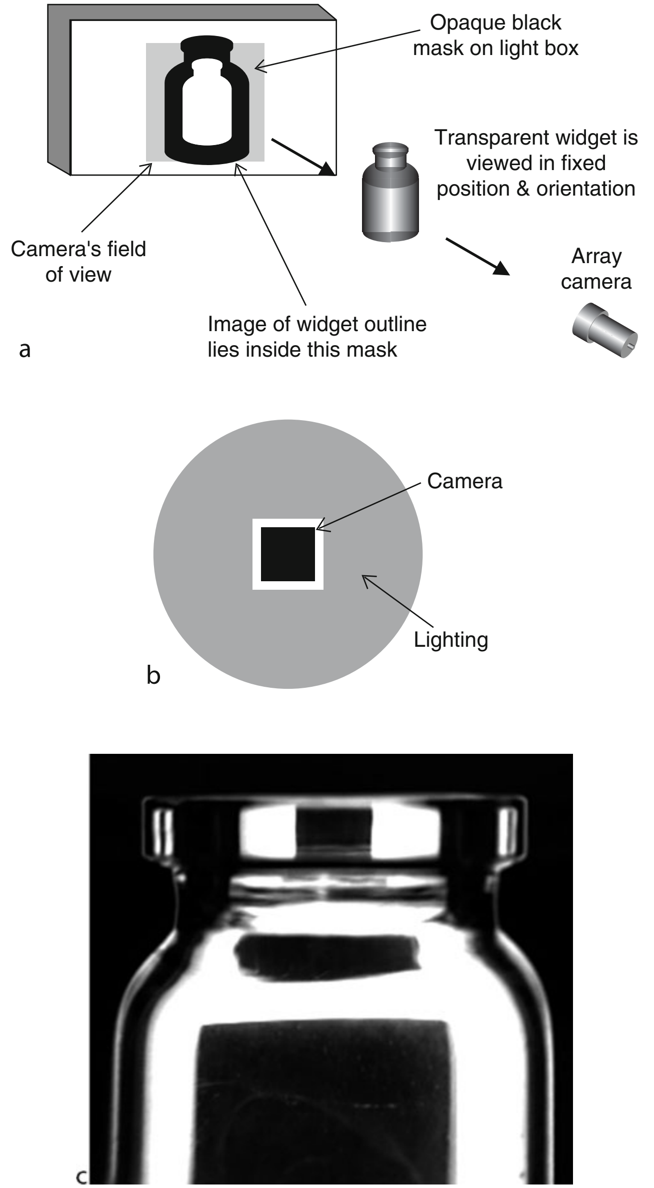


Fig. 40.23

Dark-field illumination using an array-scan camera. (a) Optical arrangement. (b) Illumination diagram (square mask). (c) Sample image: small glass vial

TYPICAL APPLICATIONS

- Inspecting plastic film
- Inspecting moulded glass
- Inspecting bi-focal lenses

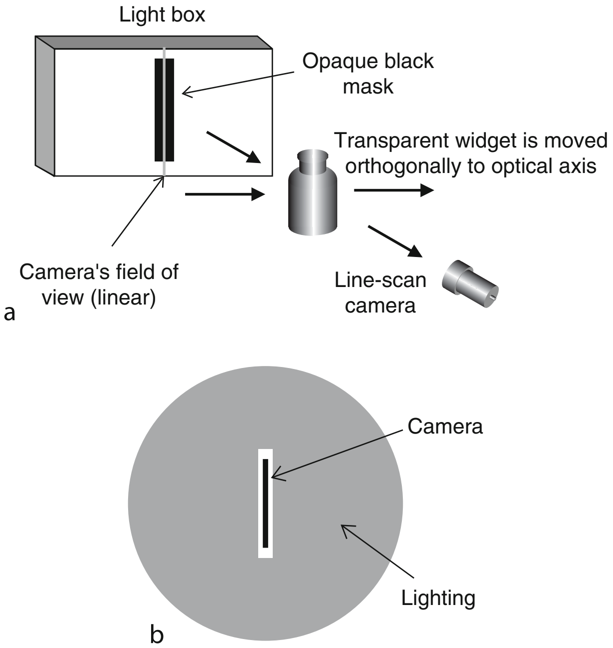

Light box
Opaque black mask
Transparent widget is moved orthogonally to optical axis
Camera's field of view (linear)
Line-scan camera
a
Camera
Lighting
b

Fig. 40.24
Dark-field illumination using a line-scan camera. (a) Optical arrangement. (b) Illumination diagram

EQUIPMENT Back-light unit, fitted with fine striped mask (line-scan camera) or checked mask (array-scan camera).
CAMERA Line-scan or array.
LENS Close focus.
MOTION Continuous linear motion for a line-scan camera. Static for an array type.
REMARKS This method detects changes in the slope of the surface. (i.e., large values in the second derivative of the film-thickness function) It is not able to produce highly accurate measurements but it does allow rapid identification and counting of drops of water and oil on clear glass and plastic sheeting. Pimples formed by solidified drops of plastic on sheeting can also be detected in this way, as can shallow pits with well-defined edges. This method works on the principle that a sloping surface deflects a light beam by refraction. If the slope changes suddenly, then the image of the background will appear distorted. Sudden changes in the refractive index can also be detected. ➲ *Figure 40.25d* It is a relatively simple matter to identify distortions and breaks in the continuity of the lines formed in the image. If a chess-board mask is used, the periodicity of the pattern is altered in one or both dimensions.
LAYOUT DIAGRAM AND SAMPLE IMAGE ➲ *Figure 40.25*

40.29 Method 29: Detecting Particles in a Swirling Liquid

OBJECTIVE To detect small particles (foreign bodies) in suspension in a clear liquid.

TYPICAL APPLICATIONS Detecting fibres, small pieces of glass, etc. in medicine vials.

EQUIPMENT

- Mechanical equipment to rotate the vial
- Dark-field illumination

CAMERA Array.

IMAGE ACQUISITION

(a) Rotate the vial at high speed

(b) Stop it suddenly

(c) Digitise an image. (Call this *I1*.)

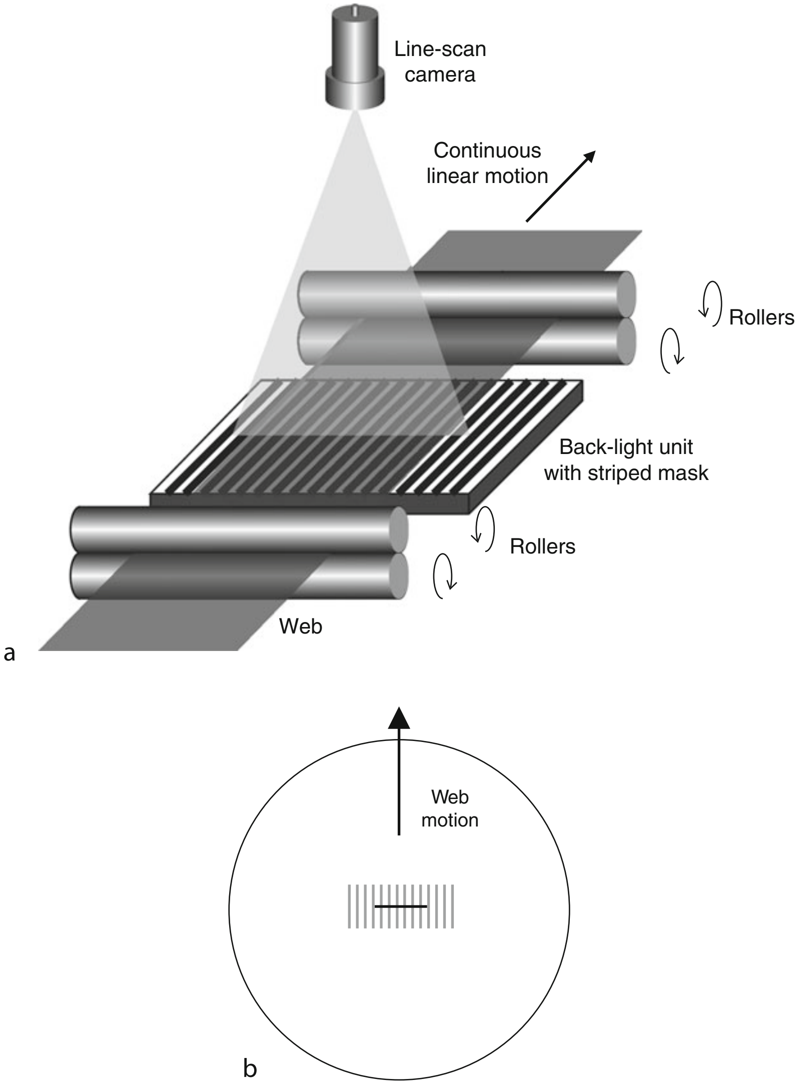


Fig. 40.25
(Continued)

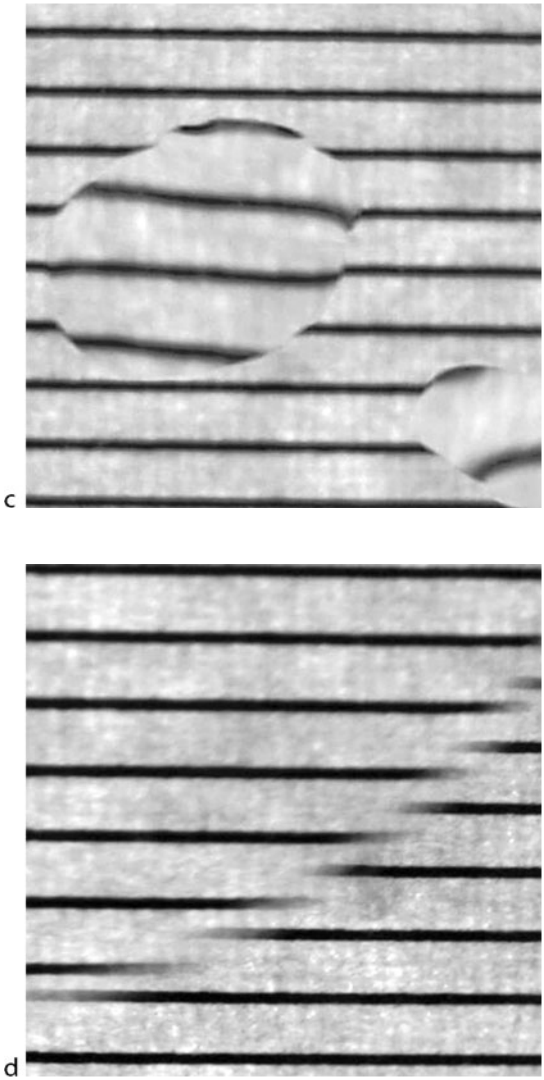


Fig. 40.25
Detecting geometric distortions and changes of refractive index on a transparent film (a) Optical layout. (b) Illumination diagram for the set-up based on a line-scan camera. (c) Sample image: Drops of water on a microscope slide. (To create this image, the back-ground consisted of an array of fine lines printed on ordinary white paper. In practice, a clear subtrate is preferable.) (d) Edge of the D-shaped short-focus lens element for bi-focal spectacles

(d) Wait for a short time
(e) Digitise an image. (Call this *I2*.)

MATERIALS AND GEOMETRY Clear liquid in a cylindrical glass or plastic container.

REMARKS This method was designed to detect small particulate foreign bodies that are denser than the liquid within the vial and which therefore naturally sink to its bottom, Small particles are

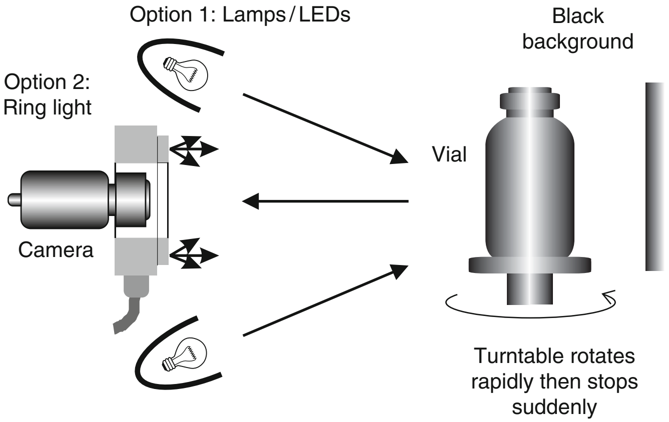


Fig. 40.26
Detecting particles in a swirling liquid. Notice that there are two lighting options shown here

not easily detected in the static liquid, when they are resting on the bottom or sides of the vial, since they cannot be distinguished from minor irregularities in the wall of the container. The vial is spun rapidly, causing the liquid to rotate with it. The liquid continues to swirl around for a few seconds after the vial has been stopped. By digitising two images in quick succession, when the particles are in different positions, it is possible to detect their motion. Images *I1* and *I2* are subtracted. Any shadows caused by imperfections in the wall of the vial are common to both images and hence are eliminated by the subtraction process. On the other hand, each moving particle creates one bright spot and one nearby dark spot in the difference image. Since the speed of rotation of the liquid is known approximately, these spots can be linked together, thereby allowing a more robust detection algorithm to be formulated. Vibration of the vial may be used instead of spinning but this does not allow pairs of spots to be linked.

REFERENCES Batchelor BG, Hill DA, Hodgson DC (eds) (1985) Automated visual inspection. IFS, Bedford. ISBN 0-903608-68-5. Also published by North Holland, Amsterdam, 1985, ISBN 0-44487577-98

LAYOUT DIAGRAM ➲ *Figure 40.26*

40.30 Method 30: Diffuse Coaxial Illumination

OBJECTIVE To provide diffuse illumination, with the camera effectively “looking through” the light source.

TYPICAL APPLICATIONS Coins, inspecting embossed and engraved surfaces.

EQUIPMENT Beam-splitter, diffusing screen. Circularly polarising filter (optional).

CAMERA Array or line scan.

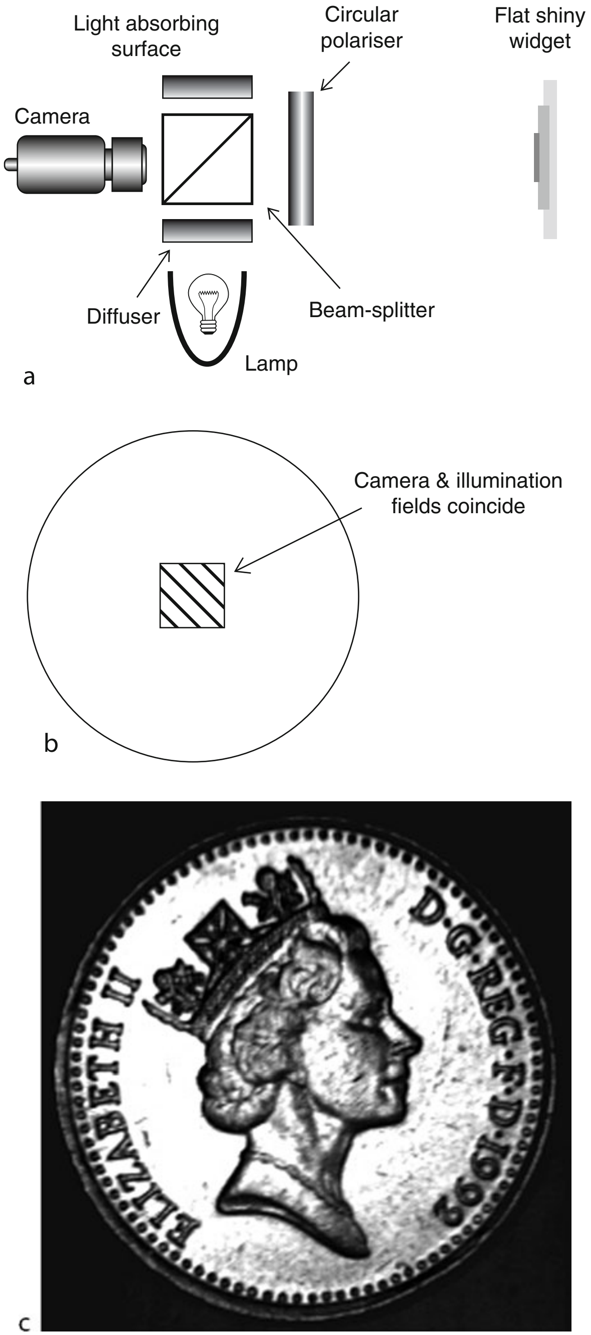
MOTION Static or linear motion.

GEOMETRY Flat with embossing or engraving.

MATERIALS Bright shiny metal.

REMARKS This is a very effective technique for viewing coins and similar objects. The flat surface (called the “table”) is bright and the embossing is dark.

LAYOUT DIAGRAM AND SAMPLE IMAGE ➲ *Figure 40.27*



◘ Fig. 40.27

Coaxial Diffuse Illumination. (a) Optical layout. (b) Illumination diagram. (c) Sample image: Coin

40.31 Method 31: Diffuse Front Illumination

OBJECTIVE To reduce the effects of glinting.
TYPICAL APPLICATIONS Inspecting objects with shiny, nearly planar surfaces, to detect scratches, cracks, pits, embossing, engraving, etc. shiny objects. Suitable for CDs, coins, electrical connectors, printed circuit boards, smooth plastic, glass, porcelain, etc.
EQUIPMENT Bright flood lamps (e.g., incandescent filament or fluorescent illuminator with a high-frequency driver) Diffusing screen with viewing aperture. This consists of a flat matt white board that has a small cut in its centre. It may be necessary to fit a short tube (painted matt black on its inside surface) into this hole, to prevent light reaching the camera directly from the lamps. A secondary light source and beam-splitter may be used to compensate for the dark aperture in the diffusing screen.
GEOMETRY Mirror-like surfaces and nearly flat surfaces with embossing, engraving, etc. Can also be useful for more complex structures, since it does not cast sharp shadows.
MATERIALS Bright metal, smooth plastic, glass, porcelain, etc., lacquered/painted surfaces.
REMARKS Excellent images of defects such as scratches can be obtained from mirror-like surfaces, which may be difficult to view otherwise. The secondary light source and beam-splitter avoid creating a reflection of the camera when viewing a mirror-like surface. In some applications, it is possible to eliminate them altogether.
LAYOUT DIAGRAM AND SAMPLE IMAGE ➲ *Figure 40.28*

40.32 Method 32: Dual Orthogonal-Beam X-ray Imaging

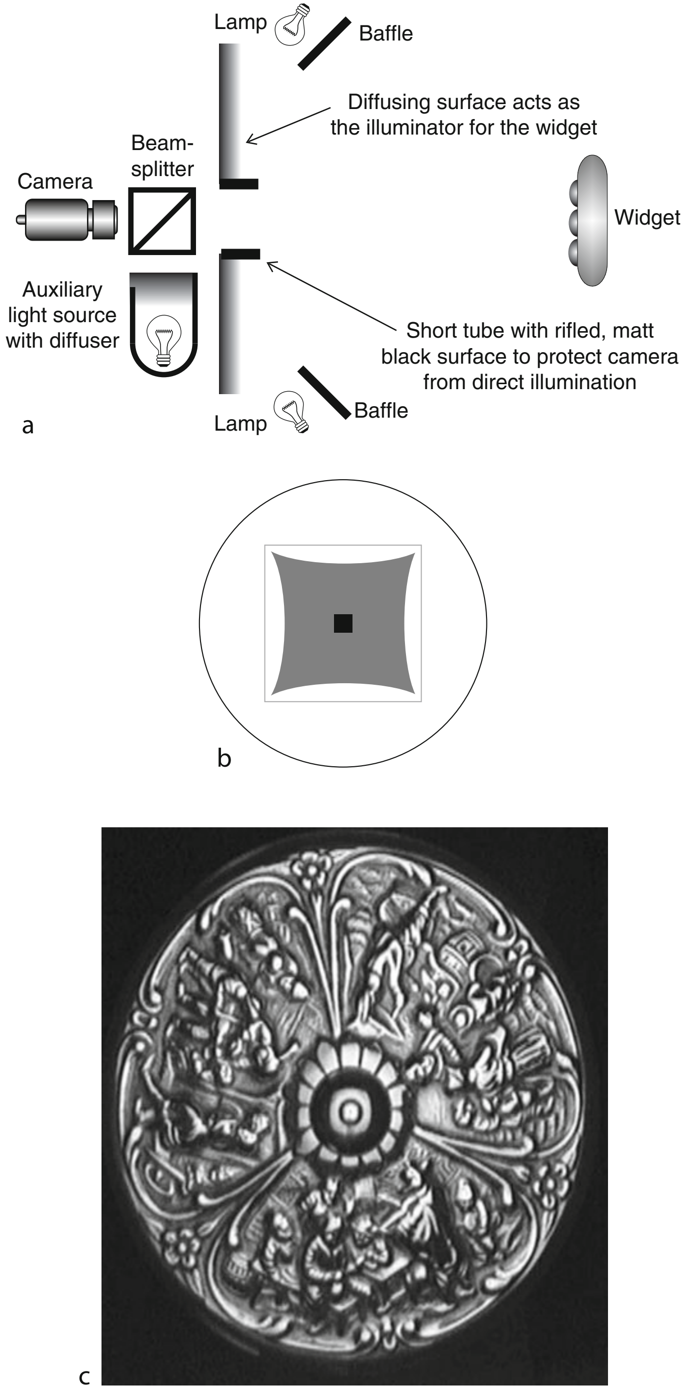
OBJECTIVE To inspect an object using two orthogonal X-ray beams.
TYPICAL APPLICATIONS Detecting foreign bodies in rigid packaged food products. The system is well suited to detecting contaminants in cartons containing multiple packages of food, as it can locate a foreign body in 3-dimensional space.
EQUIPMENT Two x-ray sources and two sensors.
CAMERA Two line-scan x-ray sensors.
MOTION Continuous linear motion.
REMARKS Several possible dual-beam x-ray systems were studied by simulation. The arrangement shown in ➲ *Fig. 40.29a* was found to be generally superior to the others. For example, the physical arrangement in which both x-ray beams are inclined at 45° to the horizontal generates images that are very difficult to understand and process.
REFERENCES

- TriAxis: three dimensional automated x-ray inspection system. AIR Grant, number CT93-10-54, Commission of the European Union. 1993–1997
- Graves M, Batchelor BG, Palmer SC (1994) 3D x-ray inspection of food products. In: Proceedings of the conference on applications of digital image processing XVII, San Diego, vol 2298**.** SPIE, Bellingham, pp 248–259. ISBN 0-8194-1622-3

LAYOUT DIAGRAM AND SAMPLE IMAGES ➲ *Figure 40.29*

40.33 Method 33: Dual-Wavelength Height Measurement

OBJECTIVES To provide a high precision method for measuring the height of a surface.


Lamp
Baffle
Diffusing surface acts as the illuminator for the widget
Beam-splitter
Camera
Widget
Auxiliary light source with diffuser
Short tube with rifled, matt black surface to protect camera from direct illumination
Lamp
Baffle
a
b
c

◘ Fig. 40.28

Diffuse Illumination. (a) Optical layout. (b) Illumination Diagram. (c) Sample image: top of an embossed, conical silver tobacco box

TYPICAL APPLICATIONS Measuring the form of machined components, such as aircraft turbine blades.
EQUIPMENT Various optical components, including lenses, mirrors, dichroic mirrors.
CAMERA Line scan.
REMARKS This is a variation of the standard light-stripe triangulation method of range measurement. (Methods 66, 68 and 107) The optical system illustrated in figure projects alternating stripes of IR and VIS light onto the widget surface. The light scattered from the widget is then resolved using a dichroic mirror and appropriate filters. There are two cameras which sense the red and IR stripes independently. In the cited work, measurement tolerances of about 50 μm were achieved.
REFERENCES Porter GB III, Mundy JL (1982) A non-contact profile sensing system for visual inspection. In: Proceedings of the conference on industrial applications of machine vision, research triangle, NC, 3–5 May 1982. IEEE Computer Society, cat no. 82CH1755-8, pp 119–129

40.34 Method 34: Endoscope for Inspecting Holes

OBJECTIVE To view inside a hole. Either the end or side walls can be examined.
TYPICAL APPLICATIONS Inspecting holes in machine parts, hydraulics cylinders, automobile/jet engines, short straight pipes/tubes, internal (female) threads,.
EQUIPMENT *Endoscope.* (Also called a *Borescope* or *Intrascope.*) Fine endoscopes are sometimes called *Needlescopes.*
CAMERA Array.

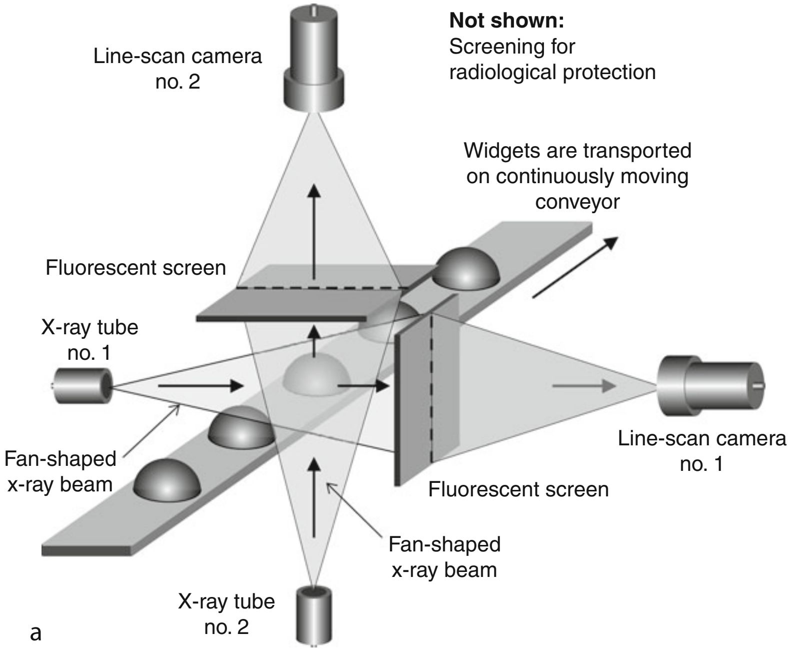


Fig. 40.29
(Continued)

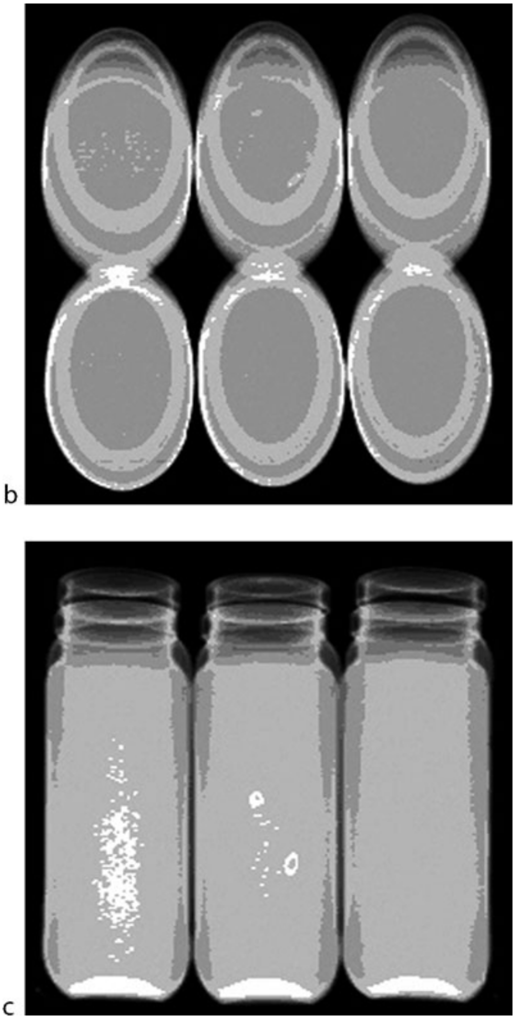


Fig. 40.29

Dual orthogonal-beam x-ray imaging. (a) Layout of the image acquisition system. (b) Sample images, displayed in negative form for easier viewing. six glass jars filled with savoury tomato sauce, were held on a card-board tray. *Top:* top view. *Bottom:* Side view. Three of the jars appear to be taller than the others because they are closer to the x-ray source (i.e., in a diverging beam). Some small foreign bodies present as bright spots. Notice that the top- and side-view images are aligned along the horizontal axis. *Images kindly supplied by Mr. Ralf Haberts, Pulsarr BV, Eindhoven Netherlands*

LENS Endoscope with C-mount fitting can be fitted in lieu of a standard lens.
MOTION Static.
GEOMETRY Holes, up to 1,000 mm deep. Longer coherent fibre optic bundles are expensive.
REMARKS The endoscope can be fitted with a mirror to allow it look backwards (mirror angle <45°), sideways (45°), or even forwards (>45°). The mirror can be rotated by hand, or

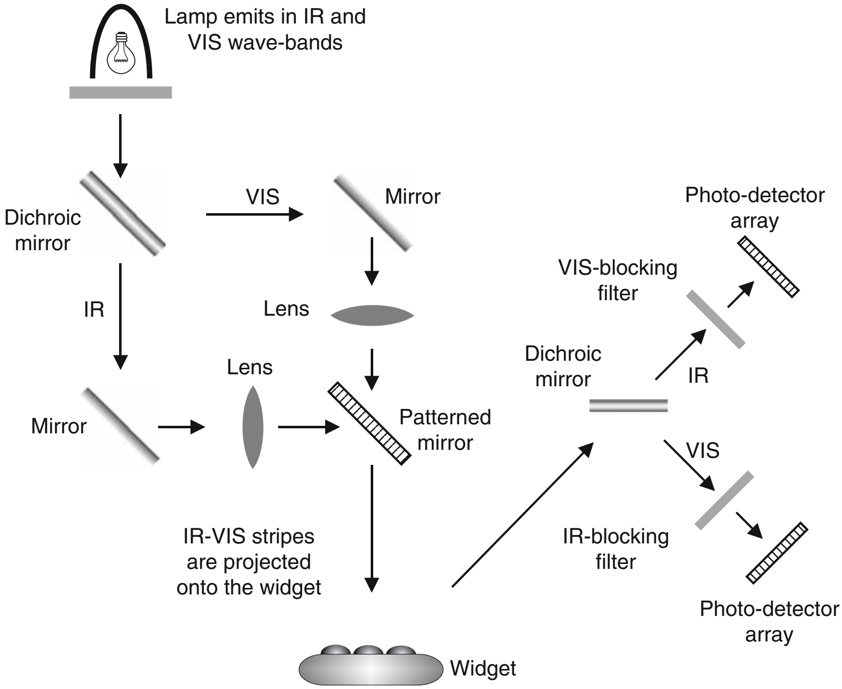


Fig. 40.30
Optical layout for dual–wavelength surface-height measurement

a motor. However, the image of features such as a ridge around the circumference of the bore also rotates. This can complicate the image processing needed. A circularly polarising filter can be added in front of the objective lens to eliminate problems due to glinting. The colour of the illumination can be altered by inserting a filter between the primary light source and the fibre-optic illumination bundle.

LAYOUT DIAGRAM AND SAMPLE IMAGE *Figure 40.31*

40.35 Method 35: Estimating Range During Image Capture

OBJECTIVE To focus a camera automatically and/or compensate for variations in the apparent size of a widget as its position relative to the camera changes.
TYPICAL APPLICATIONS Examining objects that cannot be held in a fixed position during image capture.
EQUIPMENT Two low-power, diode lasers.
CAMERA Array or line-scan.
MOTION Static or moving slowly.
GEOMETRY Has well defined flat region, roughly normal to the camera's line of sight.
MATERIALS Any material with a dispersive surface (i.e., not mirror-like).
REMARKS A simple range finder can be built using two lasers, or a single laser and a beam splitter. Two laser spots are projected onto the widget and their separation measured using the

camera and image processor. The range R can be estimated directly from the separation of the projected spots (S), using the formula:

$$R = R_0 + 2\,S.\cot(\theta)$$

where R_0 is the widget-camera distance when the spots converge. Notice that the sensitivity can be adjusted by changing the angle θ. The lasers can be switched off, if necessary, during image capture.

LAYOUT DIAGRAM ➔ *Figure 40.32*

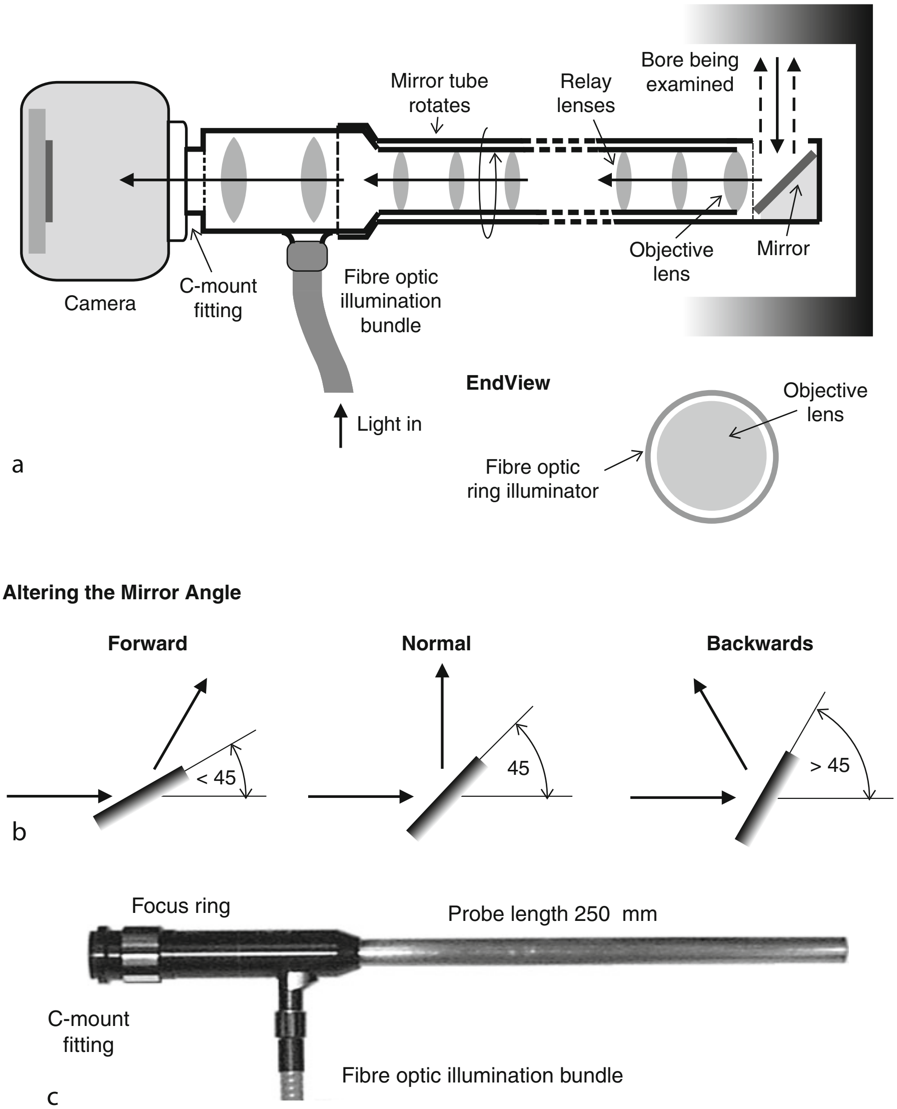


Fig. 40.31
(Continued)

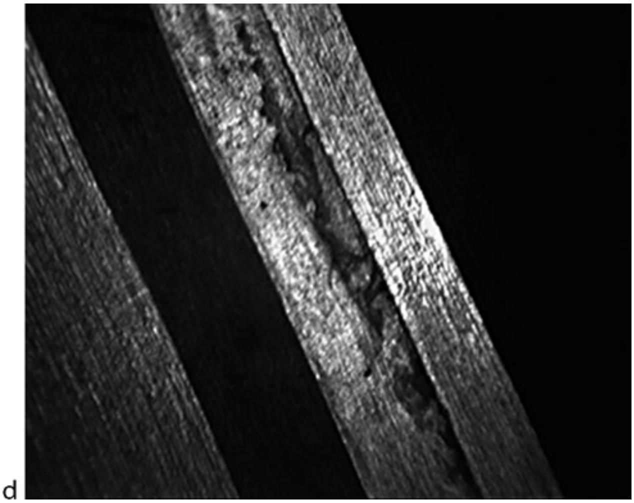


Fig. 40.31
Rigid endoscope. (a) Internal optical arrangement. Illumination is provided by optical fibres which lie parallel to the axis of the probe, between the relay-lens tube and the outer protective sheath. The mirror tube fits around the latter and can be rotated by hand or mechanically. (b) The angle of view can be adjusted by changing the angle of the mirror. (c) Industrial endoscope. (d) Sample image. Notice the variable intensity, due to the variable angle of the lighting on the widget surface. As the mirror tube is rotated, the diagonal feature (circumferential ridge inside the bore) also rotates

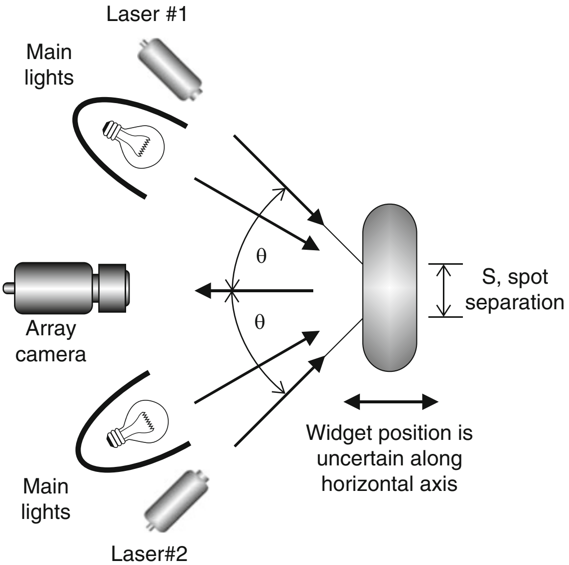


Fig. 40.32
Estimating range using two lasers to estimate the camera-widget distance

40.36 Method 36: Examine the Inside of a Pipe

OBJECTIVES To view inside a tube or pipe.
TYPICAL APPLICATIONS Inspecting sewers, drains, industrial pipework, etc. Inspecting nuclear installations is of particular interest.
EQUIPMENT Self-propelled probe, carrying one or two miniature cameras, LED lighting.
CAMERA Two miniature cameras.
LENS Light weight, miniature.
MOTION Self propelled, using electrical or pneumatic motors.
GEOMETRY Cylindrical or square bore of nearly constant diameter.
REMARKS The cavity being inspected must be reasonably dry and free from dust. In some applications, it may be possible to clean the lights and camera lens in situ, using an air purge. Since exposed solid-state devices are carried deep inside the cavity, its temperature must not exceed about 60°. Radiation levels must also be low. If the camera drags an "umbilical" cord behind it, carrying all power and video signals, the depth of penetration will be limited. On-board image storage will, of course, circumvent this restriction. Self-propelled inspection probes have been devised for pipe inspection using pneumatic and electric motors. The vehicle may travel on wheels, or "walk." Larger autonomous probes, some with on-board image storage and processing, have been in use for many years in the gas, water and petroleum industries and for inspecting tunnels, drains and sewers. In some situations, structured light (Methods 24 and 41) is used to examine the geometry of the cavity.
REFERENCES PT36-300 pipe inspection equipment. Ashtead Technology. www.ashtead-technology.com. Accessed 9 Feb 2011
DIAGRAM ➋ *Figure 40.33*

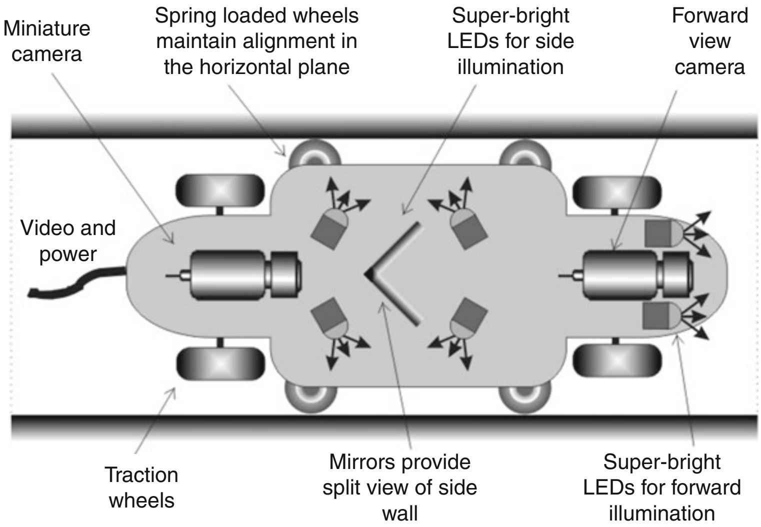

Miniature camera
Spring loaded wheels maintain alignment in the horizontal plane
Super-bright LEDs for side illumination
Forward view camera
Video and power
Traction wheels
Mirrors provide split view of side wall
Super-bright LEDs for forward illumination

Fig. 40.33
Examining the side-wall of a narrow tube using two sub-miniature CCD cameras, mounted on a motorised platform. LEDs provides the illumination. The "umbilical chord" carries the video signals, power for the cameras, LEDs and the motor. By substituting a conical mirror for the mirrors, almost 360° of the side wall can be examined. (See Method 116.)

40.37 Method 37: Fast Electronic Shutter

OBJECTIVE To provide electronically controlled sampling of an optical image, enabling a standard video camera to observe continuous or repeated motion, or fast transient events, without the use of stroboscopic illumination.

TYPICAL APPLICATIONS An electronic shutter can be used in a wide variety of situations, where stroboscopic illumination would normally be indicated but safety considerations prohibit its use.

EQUIPMENT Liquid crystal light valve, operating as an optical shutter. This may be external to, or an integral part, of the camera.

CAMERA Highly sensitive, progressive-scan, array camera.

LENS Small *f*-number, to collect maximum amount of light.

MOTION Continuous or repeated motion, or rapidly changing scene.

REMARKS Since this method uses continuous illumination, it is inherently safe. On the other hand, stroboscopic illumination presents considerable health and safety risks: inducing epileptic fits (even in people not previously known to be susceptible to epilepsy), triggering migraine and making moving/rotating machinery appear to be stationery. To achieve satisfactory operation of the camera (i.e., good signal: noise ratio), it might be necessary to apply very high levels of illumination. Since the output from the photo-sensor array is proportional to the product of the time that the shutter is "open" and the illumination intensity, observing short-duration events might require very bright lamps dissipating a lot of heat. Care should be taken to avoid excessive heating, if incandescent filament lamps are being used.

LAYOUT AND TIMING DIAGRAMS ❯ *Figure 40.34*

40.38 Method 38: Fibre Optic Image Conduit/Cross Section Converter

OBJECTIVES

- To bend and/or twist the light path when viewing in awkward locations
- To remove the camera from a harmful environment, protecting the camera from extremes of dirt, moisture, oil, heat, radiation, and vibration
- To provide a small and rugged alternative to a reducing lens system
- To convert x-rays or UV into visible light (VIS)

TYPICAL APPLICATIONS Viewing inside areas that are hot, chemically or biologically dangerous, radio-active, or under low/high pressure. (The image conduit is inserted in the wall of the work area.)

EQUIPMENT Fibre optic image conduit.

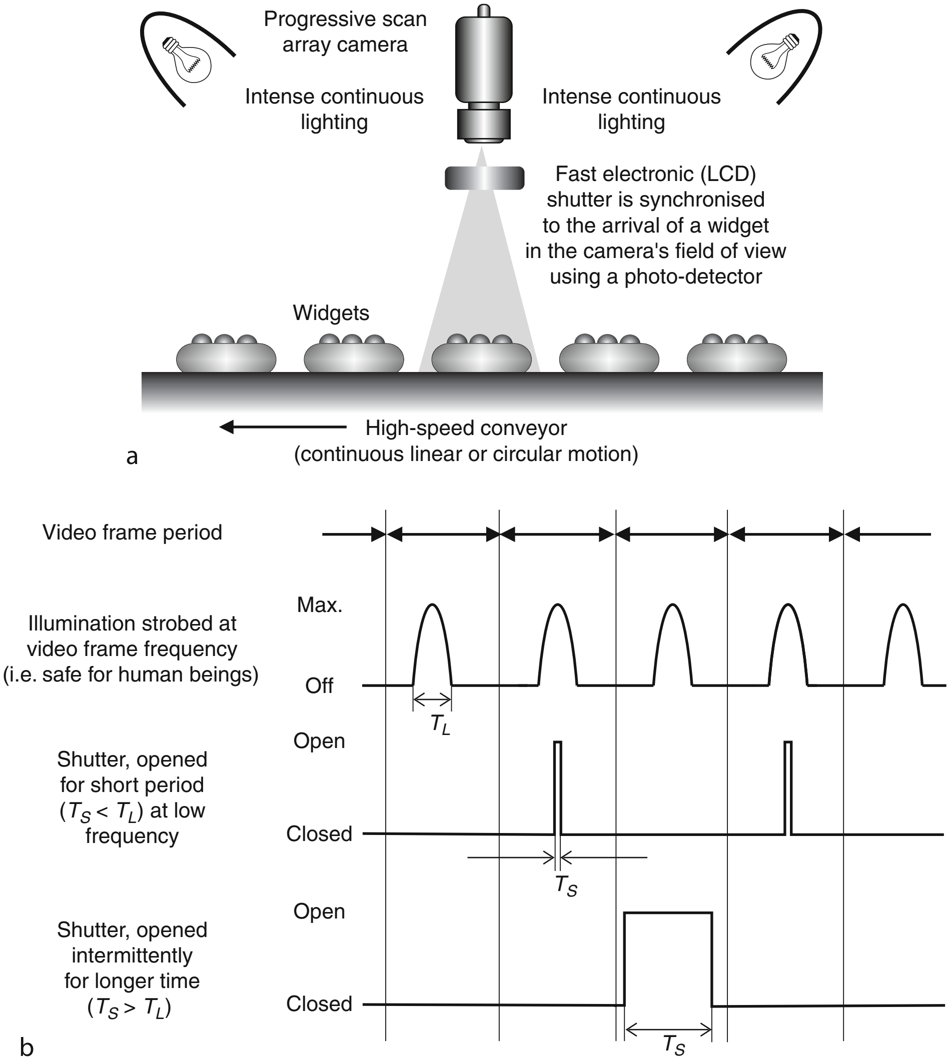
CAMERA Array scan.

LENS Standard lenses.

MOTION Static or dynamic

GEOMETRY Objects with holes.

REMARKS Fibre-optic image conduit consists of a large number of glass fibres, fused together to form a solid rod, or block. The conduit transmits an image projected onto one end of the rod undistorted to the opposite end, where it can be viewed by eye, or a video camera. Image conduit is normally supplied with ground and polished ends. By heating it in a gas flame, image conduit can be bent to conform to any prescribed path with minimal transmission


Progressive scan array camera
Intense continuous lighting
Intense continuous lighting
Fast electronic (LCD) shutter is synchronised to the arrival of a widget in the camera's field of view using a photo-detector
Widgets
High-speed conveyor (continuous linear or circular motion)
a
Video frame period
Illumination strobed at video frame frequency (i.e. safe for human beings)
Max.
Off
T_L
Shutter, opened for short period ($T_S < T_L$) at low frequency
Open
Closed
T_S
Shutter, opened intermittently for longer time ($T_S > T_L$)
Open
Closed
T_S
b

◘ Fig. 40.34

Fast electronic shutter. (a) Optical layout. The lamp positions indicated here are not critical; they may be placed wherever they are needed, to optimise the image contrast. (b) Combining safe strobing (The frequency is equal to the video frame rate: 50/60 Hz.) and an electronic shutter. This method provides the same effect as low frequency strobing, but avoids its dangers. Notice that either the lighting or the shutter can be used to limit the total number of photons reaching the photo-sensor array; the light integration period is of duration *$MIN(T_L,T_S)$* seconds

loss. Image conduit rods typically range in diameter from 1.5–6 mm, with fibre diameters from 12–100 μm. The numerical aperture is typically 0.64, making the receiving end of the rod is somewhat selective about the direction of the incoming light rays that it will transmit. A cross-section converter usually has a much greater diameter and hence must be pre-formed during manufacture. Image conduit may superimpose a fine “chicken-wire” pattern onto the image

(➋ *Fig. 40.35c*) and this may require the use of special image processing operators. Components based on image conduit technology are made to provide special functions such as:

- Quarter twist (90°)
- Half twist (180°)
- Right angle bend
- Right angle with quarter/half twist
- Change of image size. (enlarging or reducing by a factor of about 3, or less)
- Detectors for UV or x-rays. (The front face of the conduit, or cross-section converter, is simply coated with an appropriate scintillator material. The efficient optical coupling achieved in this way allows a sensitive and rugged x-ray, or UV, sensor to be built.)

Image conduit and standard fibre optic face plates provide the basis for effective camera protection in quite hostile environments. Of course, it is important to check that the glass used in the construct of the image conduit is appropriate for that hazard.
REFERENCE Schott Fiber Optics, http://www.schott.com/fiberoptics/english/search/index
LAYOUT DIAGRAM ➋ *Figure 40.35*

40.39 Method 39: Fish-Eye Lens Views Inside Hole

OBJECTIVE To view inside a hole.
TYPICAL APPLICATIONS Inspecting:

- Holes in castings
- Pipes, tubes, drains, sewers, etc
- Automobile engine cylinders by entering through the fuel-air inlet or exhaust ports

EQUIPMENT Fish-eye lens and ring light. The latter may consist of either an array of LEDs or a fibre-optic ring light.
CAMERA Array.
LENS Fish-eye or semi-fish eye lens.
MOTION Static.
REMARKS The lens may be incorporated into an endoscope, enabling it to enter the hole. A fish-eye lens has a large depth of focus, so it is usually quite easy to maintain good focus over the whole of the camera's field of view. The viewing angle of a fish-eye lens may be 150° or more.
REFERENCE Ray SF (1988) Applied photographic optics. Focal Press, London/Boston, p 258. ISBN 0-240-51226-X
LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.36*

40.40 Method 40: Flexible Inspection Cell/Robot Vision Cell

OBJECTIVE To provide a flexible inspection/parts handling environment, capable of implementing several of the important lighting-viewing methods described elsewhere in this chapter. The system should allow the user to adjust the directions of both lighting and viewing, the size of the camera's field of view, the viewing background and insert appropriate optical

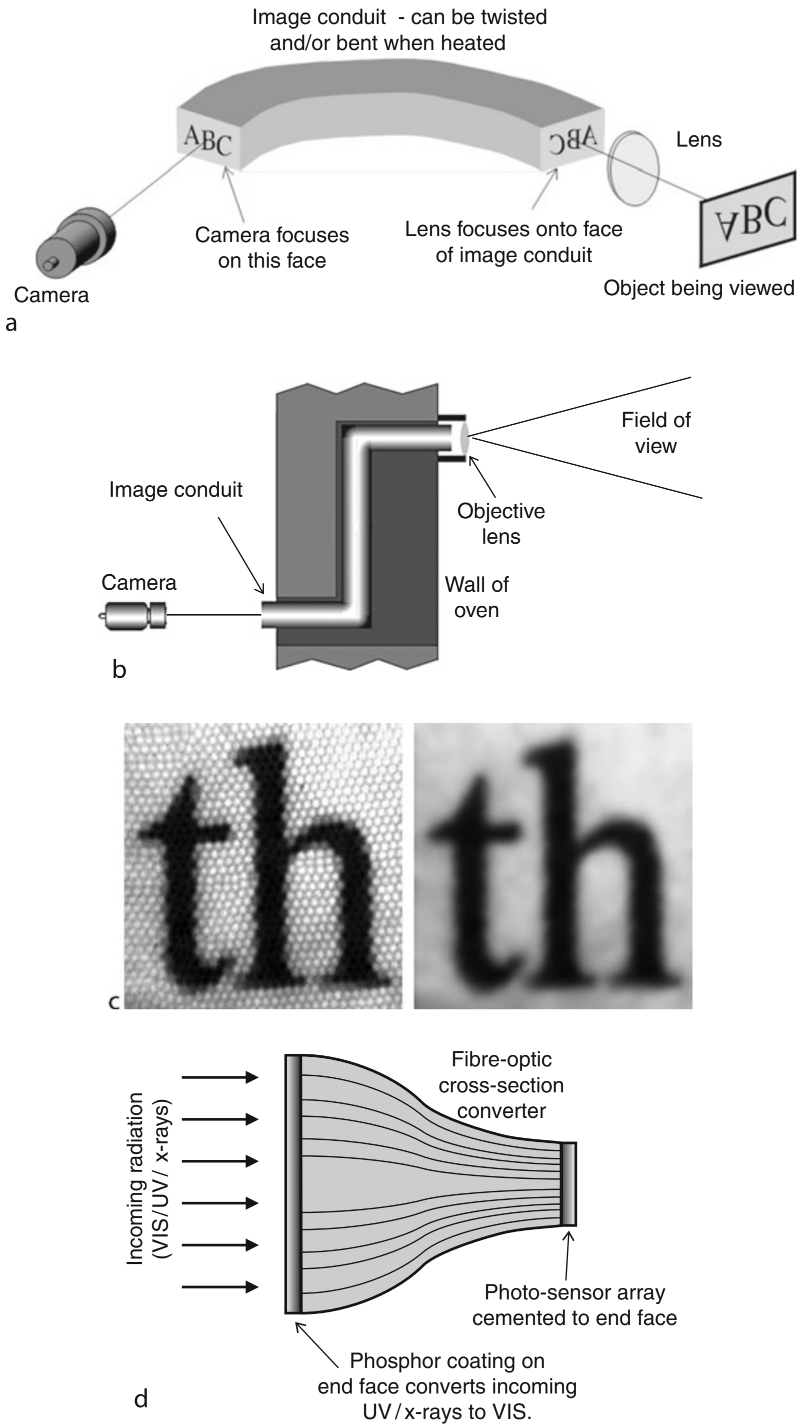


Fig. 40.35
(Continued)

elements (e.g., filters, diffusers and polarisers) in the optical path. A *Flexible Inspection Cell (FIC)* may be built around any of the popular manipulators: (X, Y, Z, θ)-table, gantry robot, SCARA robot *(Selective Compliance Assembly Robot Arm)* or revolute robot (multi-axis articulated arm).

TYPICAL APPLICATIONS Inspecting complex objects, assemblies of parts and low-volume applications. (When objects are made in small quantities, it may not be cost-effective to design and build a bespoke image acquisition sub-system.) An important application of an FIC is in prototyping vision systems.

EQUIPMENT Four of the most important types of robot can be integrated into an FIC:

(1) (X, Y, θ)-table or (X, Y, Z, θ)-table. (➲ *Fig. 40.37a*, *b*)
(2) Gantry robot. ➲ *Figure 40.37c*
(3) SCARA robot. ➲ *Figure 40.37e–h*
(4) Multi-axis articulated arm (revolute robot, ➲ *Fig. 40.37i*)

In addition the following items are needed:

(a) Computer-controlled lights. (VIS, UV and IR, as appropriate)
(b) Laser light-stripe generator (to create depth maps, Method 107)
(c) Pattern projector (ruggedised slide projector, Method 68)
(d) Several cameras facing different backgrounds (black, mid-grey, white/self-luminous). These may be either array- or line-scan types
(e) Standard, wide-angle and telephoto lenses, to provide different magnifications. (It is usually easier to calibrate a multi-camera system with fixed lenses, rather than use variable zoom lenses.) The lenses may be auto-iris type, to compensate for changes in lighting levels
(f) A variety of filters, diffusers and polarisers

MOTION Image acquisition normally takes place when the widget is static, using an array camera. Most robots allow the user to define which points in state space are to be visited but not the path between them. Hence, while it is moving, there is some uncertainty about exactly where the robot is at any given moment. However, accurately defined linear and rotary scans can be provided by an (X, Y, Z, θ)-table, driven by stepper motors. A line-scan camera can then be used. (c.f. Methods 59 and 83).

WIDGET HANDLING MECHANISMS It is possible to design a FIC around a variety of robotic manipulators. Four of the principal types are discussed below.

(X, Y, Z, θ)-table (➲ *Fig. 40.37a–b*) Provides $2^{1}/_{2}$, 3, $3^{1}/_{2}$ or 4 degrees of freedom. Avoiding collisions between the manipulator and static objects, such as lights, cameras and their support structures, is straightforward, as the work space is a cuboid. A FIC can be designed around any convenient translation stage, including devices, where the work volume is large (in excess of 1 m^3), or very small and providing micron-level position tolerances. Good

▪ **Fig. 40.35**

Fibre-optic image conduit used to bend/twist the light path in an imaging system. (a) Optical layout. (b) Image conduit embedded in the wall of a hazardous viewing environment protects the camera from heat, dirt, moisture or radiation. (c) Sample image: *Left.* Original. Notice the "chicken-wire" pattern caused by the finite diameter of the optical fibres. *Right:* Low-pass (blurring) filter effectively removes the "chicken-wire" pattern. (d) Cross-section converter, used to convert x-rays or UV into VIS

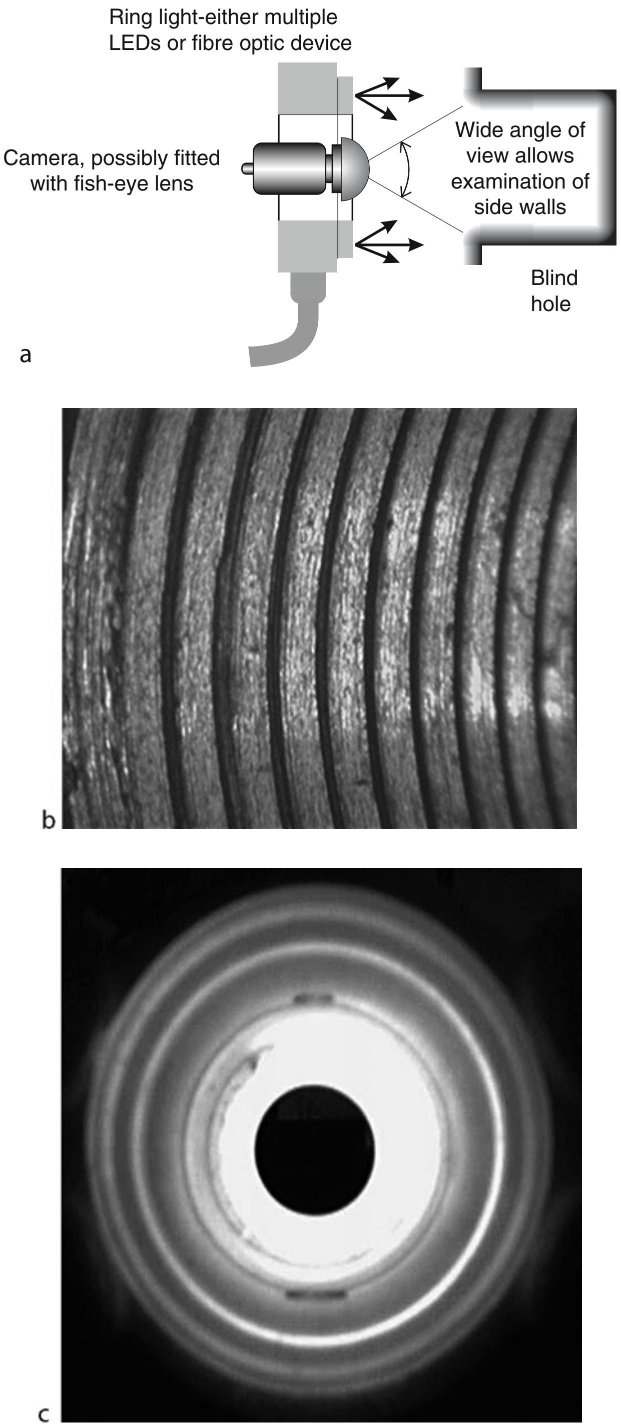

Ring light-either multiple LEDs or fibre optic device
Camera, possibly fitted with fish-eye lens
Wide angle of view allows examination of side walls
Blind hole
a
b
c

Fig. 40.36 (Continued)

precision and repeatability can be achieved if stepper motor drives and optical position feedback are used. By using other types of motor and gear mechanisms, accuracy and speed of movement can be traded against each other, as the application demands. The maximum permissible payload is good. Cameras and lights can be placed anywhere above the viewing platform, as there is no obstacle for the lighting. A large number and wide variety of lamps (e.g., IR, VIS and UV) can be used, as access is not limited. Since the widget rests directly on the viewing background (as perceived by an overhead camera), the latter may become scratched/dirty quite quickly. It is easy to fit jigs for holding awkwardly shaped widgets, as they are being inspected. A pick-and-place arm can be used for loading/unloading widgets onto the table. This is a convenient low-cost, “do it yourself” solution, capable of satisfying the needs of a wide variety of applications and it can be tailored easily for specific requirements.

Gantry robot (➲ *Fig. 40.37c–d*) Provides $2^1/_2$, 3, $3^1/_2$ or 4 degrees of freedom (i.e., same as the FIC based on an (X, Y, Z, θ)-table). The work space is a cuboid. Access for lighting is slightly limited by the legs. (This ignores the fact that a gantry robot can hang from the ceiling, rather than stand on the floor.) The legs can be used for mounting small lighting modules, for example using fibre optics or LEDs. The payload is likely to be lower than that for the (X,Y, θ)-table and the gripper obscures part of the widget. It is not possible to use an overhead camera, unless the robot releases the widget and then moves away, prior to image capture. We can add a camera mounted below and looking through the table-top. (➲ *Fig. 40.37d*) A mirror folds the optical path, allowing the camera and its lens to be moved out of harm’s way. However, the protective window and mirror must be regarded as sacrificial components, since there is a danger that the robot will drop a widget onto them. LED lighting can also be built into the table-top. This camera views the widget against a complicated and uncontrolled background (i.e., the gripper and under-side of the robot arm).

SCARA robot (➲ *Fig. 40.37e–h*) Provides $2^1/_2$, 3, $3^1/_2$ or 4 degrees of freedom. The work space is a vertical cylinder with a crescent-shaped cross section in the horizontal plane. Access for lighting is good, as the only obstacle is the arm-support column. Provided it is not too tall, a rigid pillar holding a series of small lamps (e.g., LEDs) can be placed safely beneath the robot’s arm, close to that column. An overhead camera cannot see a widget held in the gripper. However, a view looking upwards can be obtained by a camera fitted into the table-top (➲ *Fig. 40.37d*).

Revolute robot (➲ *Fig. 40.37i*) Typically, provides 4–6 degrees of freedom. Designing a FIC around a revolute robot can be difficult, since the work volume is large: almost a complete sphere. It is difficult to control a non-deterministic revolute robot to prevent it from colliding with static objects, such as cameras, lights and the support scaffolding. (For every point on the robot’s trajectory in multi-dimensional state space, the collision-avoidance program must take into account the position of every point on the arm, gripper and widget.)

However, two useful FIC configurations exist:

◘ Fig. 40.36

Examining a hole without entering it. (a) Optical set-up. (b) Sample image: Female screw thread. (c) Sample image: bore of diameter 56 mm and depth 78 mm. The bright concentric rings are reflections from shallow circumferential ridges. The short dark streaks at the top and bottom of the bright central annulus are circular ports opening into the base of the main chamber. (For an overall view of this object, see ➲ *Fig. 40.81c*.)

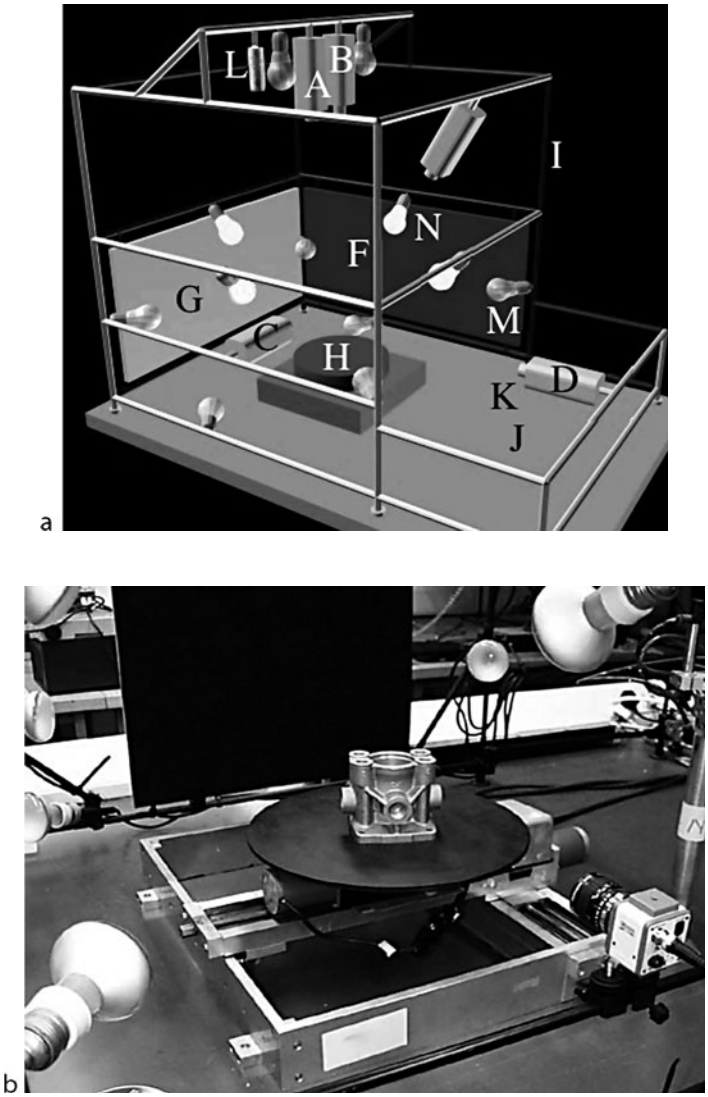


Fig. 40.37
(Continued)

1. Place the camera on the robot arm. (See Method 124)
2. Attach the camera to the body of the robot

The latter is particularly convenient, since the camera rotates with the arm and widget, allowing us to adjust the direction of lighting by simply moving the robot (i.e., varying θ in ➲ *Fig. 40.37i*). Although there is no simple relationship between the pose of the robot and the image received by the camera, the major effects can be summarised thus:

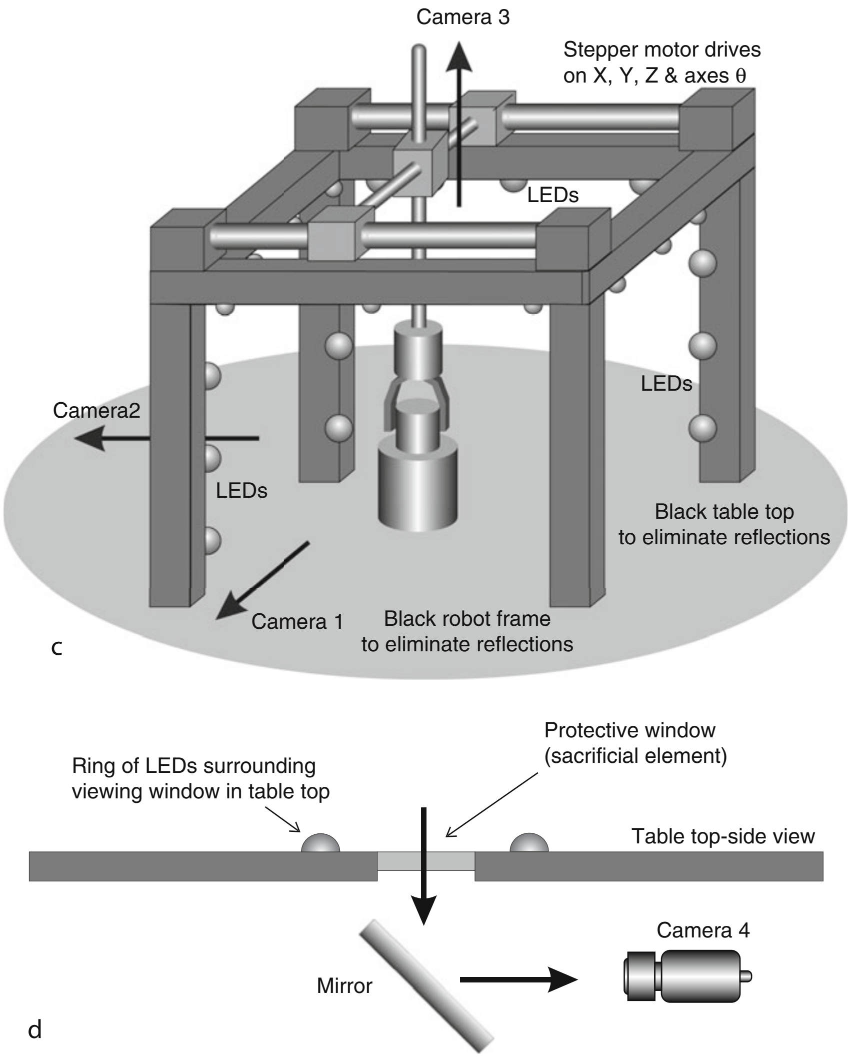
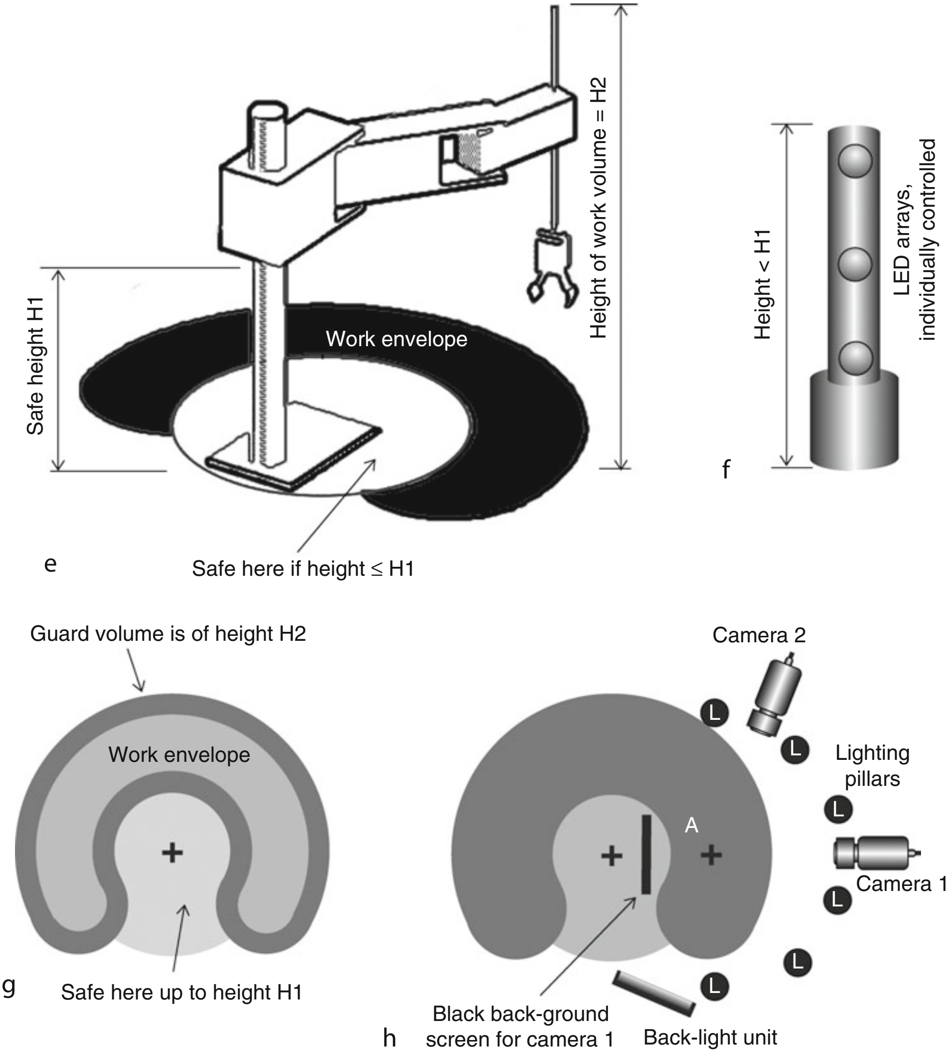


Fig. 40.37
(Continued)

Robot axis	Most significant change is to:
θ (rotation in the horizontal plane)	Lighting direction
ϕ/φ (shoulder and elbow joints)	Magnification/focus and vertical shift
δ (pitch)	Vertical scale in the image plane
ε (roll)	Horizontal scale in the image plane
Yaw (Not shown in *Fig. 40.37i*.)	Image rotation


Height of work volume = H2
Safe height H1
Work envelope
Safe here if height ≤ H1
e
Height < H1
LED arrays, individually controlled
f
Guard volume is of height H2
Work envelope
+
Safe here up to height H1
g
Camera 2
L
Lighting pillars
A
+
Camera 1
Black back-ground screen for camera 1
Back-light unit
h

■ **Fig. 40.37 (Continued)**

REMARKS The FIC was originally devised in response to the development of Flexible Manufacturing Systems and is intended to provide an inspection facility of commensurate versatility. A well-designed FIC can implement many of the lighting-viewing methods described elsewhere in this chapter but is limited to inspecting objects within a somewhat limited range of sizes (typically 25–200 mm). A filter wheel can be used to adjust the camera's spectral response, to insert a secondary lens or polariser into the optical path. ➔ *Figure 40.37j* An FIC, controlled remotely via the Internet, may be useful for prototyping vision systems for distant factories, without ever visiting them. The ALIS 600 system (➔ *Fig. 40.37k*) is similar in

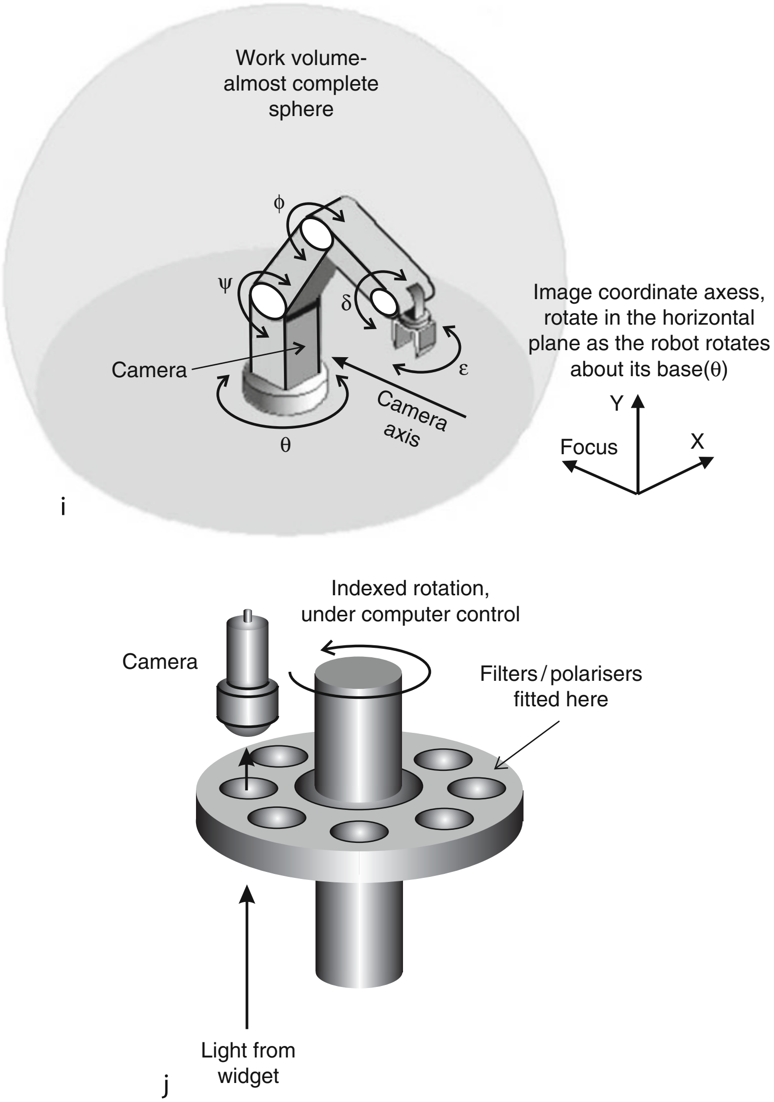


Fig. 40.37
(Continued)

general concept to the FIC described in *Fig. 40.37a* but has no robotic manipulator. Hence, all adjustments are made by hand. This might not matter too much if a vision engineer is available on-site. However, the FICs described here are intended to be operated remotely, via a LAN, WAN or cell-phone network, so that the vision engineer does not have to travel unnecessarily. The human-computer interface needed to control a remotely-operated FIC requires careful attention.

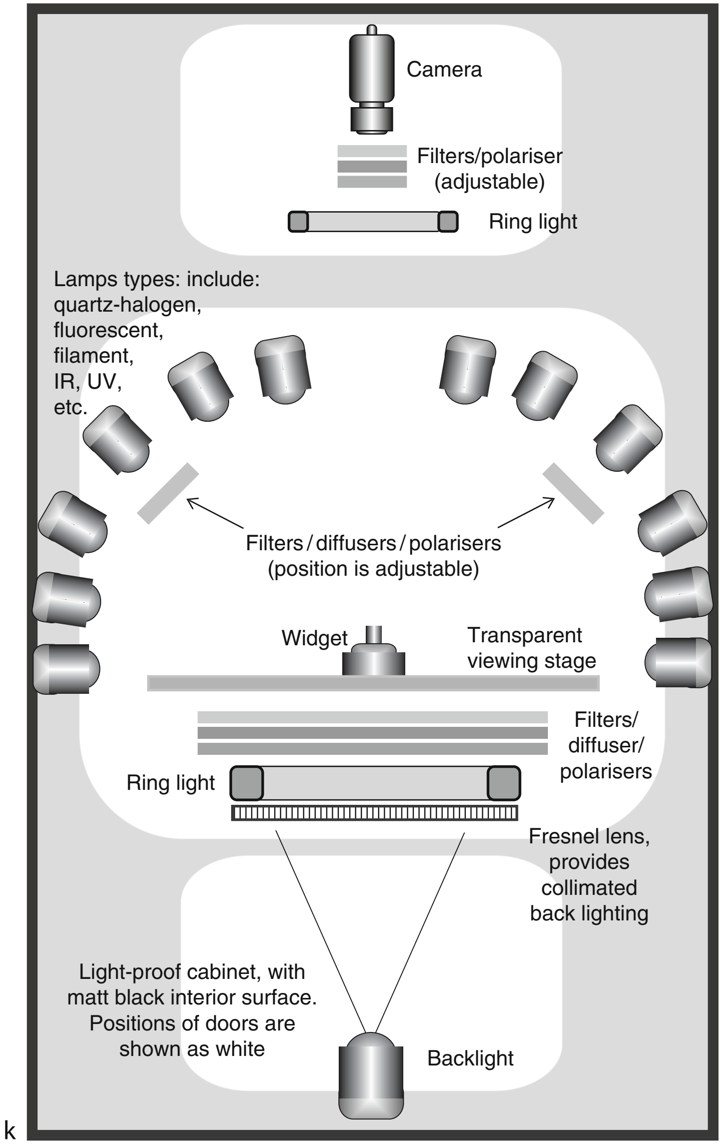

Camera
Filters/polariser (adjustable)
Ring light
Lamps types: include: quartz-halogen, fluorescent, filament, IR, UV, etc.
Filters / diffusers / polarisers (position is adjustable)
Widget
Transparent viewing stage
Filters/ diffuser/ polarisers
Ring light
Fresnel lens, provides collimated back lighting
Light-proof cabinet, with matt black interior surface. Positions of doors are shown as white
Backlight
k

Fig. 40.37

Flexible inspection cell (five following pages). (a) Flexible inspection cell, based on an (X,Y,θ)-table and a pick-and-place arm. Features: *A, C, D* and *E* – Cameras with standard lenses; *B* – Camera with telephoto lens; *F* – black back-ground for camera C; *G* – white back-ground for camera D; *H* – (X,Y,θ)-table with stepper motor drives; *I* – supporting frame (standard black-coated steel laboratory scaffolding), based on a 1m^3 cube; *J* – pick-and-place arm (not shown here for clarity); *K* – slide projector, provides patterned illumination (not shown here for clarity); *L* – laser light-stripe generator, used with camera E to generate depth maps. *M* – Lamp unlit; N – Lamp lit. [Based on artwork by Richard Hitchell]. (b) Photograph of the FIC built at Cardiff University, showing the (X,Y,θ)-table, camera C, the black back-ground board F and several lamps

REFERENCES

1. Batchelor BG, Daley MW, Karantalis G (2001) Tele-prototyping environment for machine vision. Electron Imaging 10(1):316–328. ISSN 1017–9909
2. Batchelor BG, Waltz FM (2001) Intelligent machine vision: techniques, implementation and applications. Springer, Berlin. ISBN 3-540-76224-8
3. Rauchmiller RE Jr, Harding KG, Michniewicz M, Kaltenbacher E (1990) Design and application of a lighting test bed. In: Proceedings of the MVA/SME conference on VISION 1990, Detroit. Society of Manufacturing Engineers, Dearborn, paper MZ90-615
4. ALIS 600, formerly a product of Dolan-Jenner Industries

(60 W filament type). In a more recent development, lighting is provided by a set of eight white multi-LED arrays. (c) Design for a FIC based on a gantry robot. The lighting is provided by a series of small LED modules, attached to the robot's frame in such a way that they will not interfere with its movement. (d) Adding a fourth camera to the FIC based on a gantry robot. (Side view through the table-top.) A similar arrangement can be used in conjunction with the FICs based on SCARA and revolute robots. (e) Visualising the work space for a SCARA robot. That region in 3D space swept by the robot arm is called the *work space* (or *work volume*) and must be kept clear of obstacles, to avoid collisions. The work space for a SCARA robot consists of a vertical cylinder, with a crescent-shaped cross section, and a "bridge," with clearance height H1, stretching between it and the arm-support column. In practice, a slightly larger region of 3D space, called the *guard volume*, must also be kept clear if the widget extends beyond the end of the robot's arm. The intersections of the work space and guard volumes with the horizontal plane, forms the *work envelope* (black crescent) and *guard region*. These are simply convenient ways for visualising the robot's work space. [Artwork by David Batchelor]. (f) Illumination for a FIC, based on a SCARA robot can be provided by a series of lamps, possibly arrays of LEDs, mounted on rigid pillars. Provided the lighting pillar is not taller than H1, it can fit safely beneath the robot's arm close to its base. (g) Relationship between the work envelope and the guard region for a SCARA robot. (h) Layout (plan view) for an inspection cell based on a SCARA robot, providing front, side and back illumination. During inspection, the widget is held at point A by the robot. To obtain high-precision measurements, additional cameras, with narrow fields of view may be stacked above cameras 1 and 2. Of course, it is not possible for an overhead camera to view a widget held in the robot's gripper, although it can see objects lying on the table. (This is not always convenient, as the robot is required to release its grasp, move out of the field of view during image capture and then pick up the widget again.) A camera, looking vertically upwards, through a window set into the table-top may be used instead. (See part (d) of this diagram.). (i) Work space for a multi-axis articulated arm (revolute robot). By placing the camera on the body of the robot, we can adjust the direction of lighting from a fixed point source at will. This effectively defines a set of coordinate axes that rotate with the robot (θ varies). The image coordinates are XY. (j) Filter wheel. May be used with any of the cameras within a FIC to alter its spectral response. (k) Schematic layout for a multi-function lighting system. Based on the ALIS 600 system, manufactured by Dolan Jenner, Inc. Unlike the FIC, all adjustments to the camera, lights and widget are made manually

40.41 Method 41: Fluorescence and Phosphorescence

OBJECTIVE To use fluorescence or phosphorescence to differentiate between materials which otherwise appear to be indistinguishable (i.e., they have the very similar colour, shade, sheen and texture).

TYPICAL APPLICATIONS *Manufacturing industry:* detecting leakage and spots of oil, grease and various organic chemicals. *Food:* detecting spillage of fillings from pies, cakes, etc. (differentiating the contents from the pastry case), detecting kernel of the stones in cherry flesh, detecting nematode worm infestation in the flesh of fish.

EQUIPMENT Ultra-violet light source. (For safety's sake, the lamp must be shielded to prevent stray UV light from damaging the eyes of workers nearby.) UV-blocking filter, VIS-blocking filter.

CAMERA Array camera, which must be quite sensitive to detect the low light levels emitted during fluorescence. Phosphorescence often requires an even more sensitive camera, since the light output is lower.

LENS Low *f*-number, to collect the maximum amount of light.

MATERIALS Organic materials (e.g., food), paper, cloth, some plastics, oil, grease, etc. frequently fluoresce. Some materials, most notably paper, have fluorescent whiteners added to make them during manufacture to make them *"brighter than white."* (In the 1960s, this slogan was used to promote sales of a popular domestic detergent which had a fluorescent whitener in it.)

REMARKS Fluorescence is the property of a material which glows in one wave-band (long wave-length, e.g., visible light) when it is irradiated with another (short wavelengths, e.g., ultra-violet) However, the process is more general than the familiar UV-VIS fluorescence. For example, some materials glow in the orange-red part of the spectrum, when illuminated with blue light. Certain materials have a visible glow when they are irradiated with x-rays. Fluorescence stops when the illumination is switched off, while phosphorescence continues for a short while after it is removed. Some substances fluoresce at a particular wavelength, which can be identified using optical band-pass filters. It is important to follow strict safety practices when working with short-wavelength radiation (UV and x-rays), since it can injure both skin and eyes.

LAYOUT DIAGRAM AND SAMPLE IMAGES ➲ *Figure 40.38*

40.42 Method 42: Focussed Annulus of Light

OBJECTIVE To obtain a focussed annulus of light. Using this method, a thin, continuous bright ring of light can be projected onto a plane surface.

TYPICAL APPLICATIONS Viewing tops of cans, jars, bottles, etc.

EQUIPMENT

- Ring light
- Lens with its centre removed. (This allows the camera to view the widget directly.)

CAMERA Array or circular scan.

MOTION Static.

GEOMETRY Annulus: the centre of the widget is assumed to be unimportant and does not need to be inspected.

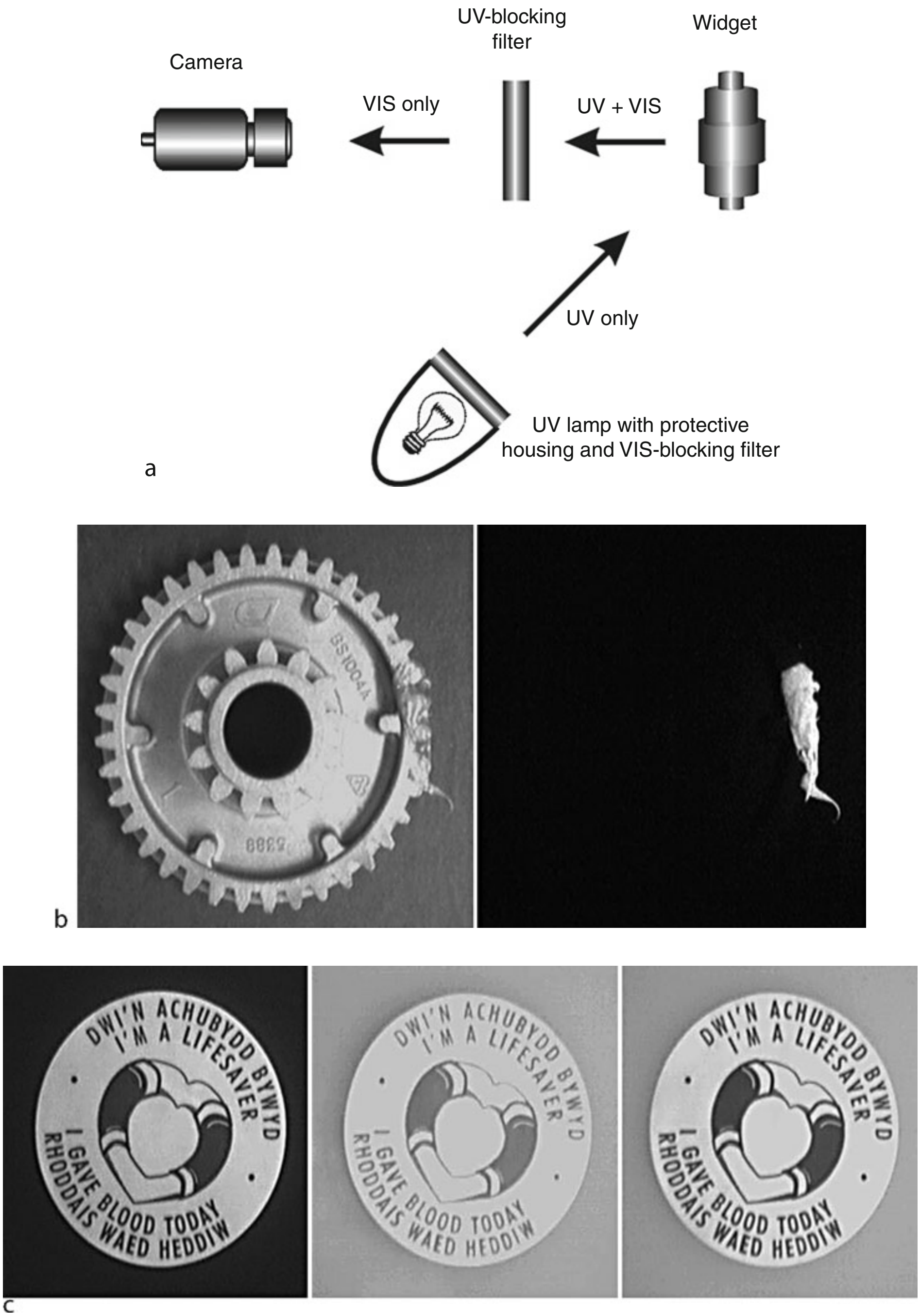


Fig. 40.38
(Continued)

REMARKS A fibre optic ring light produces a thin circle of light, whereas a LED array generates a set of bright points, interspersed with darker areas. This uneven illumination pattern can be smoothed out by adding a diffuser but this broadens the circle of light. A plastic lens can be machined easily, by milling out the centre of a conventional lens. For large widgets, a plastic Fresnel lens can be effective but colour aberration may generate coloured bands on the inner and outer edges of the annulus. This effect can be overcome, if narrow-band lighting is used. Care must be taken to ensure that rays from the ring light cannot reach the widget without first travelling through the lens.

LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.39*

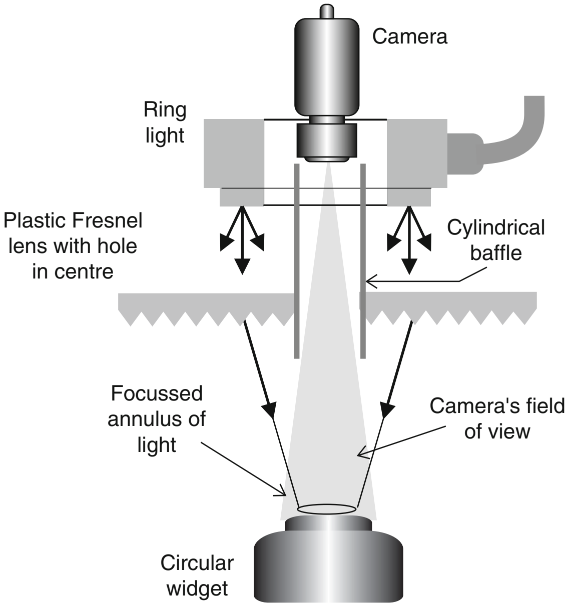


Fig. 40.39
Focussed annulus of light

Fig. 40.38
Fluorescence and phosphorescence. (a) Optical layout for ultra-violet fluorescence. (b) Sample image: Grease on a zinc die-cast gear. *Left:* Viewed in ambient lighting. *Right:* Illuminated in ultra-violet and viewed in the VIS wave-band. (c) Sample images: Self-adhesive paper label (white), with red printing and attached to a sheet of white perspex (plexiglass) *Left:* Illuminated in ultra-violet and viewed in the visible wave-band. A UV-blocking filter and VIS-blocking filter were both used. Notice that the white paper fluoresces strongly, due to the brightener added during manufacture. Neither the perspex nor the red ink fluoresce. *Centre:* Viewed in white light, with no filter. *Right:* Viewed in white light, with a green filter. (A green filter attenuates red light, from the printing, and hence optimises the contrast.) (d) Optical set-up for phosphorescence

40.43 Method 43: Forced Thermal Emission

OBJECTIVES To visualise heat emission from objects that are cold initially.
TYPICAL APPLICATIONS Investigating solid objects for internal cracks, voids, air bubbles, etc. To investigate underlying structures whose appearance has been homogenised by paint or other surface treatment.
EQUIPMENT Microwave, eddy current, laser, IR or hot-air heater. Alternatively, the sample to be examined can be heated, by forced vibration.
CAMERA Thermal imager.
OPTICS Lenses are probably germanium, although these are very expensive. (Germanium is opaque to visible light). Reflective (mirror) optics are cheaper to produce, by evaporating gold onto a suitable substrate, but are not so easy to use.
MOTION Static.
REMARKS This is an expensive method to implement, because thermal imagers and their lenses are costly. However it does offer some unique advantages, in being able to visualise underlying structure without the use of penetrating radiation. For example, a component consisting of metal and rubber bonded together, will display marked temperature differences as heat is applied locally, even if it is covered in paint, or lacquer. There are two reasons why a crack may appear brighter than the surrounding material. Firstly, flexing a structure increases stress locally around a crack, or inclusion, and thereby raises the temperature in that region. Secondly, a crack acts in a similar way to a black body radiator, which has a high IR emissivity (about 0.9). Compare this to that of polished metal, which has a much lower emissivity: approximately 0.1–0.2.
REFERENCES SIRA Institute, South Hill, Chislehurst
LAYOUT DIAGRAMS ➜ *Figure 40.40*

40.44 Method 44: Generating Light with Arbitrary Spectrum

OBJECTIVE To generate light with an arbitrary spectrum. The spectrum can be adjusted automatically during a multi-frame image-capture sequence.
TYPICAL APPLICATIONS Colour analysis.
EQUIPMENT Xenon lamp or other pan-chromatic light source, Two cylindrical and two standard lenses, two diffraction gratings, optical mask. The last mentioned can be fixed, a motorised slit, or a programmable LCD light switch.
CAMERA Array or line-scan.
REMARKS Let λ be the wavelength; L(λ) the spectrum of the light source; M(λ) the mask weighting function and D(λ) the desired spectrum. Then, we should choose M(λ) such that

$$\mathrm{M}(\lambda) = \mathrm{D}(\lambda)/\mathrm{L}(\lambda).$$

For the sake of accuracy, we must ensure that L(λ) never approaches zero. For this reason, we should choose a lamp which has a nearly flat spectral output; L(λ) is nearly constant over the appropriate range of wavelengths. The mask can be fixed (e.g., 50 mm slide); a motorised slit (to generate monochromatic light of variable wavelength and/or variable-width waveband), or fully programmable (using an LCD light switch).

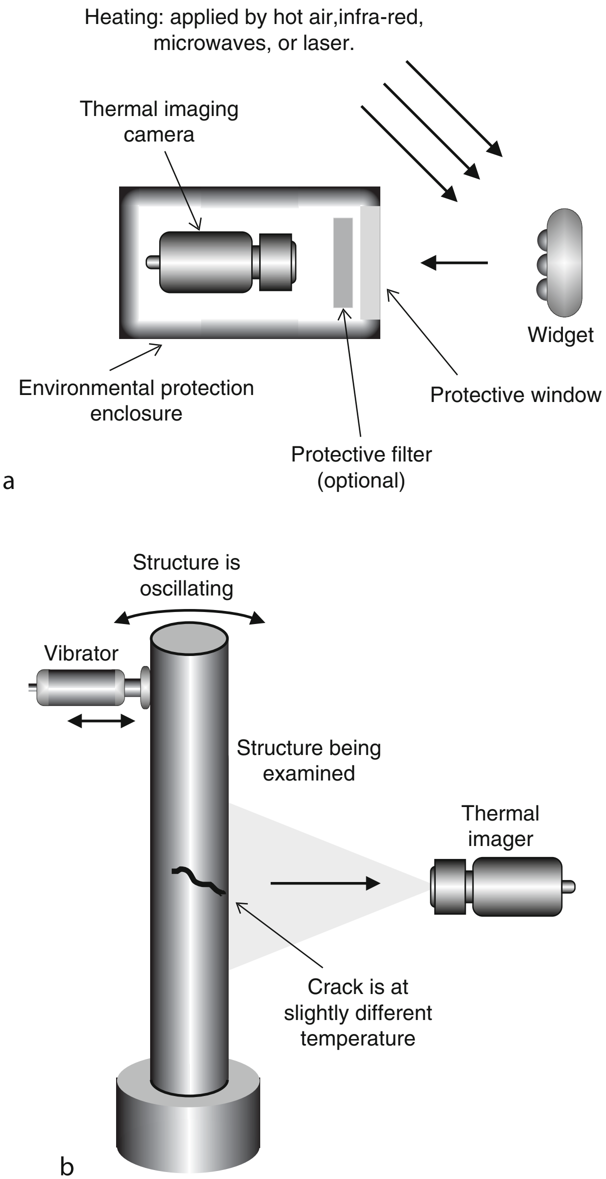

Heating: applied by hot air,infra-red, microwaves, or laser.
Thermal imaging camera
Widget
Environmental protection enclosure
Protective window
Protective filter (optional)
a
Structure is oscillating
Vibrator
Structure being examined
Thermal imager
Crack is at slightly different temperature
b

Fig. 40.40

REFERENCES Larcourt A, Torres Y (1983) Production of an arbitrary spectrum by a spatial filtering method. In: Proceedings of the SPIE conference on applications of digital image processing, vol 397, pp 198–202

LAYOUT DIAGRAM *Figure 40.41*

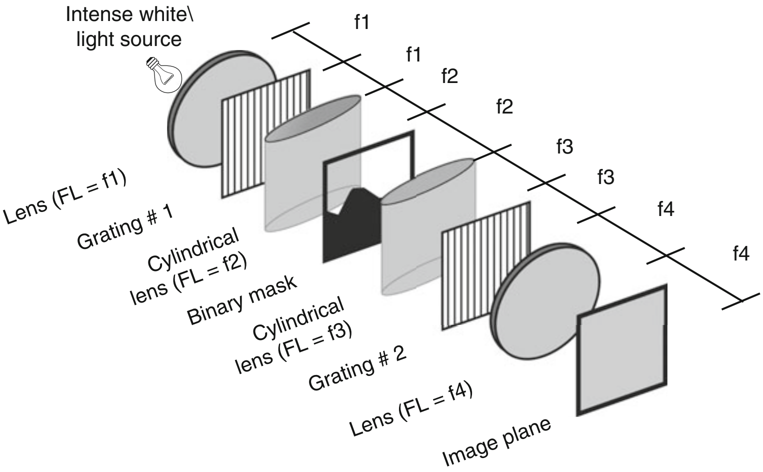


Fig. 40.41
Generating light with an arbitrary spectrum

40.45 Method 45: Granular Material (Pellets) in Free Flight

OBJECTIVE To inspect granular material or pellets in free flight.
TYPICAL APPLICATIONS Inspecting materials such as

- Seeds, nuts, peas, beans, etc
- Breakfast cereals and similar processed food products and semi-processed food materials
- Low value confectionary (i.e., not fragile chocolates)
- Gravel-like minerals
- Industrial materials in the form of pellets, frozen droplets, etc

EQUIPMENT

- High-speed conveyor, capable of projecting the pellets to be inspected into nearly horizontal flight past the camera. Alternatively, a slide-way may be used instead
- Array of air jets, activated by the image processor, to deflect "bad" pellets
- Long linear light source, such one of the following
    1. Fluorescent tube driven at high frequency (to avoid strobing effects)
    2. Linear array of LEDs
    3. Fibre optic bundle with a linear output aperture
    4. Tube of *TIRF* film
    5. Rod of glass/perspex with light injected at on/both ends

CAMERA Line-scan.
MOTION Continuous nearly linear motion is provided by the high-speed conveyor. Notice that the pellets being inspected have a finite velocity component along the camera's optical axis. This is unlikely to cause problems with focussing but must be taken into account when designing the *accept/reject* deflector system.
MATERIALS Granulated materials, powder and pellets.
REMARKS A suitable viewing background for a line-scan camera can be provided by a rotating cylinder, kept clean by continuous washing/brushing. By careful design of the

lighting, it is possible to avoid illuminating the background, in which case accumulations of debris are unimportant, (This remark refers only to Machine Vision, not to hygiene.) It is, of course possible, to rotate the optical system by about 45° and use a slide-way, instead of the conveyor belt. It is important for proper operation of the *accept/reject* deflector system that the velocity of pellets passing through the light beams be known accurately. Rapid-response air jets (with switching speeds of tens of milliseconds) are available commercially.

REFERENCES Szanto G (1994) A taste of vision. Image Process 6(1)

LAYOUT DIAGRAMS ➋ *Figure 40.42*

40.46 Method 46: Grazing Illumination

OBJECTIVES

- To highlight shallow surface features, created by engraving, embossing, rolling, pressing, stamping, casting, moulding, etc
- To detect surface imperfections, such as scratches, cracks, pits, dirt, dust, grit, etc. on a smooth mirror-like surface

TYPICAL APPLICATIONS Inspecting flat machined metal surfaces, plastic, glass, glossy painted and lacquered surfaces, plastic sheeting, ceramic tiles, etc.

EQUIPMENT Collimator. For close-range operation, it may not be necessary to use a collimator. In some applications, a very small ("point") source of light, such as a LED or fibre optic light guide, is adequate.

CAMERA Array.

LENS Short focus.

MOTION Static.

GEOMETRY Nominally flat and smooth. Both depressions and raised features are visible.

MATERIALS Any reflective surface.

REMARKS Care must be taken to prevent the camera from seeing a reflection of itself. (This can usually be achieved by tilting the widget slightly.) Linear features, such as scratches and cracks that are approximately normal to the light path are highlighted, Cliff-like edges facing the light source tend to cause glinting, while those facing away create shadows. Linear features that are nearly parallel to the light beam are suppressed. This anisotropy can cause difficulties but is useful in certain situations.

LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.43*

40.47 Method 47: Hand-held Sensor for Fixed-range Close-up View

OBJECTIVE To view an object at fixed, close range in high magnification using a hand-held sensor.

TYPICAL APPLICATIONS

- Sampled inspection of paint-work, machined and polished surfaces
- Reading dot-matrix codes
- Reading alpha-numeric codes and labels (e.g., product batch numbers, best-before and sell-by dates)
- Hardness testing

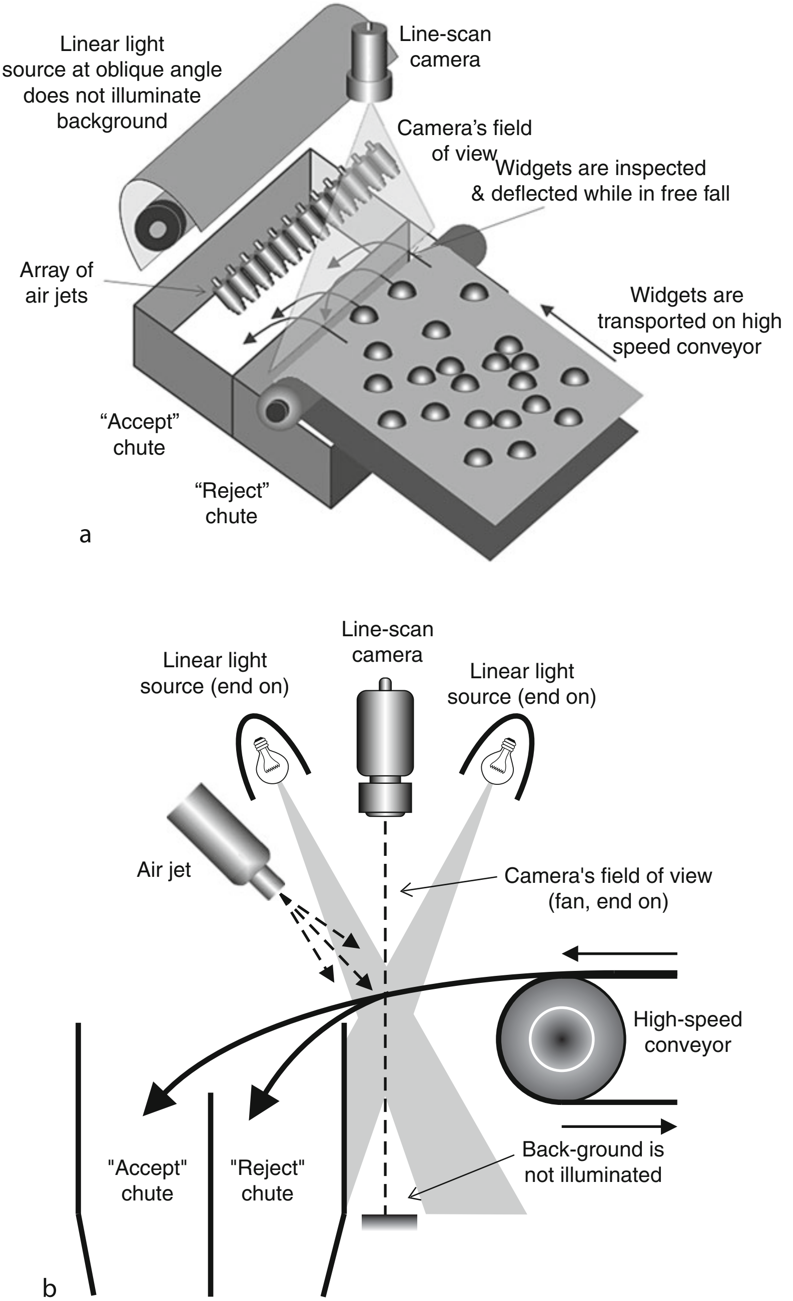
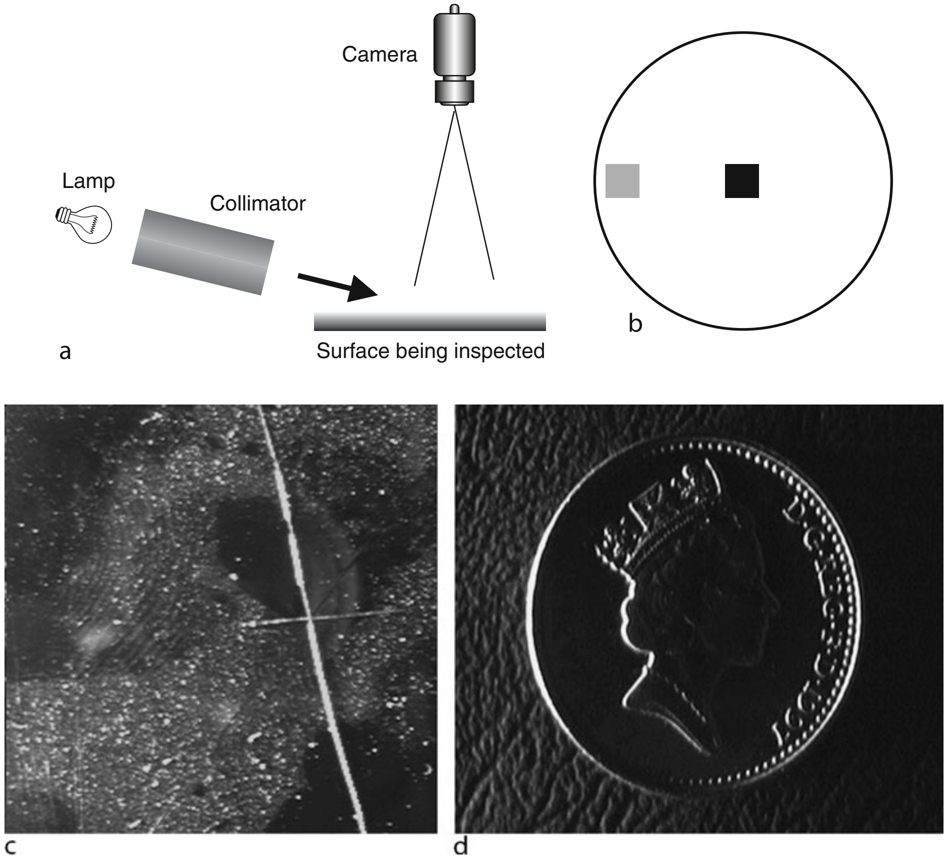

Linear light source at oblique angle does not illuminate background
Line-scan camera
Camera's field of view
Widgets are inspected & deflected while in free fall
Array of air jets
Widgets are transported on high speed conveyor
"Accept" chute
"Reject" chute
a
Line-scan camera
Linear light source (end on)
Linear light source (end on)
Air jet
Camera's field of view (fan, end on)
High-speed conveyor
Back-ground is not illuminated
"Accept" chute
"Reject" chute
b

Fig. 40.42

Inspecting granular material. (a) Overall view. Not shown: second light source. (b) Side view, showing the optical arrangement and the *accept/reject* deflector (array of air jets). Notice that the back-ground is not illuminated and hence appears to be dark, compared to the material in the flight path and within the coincident light beams



■ Fig. 40.43

Grazing Illumination. (a) Optical arrangement. (b) Illumination diagram. (c) Mirror with a lot of surface dirt, a finger-print and two scratches (A third diagonal scratch is invisible when lit from this direction). A clean and perfect mirror, would be very dark. (d) Coin. In both cases, the light is projected from the left. Notice that the textured background is clearly visible outside the shadow of the coin

EQUIPMENT Purpose-built optical head shown in ➜ *Fig. 40.44*.
LENS Close-up lens is part of the optical head.
MOTION Static. Probe is hand held.
GEOMETRY Nearly flat within small field of view.
MATERIALS Free from dust, lint, swarf, etc.
REMARKS Old-fashioned oscilloscope cameras were built on this principle. A probe of this kind was developed specifically for viewing moles and other skin features. During image acquisition, the probe is held against the surface being inspected by a human operator, or a compliant robot. The dome on the probe has a hole its centre, so there is no obstruction to the camera’s view. The inside of the dome should be blackened, to eliminate reflections of unwanted stray light. If dust is a serious problem, an air purge may be used to keep the front window clean. In any case, the dome should be removable, to

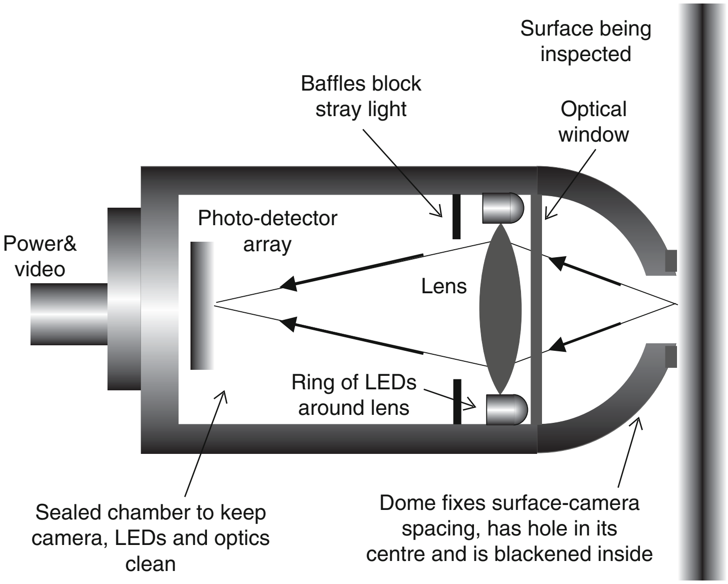

Surface being inspected
Baffles block stray light
Optical window
Power& video
Photo-detector array
Lens
Ring of LEDs around lens
Sealed chamber to keep camera, LEDs and optics clean
Dome fixes surface-camera spacing, has hole in its centre and is blackened inside

Fig. 44
Hand-held sensor for fixed-range close-up viewing of small surface defects

allow easy cleaning of the front surface of the protective window. The dome may be replaced by a wire cage, or cone, if greater viewing distances and/or a larger field of view are needed. Tri-colour or NIR LEDs may be used to adjust the spectral balance of the light. The LEDs can be strobed, to “freeze” the motion of the surface and/or increase their light output. The direction of the light incident on the widget surface can also be adjusted by controlling each device individually.
REFERENCES Micro-scopeman ring light and camera probe. Moritex, Cambridge
Handheld digital microscope. Dino Lite, http://www.dino-lite.com/. Accessed 10 Feb 2011
LAYOUT DIAGRAM ➲ *Figure 40.44*

40.48 Method 48: Hardness Testing

OBJECTIVES To measure the hardness of a metal surface.
TYPICAL APPLICATIONS Measurement of cutting tools (drill, milling tool, etc.) specialised steels, etc.
EQUIPMENT Vickers or Brinell hardness tester.
CAMERA Array.
LENS High magnification.
MOTION Static.
GEOMETRY Smooth and flat at the point where the hardness is to be tested. It may be necessary to machine a small flat section (about 2 × 2mm^2) prior to hardness testing.
MATERIALS Steel.
REMARKS The Vickers method of testing for hardness uses a diamond tool, which is pressed into the surface with a known force. The Brinell test uses a spherical tool. The size of the

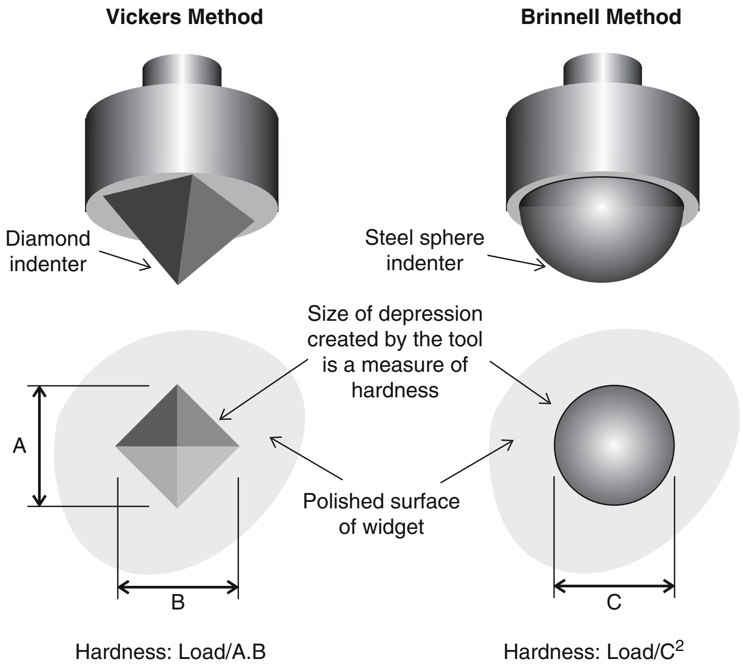


Fig. 40.45
Measuring surface hardness

indentation is measured and indicates the surface hardness. Commercial devices for automatic hardness testing use image processing to measure the size of the indentation. Diffuse illumination is achieved by Method 31.

REFERENCES

- Hodgson DC (1985) Microprocessor-based visual inspection systems. In: Batchelor BG, Hill DA, Hodgson DC (eds) Automated visual inspection. IFS, Bedford. ISBN 0-903608-68-5
- Clemex CMT microhardness testing. Clemex Technologies, Longueuil. http://www.clemex.com/products/hardness-tester-cmt.htm

LAYOUT DIAGRAM ➋ *Figure 40.45*

40.49 Method 49: Height Analysis by Projecting Patterns

OBJECTIVE To provide a rapid method of obtaining information about the 3D geometry of a surface.

TYPICAL APPLICATIONS Inspecting mouldings, castings, food products, such as cakes, loaves, etc.

EQUIPMENT Computer-controlled LCD projector, acting as a spatial light modulator.

CAMERA Array.

MOTION The widget must be static during a pre-defined number of video frames (typically 4–16).

GEOMETRY The widget surface should not be overly complicated, nor have a large number of step-like features.

MATERIALS Matt, non-reflective. Plastics, ceramics, painted surfaces.

REMARKS When a set of parallel lines is projected onto a surface, the position of each bright-dark, or dark-bright, transition point can provide information about its height. However, this results in some ambiguity in the measured height values if, as often happens, adjacent light stripes cannot be distinguished. This ambiguity can be resolved by the *Coded Light Approach*, in which a small number of binary images is projected onto the widget in turn. This results in an absolute measurement of surface height and allows us to construct a depth map relatively quickly. A sequence of binary images is projected onto the widget surface, using a spatial light modulator. (One such device resembles an LCD class-room projector.) The number of each row in this series of images is represented using the Gray code. (See ➲ *Fig. 40.46b*. Despite its name, this generates a series of *2-level* illumination patterns.) Suppose that we project *n* images in quick succession. (Typically *n* = 10, This takes ten video field periods and hence can be

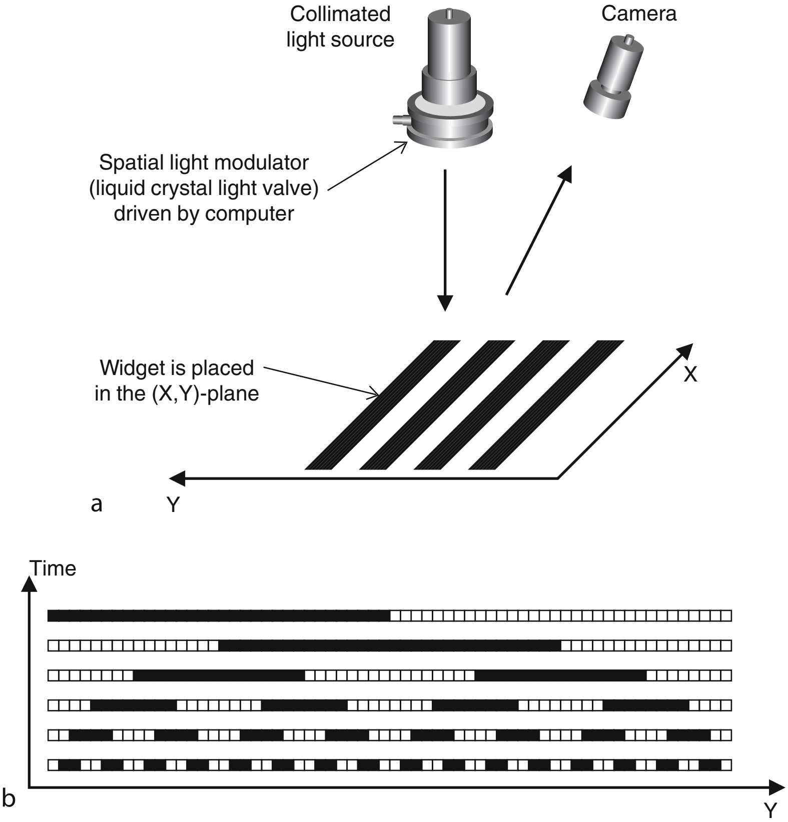

Collimated light source
Camera
Spatial light modulator (liquid crystal light valve) driven by computer
Widget is placed in the (X,Y)-plane
X
a
Y
Time
b
Y

Fig. 40.46

Projecting a series of illumination patterns to measure surface height. (a) Optical arrangement. (b) Gray code lighting pattern. This requires six video frames and resolves the surface height into 64 different levels. The same pattern is projected for all X values

performed in less than half a second.) All widget points illuminated by the same line in the illumination pattern see the same (temporal) sequence of bright-dark-bright transitions. The image processing is straightforward as this requires only simple thresholding. Moreover, it can be made to be tolerant of changes in the surface reflectance, by simply noting the image brightness when each point is illuminated (intensity = $I_{on}(x,y)$) and when it is not (intensity = $I_{off}(x,y)$). We then apply spatially dependent thresholding at level $(I_{on}(x,y)) + (I_{off}(x,y))/2$ to the (x,y) pixel. After *n* images have been projected, each point is associated with an *n*-bit code, which can be interpreted using a 2^n-element look-up table, to give the surface height measurement. Suppose that we have observed the following image sequence during a sequence consisting of *n* projected illumination patterns.

$$[Q_1(x,y), Q_2(x,y), \ldots, Q_n(x,y)]$$

($Q_i(x,y) = 0$ *or* 1, for all $i = 1,\ldots, n$.) Then, the height of the surface at the position (x,y) can be inferred by noting which row in the projected pattern has the following Gray code

$$(Q_1, Q_2, \ldots, Q_n)$$

Once the row number has been deduced, the usual triangulation formula is used to calculate the surface height. Although this method requires several video frames to generate a fairly crude depth map ($<< 2^n$ different height contours), it is rather faster than some alternative structured-lighting arrangements. (Methods 24, 41 and 107) Suppose that we mistakenly infer that one of the $Q_i(x,y)$, $(i = 1,\ldots,n)$ is *0* when it should be *1*, or vice versa. The effect on the estimate of surface height is limited to an error of ±1 levels.

REFERENCES Programmable line projector. Model, LCD-1280. ABW GmbH, Frickenhausen. http://www.fh-nuertingen.de/~ag3d/daten/shortform.html. Accessed 22 Apr 2003

LAYOUT DIAGRAM ➋ *Figure 40.46*

40.50 Method 50: High Resolution View of Web

OBJECTIVE To inspect a wide continuously moving web at high speed, so that small defects can be detected.

TYPICAL APPLICATIONS Inspecting a wide variety of web and sheet products, including glass, steel, paper, plastic film, etc. and flat objects, such as bare printed circuit boards, flat-pack furniture components, plaster-board, etc.

EQUIPMENT Specially constructed camera mounting system, providing minimal overlap between the fields of view of adjacent cameras. The cameras are aligned so that they effectively form one continuous line sensor. The camera mounting bar should have a low coefficient of thermal expansion.

CAMERA Several line-scan sensors.

LENS It may be necessary to use custom lenses so that they fit within the small space available on the mounting bar.

MOTION Continuous motion, closely controlled speed.

GEOMETRY Continuous web or sheet.

MATERIALS Paper, card, steel, plastic, etc.

REMARKS In one system built in the mid-1980s, an effective cross-web resolution of about 5×10^4 pixels was achieved. Lighting was effected by a series of fibre-optic line illuminators,

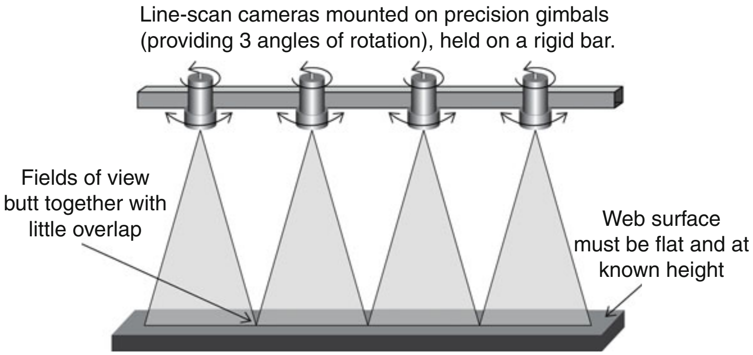

Line-scan cameras mounted on precision gimbals (providing 3 angles of rotation), held on a rigid bar.
Fields of view butt together with little overlap
Web surface must be flat and at known height

Fig. 40.47
Multiple line-scan-cameras viewing a wide web

with feedback control of light levels. At that time, to achieve the required processing speed (2.7×10^{10} MIPs) field-programmable gate arrays. (FPGAs) were used. Nowadays, several general-purpose processors, operating concurrently, would probably be used instead. Lighting was effected by a series of fibre-optic line illuminators. This system was devised for inspecting bare printed circuit boards and the associated art-work. There was minimal overlap between the fields of view of the individual cameras, which were carefully aligned using precision gimbals.
LAYOUT DIAGRAM ➔ *Figure 40.47*

40.51 Method 51: Highly Variable Ambient Lighting

OBJECTIVE To operate a vision system in high ambient lighting conditions, which may be highly variable.
TYPICAL APPLICATIONS
- Working out of doors.
- Working in situations where it is not possible to provide effective shrouding against variations in natural light levels.
- Operating a structured lighting system without shrouding ambient light.

EQUIPMENT
- Source of nearly monochromatic light, e.g., laser, high intensity sodium vapour lamp.
- Narrow pass-band filter, matched to the light source.

CAMERA Monochrome – there is no ability to sense colours.
MATERIALS The spectral response of the object to the particular wavelength of light emitted from the lamp must be high.
REMARKS This technique relies on the fact that ambient light is normally panchromatic and hence contains a relatively small amount energy within the narrow pass band of the filter. On the other hand, a monochromatic light source such as a sodium vapour lamp emits most of its output energy within the filter pass-band. Since the illumination is monochromatic, it is impossible to sense different colours. An alternative is to use pulsed lighting. Two frames are

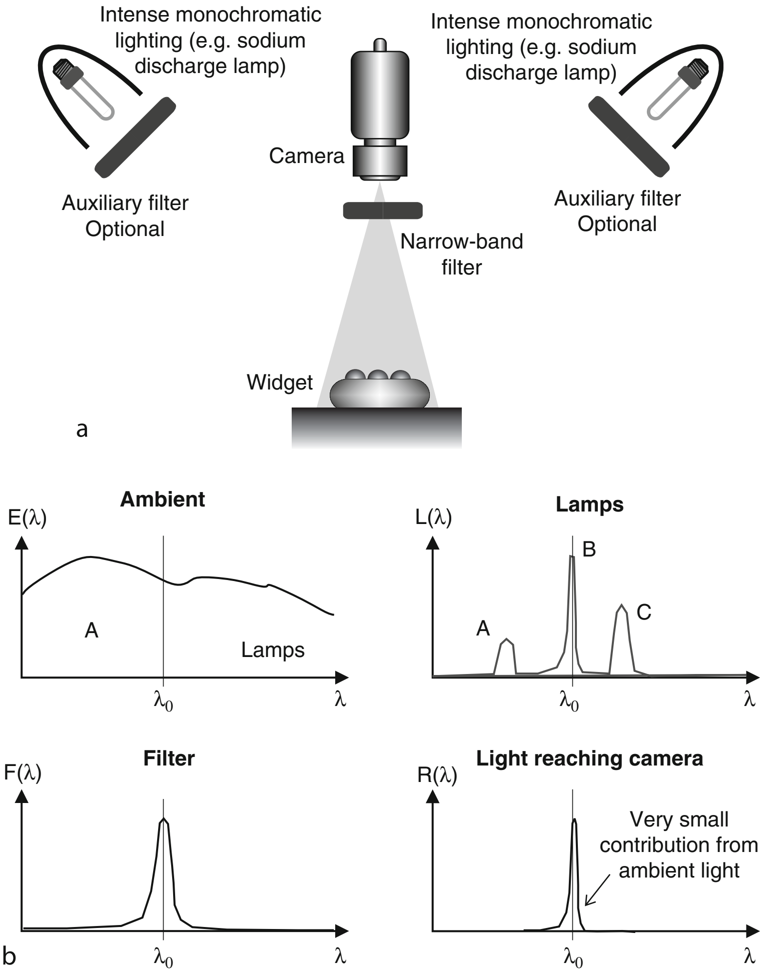

Intense monochromatic lighting (e.g. sodium discharge lamp)
Intense monochromatic lighting (e.g. sodium discharge lamp)
Camera
Auxiliary filter Optional
Auxiliary filter Optional
Narrow-band filter
Widget
a
Ambient
E(λ)
A
Lamps
λ0
λ
Lamps
L(λ)
B
A
C
λ0
λ
Filter
F(λ)
λ0
λ
b
Light reaching camera
R(λ)
Very small contribution from ambient light
λ0
λ

Fig. 40.48

Viewing in highly variable ambient lighting. (a) Optical arrangement. Auxiliary filters might be used to block infra-red, or secondary peaks in the lamp-emission spectrum. (b) Spectral characteristics of the ambient lighting (E(λ)), lamps (L(λ)) and filter (F(λ)). The spectral composition of the light reaching the camera is given by the following expression. [E(λ) + L(λ)].W(λ).F(λ) where W(λ) is the spectral reflectivity function of the widget

captured in quick succession, one with the lights on and the other with the lights off. The resulting images are then subtracted. The difference image shows the effects of the illumination, as if there were no ambient light. The applied illumination must be of comparable brightness to that of the ambient light. The camera should not have automatic gain control (AGC) switched on.

LAYOUT DIAGRAM AND FILTER CHARACTERISTICS ➲ *Figure 40.48*

40.52 Method 52: Homogenising Illumination

OBJECTIVES To provide uniform illumination, eliminating variations in intensity due to lamp deficiencies and limitations.
TYPICAL APPLICATIONS Providing uniform lighting over a large area is crucial for the successful solution of a very wide range of applications including print inspection, analysis of surface texture, etc.
EQUIPMENT Diffuser, which may use a surface effect (e.g., ground glass), dispersion within a translucent material, "randomised" fibre optic bundle, scattering from a roughened surface, or diffractive optics.
REMARKS Many of the most important lighting-viewing methods described in this chapter (e.g., front, back and dark-field illumination) rely upon the provision of highly uniform diffuse lighting over a large area. It is often impossible to provide this with a naked light source; some sort of diffuser is required between the lamp(s) and the widget. Filament lamps, discharge tubes and LEDs do not generate homogenous beams of light (➋ *Fig. 40.49b*). In the past, diffusers almost always depended upon light being dispersed as it travels through a sheet of translucent material but, nowadays, fibre-optic, diffractive or reflective diffusers may be preferable in some situations (➋ *Fig. 40.49a*). The traditional method of producing diffusers uses etched, shot-blasted or ground surfaces, while translucent plastic sheeting ("opal") uses material that contains numerous tiny particles that disperse the light and provides a much cheaper alternative for large optical systems. A "randomised" fibre-optic bundle can also be used to produce a diffusing effect. However, very much more effective homogenisation can often be achieved using a diffraction plate. Diffusers of this kind can be made in the form of flexible or rigid sheeting. They can do much more than simply homogenise the light projected onto the widget. For example, they can shape the output beam so that its cross-section is an ellipse, circle, rectangle or square. One manufacturer produces diffractive diffusers that can generate output beams that have:

(a) Narrow, medium or wide angles (1°–70°).
(b) Elliptical cross-section (ratio of longest to shortest axis up to 50:1).

Diffusers are also needed sometimes for homogenising the micro-variations in the light output from fibre optic illuminators.
REFERENCES Physical Optics Corporation, Torrance. http://www.poc.com/lsd/default.asp
LAYOUT DIAGRAMS AND SAMPLE IMAGES ➋ *Figure 40.49*

40.53 Method 53: Hypercentric (Pericentric) Lens

OBJECTIVE To obtain top and an all-round side views of an object (The bottom is not viewed).
TYPICAL APPLICATIONS Inspecting cylindrical objects and mechanical components.
EQUIPMENT Hypercentric lens.
CAMERA Single array camera.
MOTION Static.
GEOMETRY Approximately cylindrical.
MATERIALS Not critical.

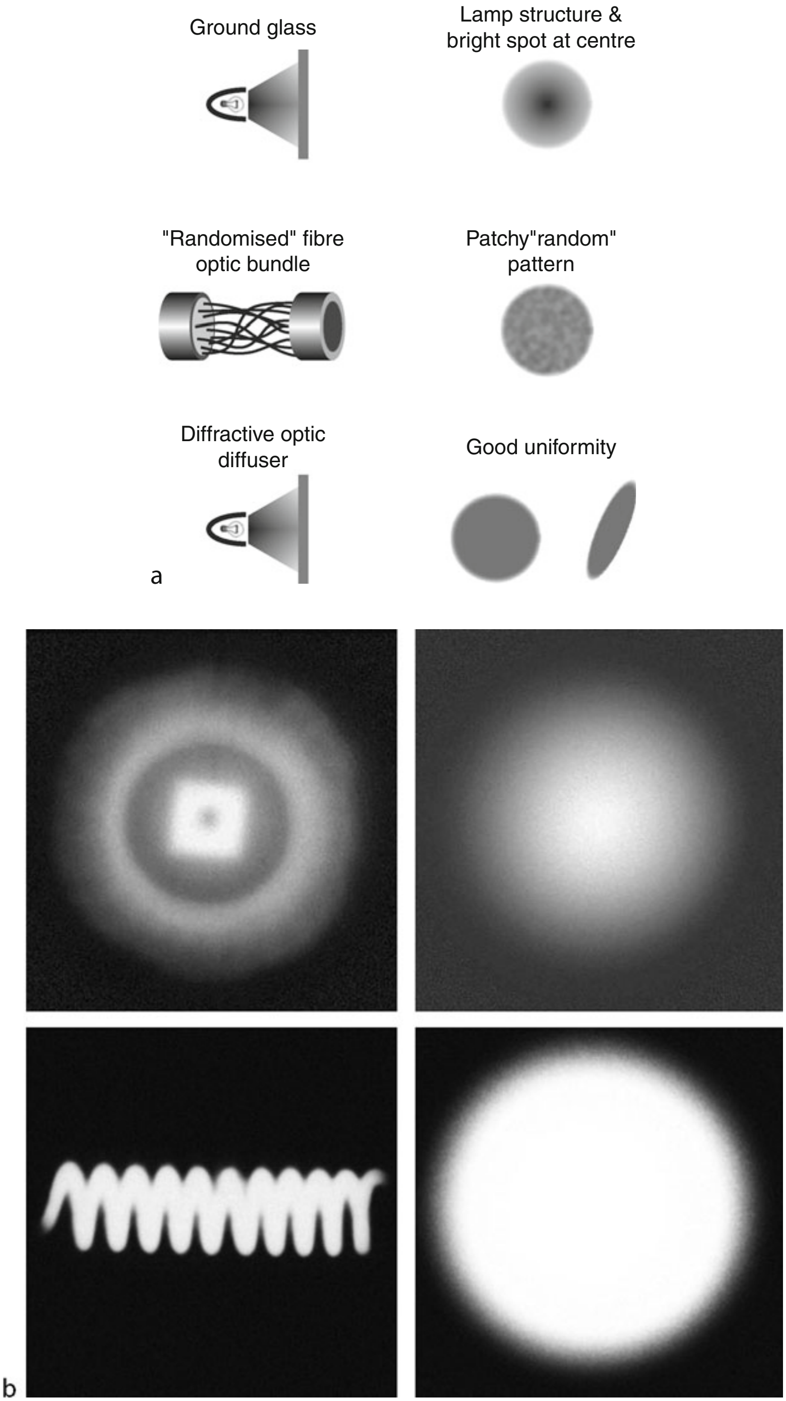


Fig. 40.49
(Continued)

REMARKS A hypercentric lens, alternatively called a pericentric lens, provides a converging view of an object. This allows the top and sides of an object such as a cube or cylinder to be viewed simultaneously (Recall that a fish-eye lens provides a diverging view). A hypercentric lens receives rays of light as if they all originate from a single point, the convergence point.
REFERENCES Hypercentric lens. Light Works, LLC. http://www.lw4u.com. Accessed 2 Feb 2011
LAYOUT DIAGRAM ➋ *Figure 40.50*

40.54 Method 54: Illuminate Top Surface Only

OBJECTIVE To illuminate the top surface of an extrusion without projecting light onto the background.
TYPICAL APPLICATIONS Inspecting extruded flat topped cakes, confectionery bars, food and other products of variable height.
EQUIPMENT Source of collimated light with well defined beam width.
CAMERA Line scan camera (may use colour).
MOTION Continuous linear motion.
GEOMETRY Nearly flat top, height variable but held within closely controlled limits.
REMARKS A collimated beam of defined width is projected onto the surface, at an inclined angle. Provided the top surface of the extrusion varies within certain limits, it remains well lit, within the camera's very narrow field of view. However, the top of an under-height or over-height extrusion is dark. The background is also dark. It is possible to use two light sources, one shining down-stream and the other up-stream, to avoid casting shadows around step-like features.
LAYOUT DIAGRAM ➋ *Figure 40.51*

40.55 Method 55: Illuminating a Wide Moving Web

OBJECTIVE To provide a thin band of light on a continuously moving web (i.e., a product made in the form of a flat or nearly flat sheet).
TYPICAL APPLICATIONS Inspecting fabric, printing, dough rolled into a sheet, plastic film, sheet glass, floor and wall tiles, wall panels, flat panels for domestic kitchen equipment (refrigerator, cookers, etc.), printed circuit boards.

Fig. 40.49
(a) Techniques for homogenising illumination (b) Sample images: *Top-left:* Output from a LED is non-uniform and clearly shows details of its internal structure; *Top-right:* Homogenised beam produced from the LED by a diffractive diffuser; *Bottom-left:* The beam from an incandescent filament lamp, clearly showing the details of the hot wire coil; *Bottom-right:* Homogenised beam produced by a diffractive diffuser applied to the light from the filament lamp. Notice that this device also adjusts the aspect ratio of the beam cross-section. A similar beam-stretching effect applied to the LED would, of course, produce a beam with an elliptical cross-section, which might be desirable in some circumstances. It is perfectly feasible to adjust the eccentricity of the beam profile

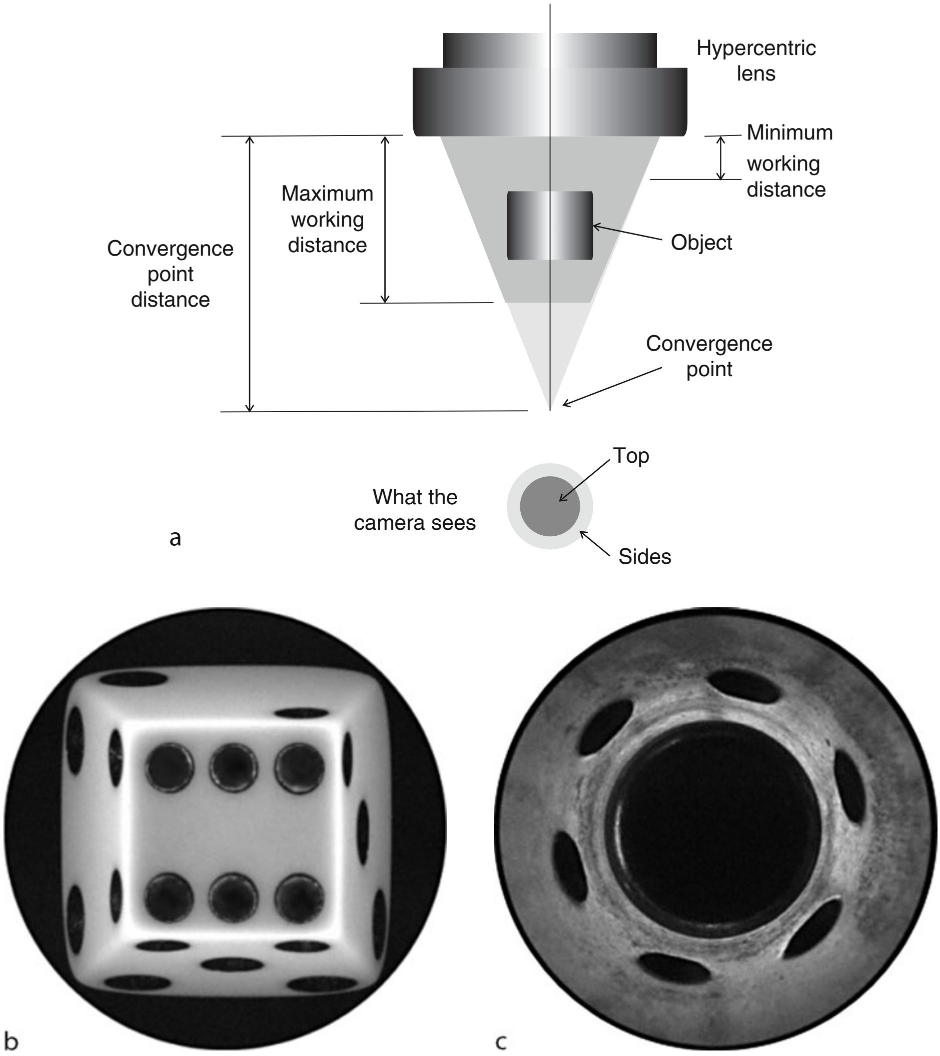


Fig. 40.50
Hypercentric lens (a) Viewing cone. (b) Gaming die. (c) Cylinder with six side ports [Images kindly supplied by Spencer Luster, Light Works. http://www.lw4u.com]

EQUIPMENT One of the following:

(1) Fluorescent tube(s).

(2) Glass/perspex (plexiglass) rod, coated with white paint, except for a narrow strip, which is kept clear. The end faces are optically polished. A lamp is placed at one end of the rod and a sheet of retro-reflective material at the other (This can be purchased from any automobile spares supplier). Alternatively, lamps can be placed at both ends.

(3) Tube formed by rolling a sheet of *totally internally reflecting film (TIRF)*.

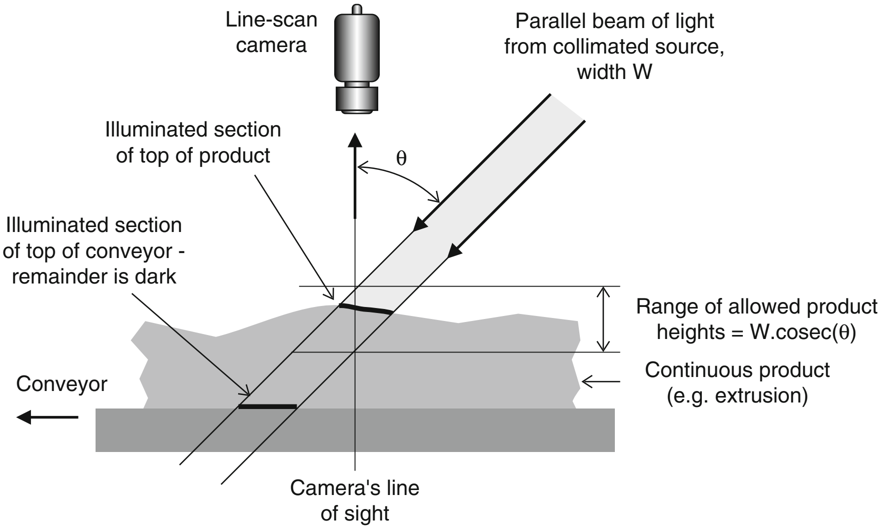

Line-scan camera
Parallel beam of light from collimated source, width W
Illuminated section of top of product
θ
Illuminated section of top of conveyor - remainder is dark
Range of allowed product heights = W.cosec(θ)
Continuous product (e.g. extrusion)
Conveyor
Camera's line of sight

Fig. 40.51
Illuminating an extrusion but not its background

(4) An array of LEDs, with a suitable diffuser. LED replacements for linear fluorescent tubes are now available
(5) Fibre optic cross-section converter, with a linear output aperture *(line illuminator)*.
(6) Laser fitted with cylindrical lens.

CAMERA Line scan.

MOTION Continuous linear motion. A similar system optical, used in conjunction with rotary motion, can also be devised.

GEOMETRY Approximate limits for the maximum web width are as follows:

- 2 m, using fluorescent tubes.
- 2 m, using a coated glass/plastic rod.
- 2 m, using TIRF.
- Virtually unlimited, using LEDs (10 m is perfectly feasible but expensive).
- 300 mm, using fibre optics (Cost is the limiting factor).
- About 500 mm, using a laser that is eye safe.

REMARKS There several ways to project a line of light on to a flat surface.

FLUORESCENT TUBES The light output from a fluorescent tube fluctuates markedly at a frequency equal to twice that of the mains electricity supply: 100/120 Hz. Hence, a fluorescent tube driven directly from the mains supply may well generate hum bars with a camera operating at 50/60 frames/s (*Fig. 40.52e*). Thermal inertia reduces this effect with incandescent filament lamps. For this reason, fluorescent tubes are normally driven by a high-frequency power source (typically 10–50 KHz), to avoid creating strobing effects. Sometimes, it is useful to synchronise the camera's pixel clock, the fluorescent-tube power unit and the web-drive mechanism. (This will probably require the use of a shaft-position encoder, attached to one of the web-drive rollers, and phase-lock loops, for frequency multiplication). In this way, we can ensure that, within limits, the pixel size is independent of the web

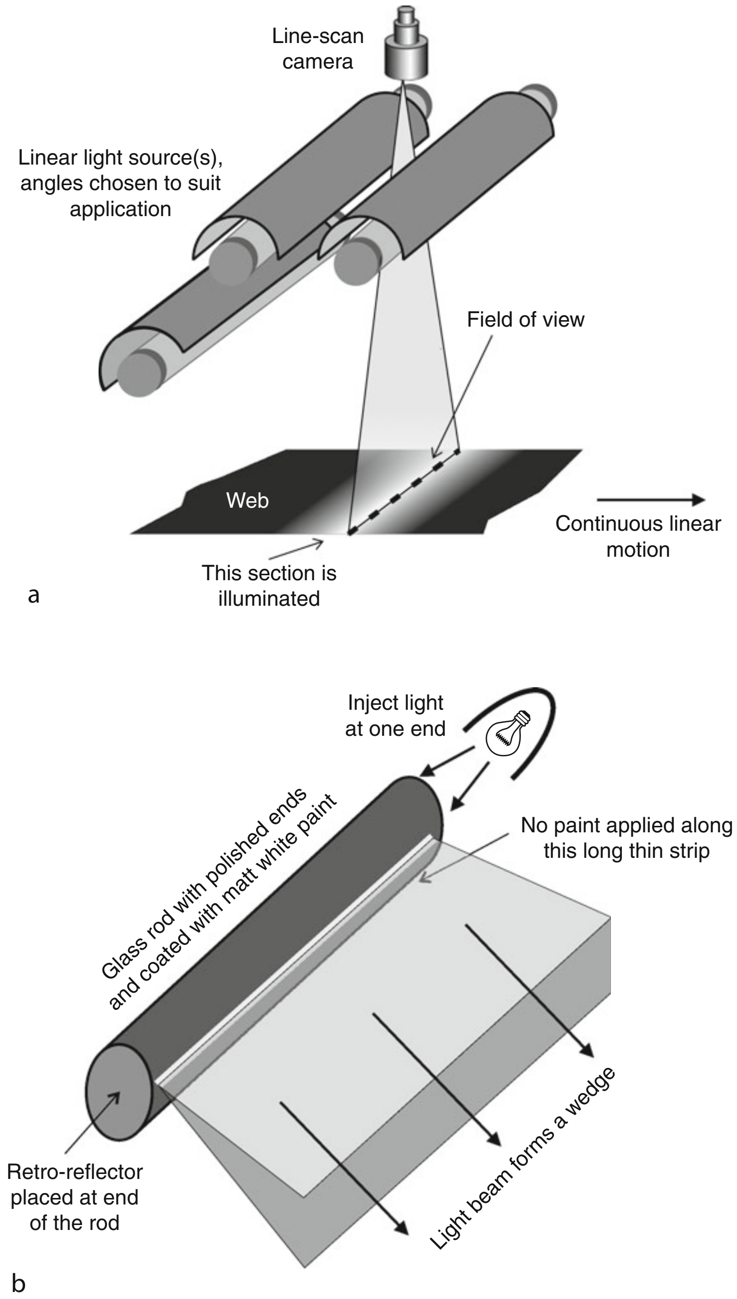

Line-scan camera
Linear light source(s), angles chosen to suit application
Field of view
Web
Continuous linear motion
This section is illuminated
a
Inject light at one end
Glass rod with polished ends and coated with matt white paint
No paint applied along this long thin strip
Light beam forms a wedge
Retro-reflector placed at end of the rod
b

Fig. 40.52
(Continued)

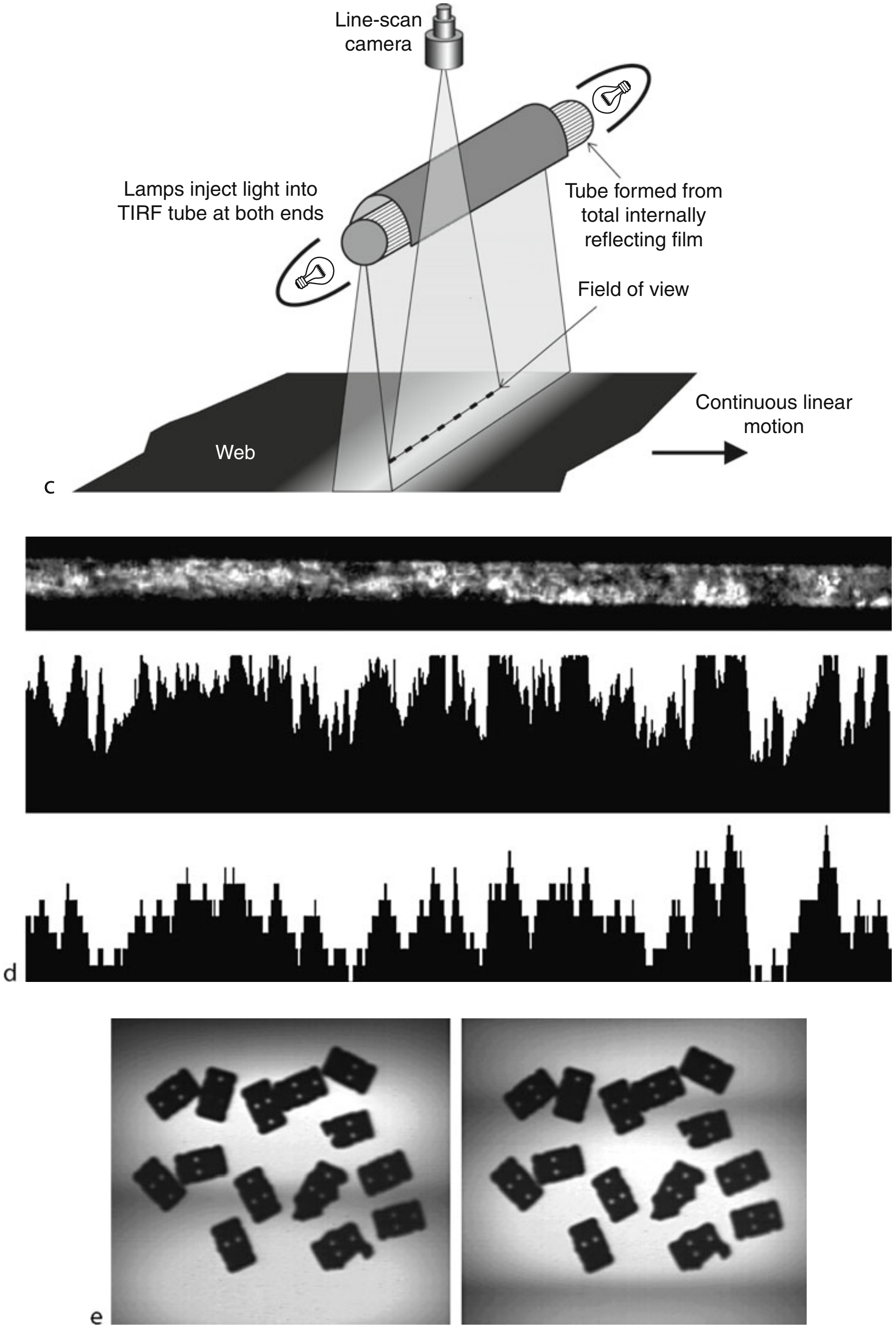

Line-scan camera
Lamps inject light into TIRF tube at both ends
Tube formed from total internally reflecting film
Field of view
Continuous linear motion
Web
c
d
e

Fig. 40.52
(Continued)

speed and thereby avoid producing hum bars. Moreover, the lighting cycle is synchronised to the web motion, so each repeated pattern (e.g., printing or embossing) is viewed under the same lighting conditions. Fluorescent lamps emit light around 360°, so it is normal to use a cylindrical reflector to limit the beam angle. Additional focussing, to generate a finer line, can be achieved, using a low-cost cylindrical Fresnel lens. It is not possible to change the colour of the projected light, except by placing a filter in the optical path (In view of the size, this would probably be a low-precision filter based on a gelatine film). The brightness of a flourescent tube is lower at the ends than the central region. It is therefore sensible not to use the end 10–15 cm for illuminating the web.

COATED GLASS ROD This method is used when it is important to produce a long thin light source that has few small-scale irregularities (When viewed at a macroscopic scale, a fibre-optic illuminator appears to produce a uniform illumination field. However, when the illumination pattern is examined more closely, irregularities can be seen, due to the finite size of the individual fibres in the matrix. The situation is worse with arrays of LEDs, since the size of the individual illumination elements is much greater). Clearly, it is important to ensure that the light output from the rod is nearly constant. Simple analysis shows how this can be achieved.

Rod illuminated at one end. Let us make the following assumptions:

(a) The rod is uniform along its whole length, X_0.

(b) As it travels along the rod, the energy within a beam of light decays exponentially, as a function of the distance travelled, *X*. (This refers to a single beam and ignores the beam returned by the retro-reflector.) Then, the light energy within the rod is proportional to

$$\exp(-A.X)$$

where the rate of light loss is determined by the parameter, *A*.

(c) The proportion of light energy returned down the rod by the retro-reflector is determined by a constant K ($K \leq 1$).

(d) All light returning to the end of the rod adjacent to the lamp is then lost.

Then, the total light energy at a distance X from the lamp is the sum of the forward and return beams and is given by S(X), where

$$S(X) = \exp(-A.X) + K.\exp(-2.A.X_0).\exp(AX)$$

To show that this produces a nearly uniform light distribution along the rod, let us substitute some typical values into this equation. Suppose that there is a 20% light loss along

Fig. 40.52

Illuminating a *straight line*. (a) Fluorescent tubes. (b) Coated glass or perspex (plexiglass) rod. In an alternative arrangement, lamps can be placed at both ends of the rod. (c) Total internally reflecting film (TIRF). Again, lamps can be placed at one or both ends of the tube. (d) Fibre optic line illuminator. *Top:* Close-up view of the ends of the fibres. Notice how uneven the light pattern is. *Centre:* intensity integrated vertically. This indicates the line intensity profile that would be achieved if a cylindrical lens were used to focus the light onto a plane. *Bottom:* Simple intensity profile. This is what a perfect line-scan camera (i.e., zero width and infinite resolution) would see if it looked directly at the fibre-optic illuminator. (e) Hum bars produced using fluorescent lighting, driven from the mains electricity supply

the rod (i.e., $\exp(-A.X_0) = 0.8$; $-A.X_0 = 0.223$) and that $K = 0.9$ (10% light loss at the retro-reflector), then

$S(0) = 1.576$; $S(X_0) = 1.520$; $S_{min} = 1.517$

(This represents −3.55% change compared to S(0). Compare this with the equivalent entry in the table below).

Light injected at both ends of the rod. The formula is revised as follows:

$$S(X) = \exp(-A.X) + \exp(-A.(X_0 - X))$$

In this case, the intensity variation is rather better, as the following change table shows:

Light fall-off parameter along the length of the rod: $\exp(-A.X_0)$	Intensity at the ends of the rod $S(0) = S(X_0)$	Intensity at the middle of the rod $S(X_0/2)$	Change[a] (%)
0.8 ($A.X_0 = 0.22$)	1.8	1.789	0.69
0.5 ($A.X_0 = 0.69$)	1.5	1.414	5.73
0.3 ($A.X_0 = 1.20$)	1.3	1.095	15.77
0.1 ($A.X_0 = 2.3$)	1.1	0.632	42.55

[a] Values in the column labelled *Change* are calculated using the formula: $100.[S(0)-S(X_0/2)]/S(0)$

This table shows that it is important to maintain a high light level inside the rod, compared to that emitted from the slit in the coating. In other words, the slit should be as narrow as possible. It is possible to change the colour of the projected light mechanically, by inserting a filter in front of each light source.

TOTAL INTERNALLY REFLECTING FILM (TIRF) TIRF was the name given originally to a product of the 3M Company. It consists of a sheet of clear, transparent material which is smooth on one side and has a series of fine ridges on the other. The film acts like a set of parallel, lossy optical fibres, so that light injected at one end of the sheet will be scattered over its whole surface. The rolled tube thus acts as a long thin light source. To a first approximation, the brightness falls exponentially as the beam travels along the roll of TIRF. Again, it is difficult to change the colour of the projected light automatically.

LED ARRAY It is a simple matter to set up a long line of LEDs but it requires careful adjustment to obtain a uniform beam; individual LEDs must be carefully matched in colour and brightness, and they must be aligned optically. The LEDs can be of any colour, including non-VIS light. The colour can even be switched (Multi-LED modules are available containing red, green and blue diodes in a single package). In addition, LEDs can be strobed, to increase the brightness for a short period during image acquisition. Using electronic switching of individual diodes, a point of light can be made to "run along" the array. Since LEDs are fairly large (compared to individual optical fibres), a diffuser is normally needed. By placing the diffuser a little way away from the LEDs, their beams overlap, thereby improving the uniformity along the array. This may broaden the projected beam.

FIBRE OPTIC BUNDLE A fibre optic bundle can provide a very bright and very thin line illuminator. "Randomised" bundles improve the uniformity of illumination that can be achieved. Uniformity must be considered on two scales: macroscopic uniformity is usually

good but microscopic changes of brightness are troublesome (See ➋ *Fig. 40.53d*). If the fibres are of comparable size to the virtual pixels (real pixels projected backwards through the lens) artefacts, due to Moiré effects, can be produced in the digital image. Small-scale non-uniformity can be improved using a diffuser, but this spoils the main advantage of this method: the precise shape of the output aperture. As a fibre bundle is flexed, individual fibres may break, resulting in local dark spots in the line of light. The colour of illumination can be changed mechanically. (Filters have to be inserted mechanically in the optical path). This option is expensive and the length of the line is limited, compared to other methods.
LASER AND CYLINDRICAL LENS A diode laser, fitted with cylindrical or diffractive lens, is often used to project a line of light onto a widget, for 3D measurement (i.e., range/depth map acquisition. See Method 107) Using the same optical arrangement here is tempting but not always successful. Firstly, it produces speckle patterns which cause brightness variations in the digital image (For range measurement, speckle is not important, because the technique relies on measuring the position, but not the intensity, of the projected line). Secondly, the projected line is very thin and can easily become misaligned with the camera's field of view. Thirdly, it is a good idea to avoid using lasers if at all possible, on the grounds of safety. Fourthly, it is not possible to change the colour of the line of light, since this is determined by the laser.
REFERENCES Total internally reflecting film. 3M Company, St. Paul, Minnesota, USA
DIAGRAMS ➋ *Figure 40.52*

40.56 Method 56: Illumination within a Hollow Cylinder

OBJECTIVE To provide a source of diffuse illumination around and above a shiny object, so that surface relief features can be seen.
TYPICAL APPLICATIONS Coins, engraved, embossed objects, cracks on flat surfaces.
EQUIPMENT One or more ring lights projecting light inwards. Diffusing material in the form of a hollow tube.
CAMERA Array.
MOTION Static.
GEOMETRY Embossed or engraved.
MATERIALS Bright shiny metal, glass, plastic or ceramic.
REMARKS This is a convenient alternative to the omni-directional light source (Method 85). The height of the diffusing tube can be adjusted to suit particular applications.
DIAGRAMS AND SAMPLE IMAGE ➋ *Figure 40.53*

40.57 Method 57: Injecting Light into Glassware

OBJECTIVE To view edge defects on a transparent plate or sheet of clear glass or plastic.
TYPICAL APPLICATIONS Detecting chipping on the edges/corners of glass or plastic sheeting.
EQUIPMENT One, two or more light sources. To inspect sheet material, 'linear' sources would probably be used. A fibre-optic bundle with a "linear" aperture, or a line of LEDs would be appropriate. A laser, fitted with a cylindrical lens and generating a fan-shaped beam, can also be used.
CAMERA Array.
MOTION Static.

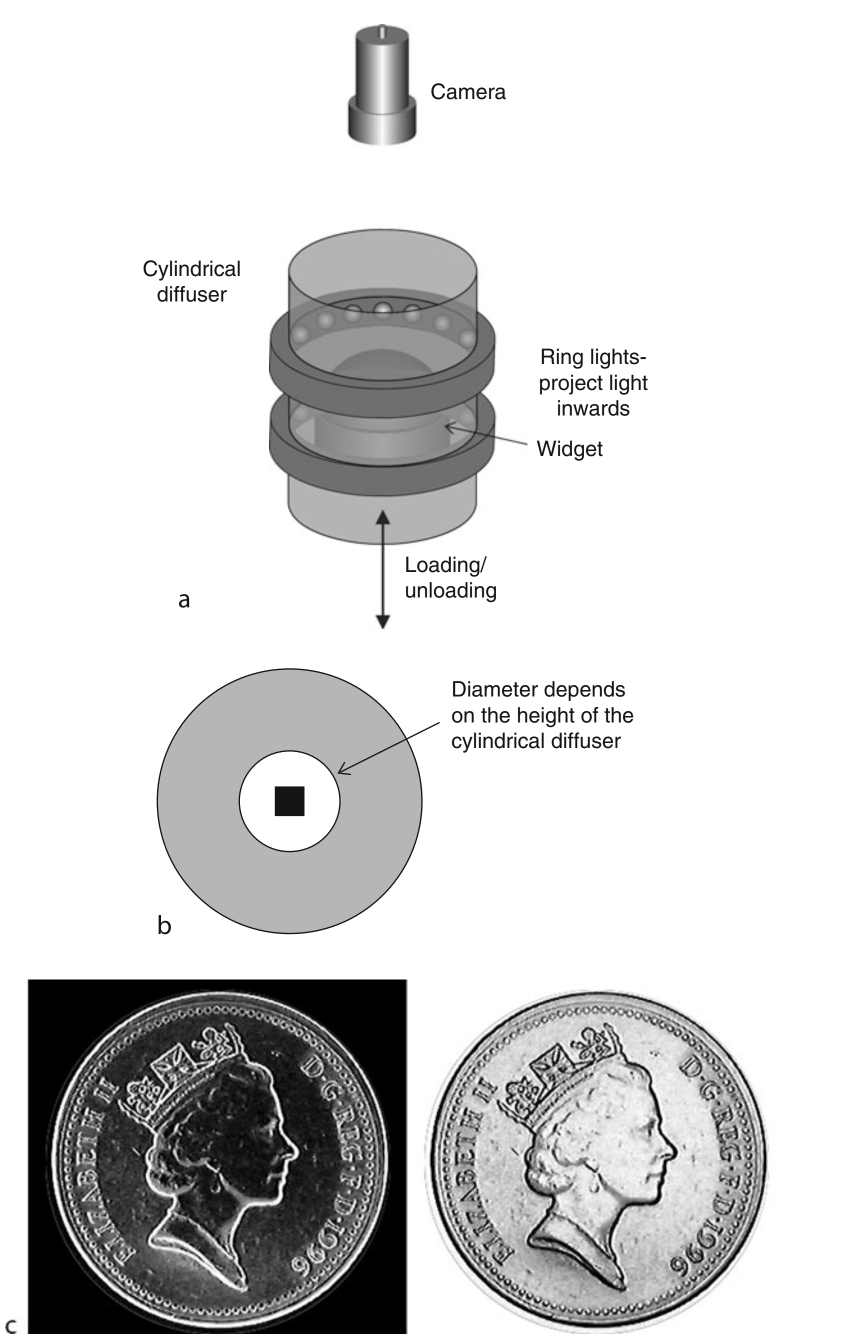

Camera
Cylindrical diffuser
Ring lights-project light inwards
Widget
Loading/unloading
a
Diameter depends on the height of the cylindrical diffuser
b
c
ELIZABETH II D·G·REG·F·D·1996
ELIZABETH II D·G·REG·F·D·1996

Fig. 40.53

Illumination within a hollow cylinder. (a) Optical layout. (b) Illumination diagram. (c) Sample image: Polished ‘‘copper’’ coin. *Left:* Original image. *Right:* Negative image, shows more detail

MATERIALS Transparent glass or plastic.

REMARKS Light is injected into the transparent sheet along one or more edges. The light is transmitted by total internal reflection through the sheet, approximately parallel to the surface, until it arrives at a point where it is damaged. The jagged surface of a chip scatters light through a wide range of angles. It may be necessary to experiment with the placement and angle of the light sources to obtain the best effect. This method is also able to highlight cracks in the glass, provided they are approximately orthogonal to the path of the light through the glass. This method can be modified to inspect the rim of a circular object, such as a the mouth of a bottle, jar or drinking glass. In this case, a ring light is used to project light into the rim.

DIAGRAMS ➋ *Figure 40.54*

40.58 Method 58: Inspecting a Flat Moving Web Using a Laser Scanner

OBJECTIVE To inspect a flat web, providing very high resolution.

TYPICAL APPLICATIONS Inspecting magnetic recording surfaces, steel strip, plastic film, and wide variety of other "strip" products.

EQUIPMENT Flying-spot laser scanner, with a rotating multi-facet mirror, or a mirror mounted on a galvanometer. One or more photo-multipliers may be used to measure the scattered light. A parabolic-cylindrical mirror is optional.

CAMERA None, image is constructed from video signal generated by the photo-detector.

MOTION Web is moving continuously, at known speed.

GEOMETRY Flat.

MATERIALS Smooth, may be matte or shiny.

REMARKS Flying-spot laser scanners are able to provide high resolution sensing of a wide, fast-moving web. A continuous digital "image" is generated; it is not segmented into distinct frames. Since the reflected/scattered light from a single laser beam can be analysed by several photo-detectors simultaneously, it is possible to generate multi-channel images that combine both near-specular and non-specular components (Each channel is monochromatic). The mechanical scanning devices shown in ➋ *Fig. 40.55b* and *c* both steer the beam over a fan-shaped region of space. Hence, the angle of incidence varies across the web. This can be corrected using a parabolic-cylindrical mirror. Using the arrangement shown in ➋ *Fig. 40.55d*, the cross-web deflection of the beam is proportional to

$$[1 - \cos(\theta)]/\sin(\theta)$$

where θ is the angle of the scanning mirror. (θ is a linear function of time for the rotating-mirror scanner). Hence, θ should be kept small to ensure regular spatial sampling across the web (Typically, $-30^\circ \leq \theta \leq 30^\circ$). For many web-inspection applications, particularly those requiring only the detection of spot-like defects, scan linearity is unimportant. Laser scanners are better suited to this type of application, rather than precise metrology. Other scanning methods exist, including Bragg cells (acousto-optic modulators) and piezo-mirrors.

REFERENCES Batchelor BG, Hill DA, Hodgson DC (eds) (1985) Automated visual inspection. IFS, Bedford. ISBN 0-903608-68-5. Also published by North Holland, Amsterdam, 1985, ISBN 0-44487577-98

Piezo mirror tilters. Physik Instrumente (PI) GmbH, Co.KG, Karlsruhe. http://www.physikinstrumente. Accessed 4 Feb 2011

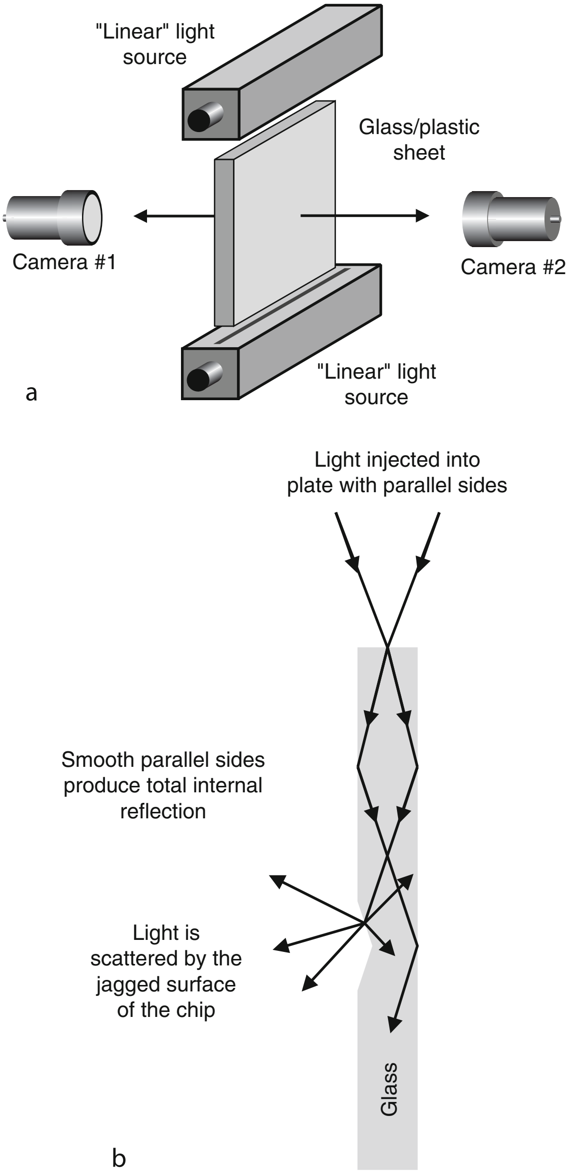

"Linear" light source
Glass/plastic sheet
Camera #1
Camera #2
"Linear" light source
a
Light injected into plate with parallel sides
Smooth parallel sides produce total internal reflection
Light is scattered by the jagged surface of the chip
Glass
b

Fig. 40.54

Injecting light into the edge of a glass sheet. (a) Optical layout. Notice that two cameras can be used on either side of the glass sheet. It may be appropriate to use an L-shaped light source to inspect the corners of a rectangular sheet. (b) Light path through the glass sheet (cross-section)

Acousto-optic modulator. Wikipedia, http://en.wikipedia.org/wiki/Acousto-optic_modulator. Accessed 4 Feb 2011

Also see ➋ Chap. 9.

LAYOUT DIAGRAMS ➋ *Figure 40.55*

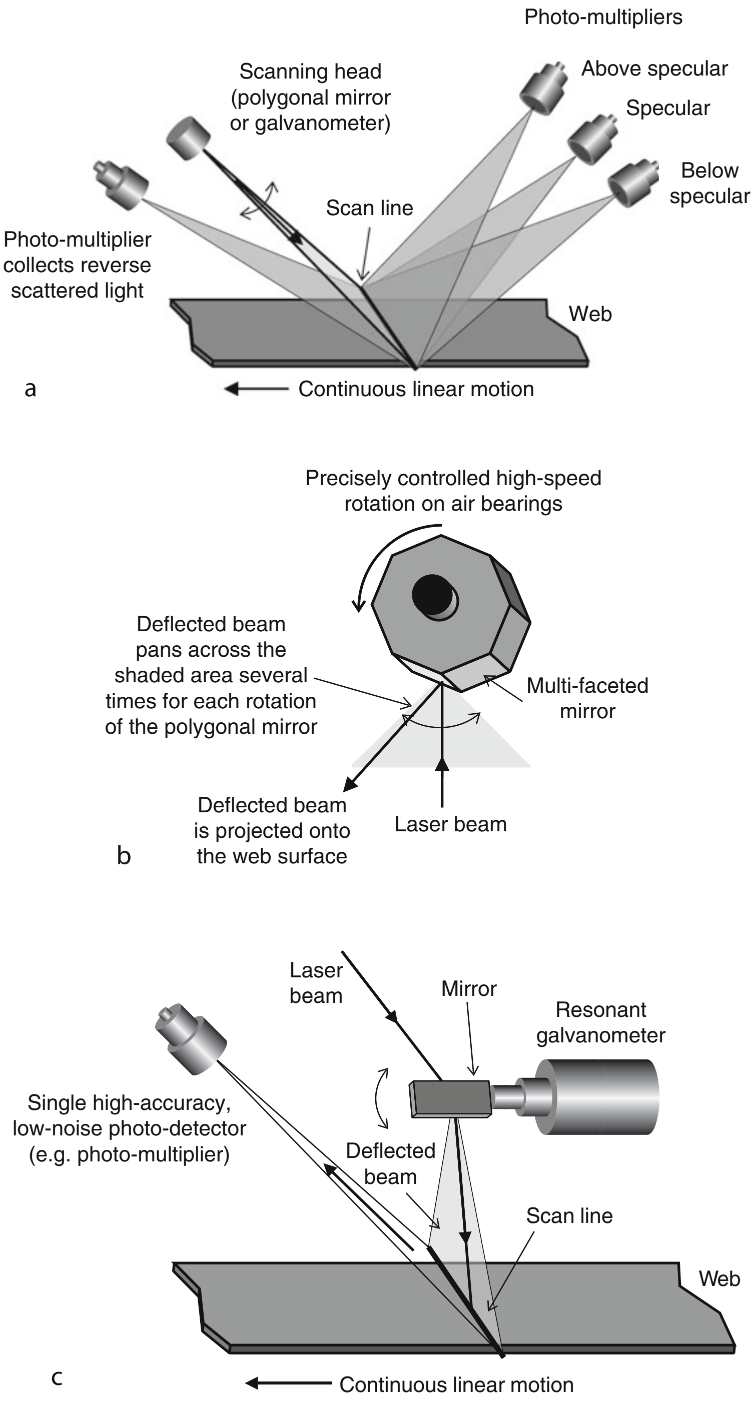


Photo-multipliers
Above specular
Specular
Below specular
Scanning head (polygonal mirror or galvanometer)
Scan line
Photo-multiplier collects reverse scattered light
Web
a
Continuous linear motion
Precisely controlled high-speed rotation on air bearings
Deflected beam pans across the shaded area several times for each rotation of the polygonal mirror
Multi-faceted mirror
Deflected beam is projected onto the web surface
Laser beam
b
Laser beam
Mirror
Resonant galvanometer
Single high-accuracy, low-noise photo-detector (e.g. photo-multiplier)
Deflected beam
Scan line
Web
c
Continuous linear motion

Fig. 40.55
(Continued)

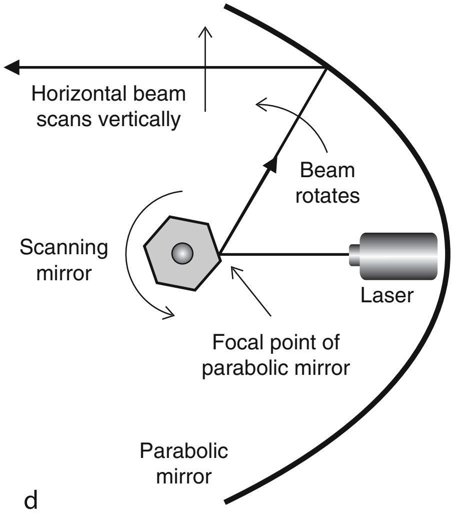


Fig. 40.55

Laser scanner for inspecting a flat web. (a) Overall view, showing various possible positions for the photo-sensor. Several photo-sensors may be used simultaneously giving a set of images that are all aligned exactly. The scanning head may be located so that the incident beam strikes the web obliquely, as shown here, or normally. (b) Using a multi-faceted mirror to deflect the laser beam. (c) Using a small, light-weight mirror, mounted on a resonant galvanometer. It may be possible to use a programmable (i.e., non-resonant) galvanometer but this does reduce the operating speed. (d) By placing the scanning head at the focal point of a parabolic mirror, it is possible to produce a scanning beam that is always parallel to the optical axis. A similar effect could, in theory, be obtained using a lens but it would be extremely expensive to scan a wide web in this way

40.59 Method 59: Inspecting a Rotating Object

OBJECTIVE To view an object while it is rotating continuously. Either the end or curved surfaces of a cylinder may be examined.
TYPICAL APPLICATIONS Inspecting objects such as

- Porcelain plates, mounted on the potter’s wheel, or decorating turn-table.
- Lathe-turned objects, while they are still mounted on the machine.
- CDs, gears, O-rings, tyres, cans, jars, bottles and numerous other circularly symmetrical components.
- Widgets that possess features such as concentric circles, radial lines (“wheel-spoke” patterns) and spirals.
- Printed, engraved, embossed features around the circumference of a disc, sphere, cylinder or cone: helices, paraxial lines, etc.

EQUIPMENT Line-scan camera. Turn-table, lathe, or other rotary mechanism.
CAMERA Line-scan.
MOTION Continuous or indexed circular motion.
GEOMETRY Circularly symmetrical components.

REMARKS The camera can view the end face (camera 1 in ➋ *Fig. 40.56a*, or the side of a cylindrical object (camera 2). Images are addressed by their polar co-ordinates.

REFERENCES Batchelor BG (1978) Circular pictures, their digitisation and processing. Comput Digital Tech Proc IEEE 1(4):179–189

LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.56*

40.60 Method 60: Internal Analysis of an Aerosol Spray or Dust Cloud

OBJECTIVES

- To measure the size distribution of microscopic, air-borne droplets or particles.
- To measure droplet/particle density in a succession of cross-section samples, taken within the same plane, at pre-determined time intervals.

TYPICAL APPLICATIONS Measuring particle/droplet size and density in coating jets, fuel/paint sprays, smoke plumes, etc.

EQUIPMENT

- The cell illustrated in ➋ *Fig. 40.57a* includes a laser and various lenses and is used for droplet/particle size analysis.
- The optical system shown in ➋ *Fig. 40.57b* is useful for determining the droplet/particle density.

In both cases, environmental-protection enclosures are necessary for the laser and camera. Air curtains can be used to keep exposed optical surfaces clean. They can also be kept dry and dust free, by electro-static means: applying a high electrical potential of appropriate polarity (Conductive glasses are made especially for this purpose and should be used in both protective windows).

CAMERA Array. It may be necessary to synchronise the camera and laser for the second method. This may require the digitisation and storage of one or more separate video frames.

REMARKS In the first method, the camera sees the far-field diffraction pattern and is insensitive to variations in the droplet/particle density across the aerosol. This yields the Fourier transform of the particle cloud and can give valuable information about the distribution of droplet/particle sizes. The second method provides a macroscopic (cross-sectional) view of the spray cone, at a number of moments in time. It is relatively insensitive to droplet/particle size but is able to provide information about their (macroscopic) spatio-temporal distribution.

LAYOUT DIAGRAM ➋ *Figure 40.57*

40.61 Method 61: Internal Features on Transparent Objects

OBJECTIVE To highlight features in transparent materials, i.e., opaque impurities, air bubbles, physical features such as cracks, crizzles (shallow cracks), scratches, embossing, ‘bird-swings’ (a filament of glass stretching across the inside of a bottle), ‘spikes’ (a ‘stalagmite’ of glass), flash, sprue, shards of glass or other loose transparent material.

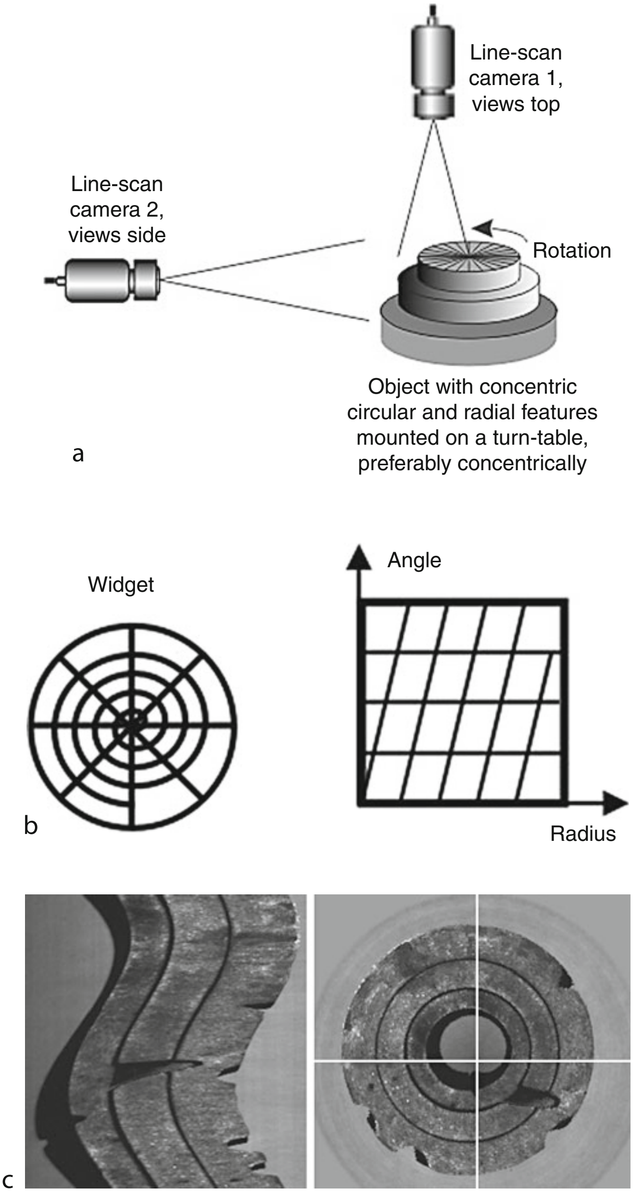

Line-scan camera 1, views top
Line-scan camera 2, views side
Rotation
Object with concentric circular and radial features mounted on a turn-table, preferably concentrically
a
Widget
Angle
Radius
b
c

Fig. 40.56

Inspecting objects on a turn-table using a line-scan camera. (a) Scanning system. (b) Image mapping, (for camera 1). Radial features are mapped into parallel *horizontal lines*. *Concentric circles* produce *parallel vertical lines* (not shown). Archimedean spirals generate parallel *inclined lines*. (c) Sample images: light-grey alloy casting. *Left:* Image derived directly from the camera. The sinusoidal ''wiggle'' is caused by the fact that the casting was not mounted concentrically on the turn-table. *Right:* By applying a Cartesian-to-polar axis transformation, the original geometry can be reconstructed. The centre of rotation is located at the middle of this image

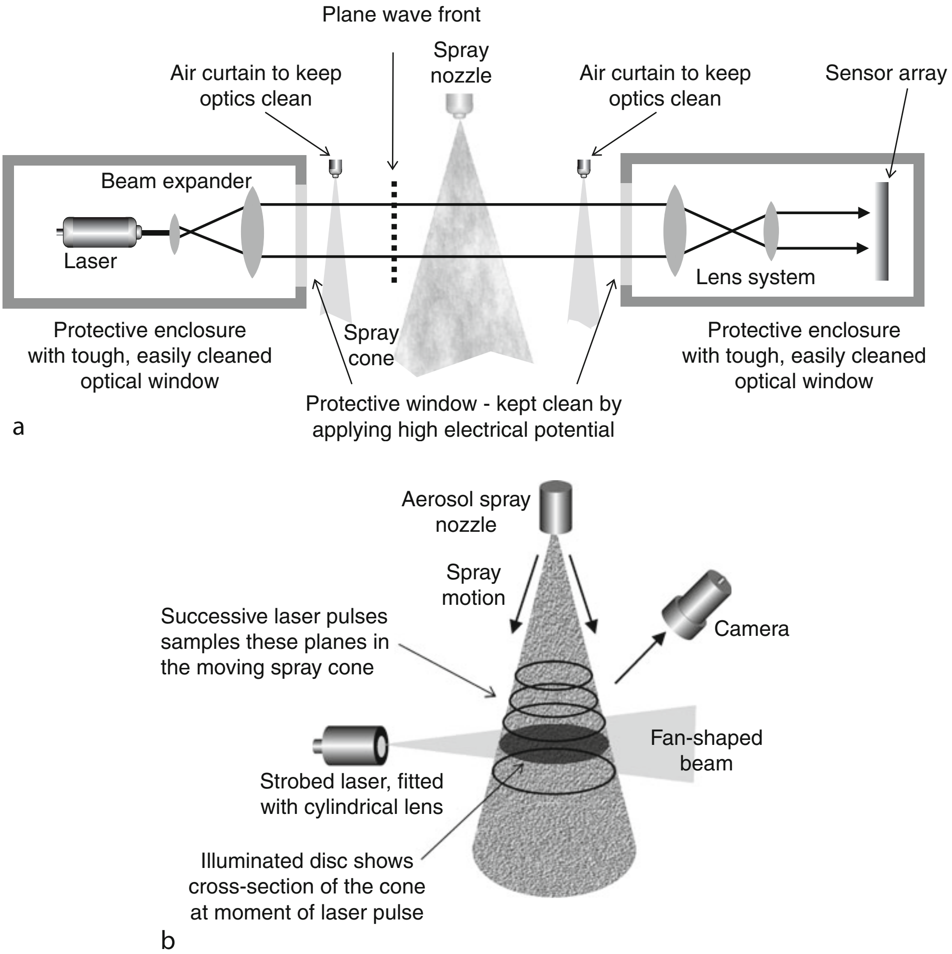

Plane wave front
Spray nozzle
Air curtain to keep optics clean
Air curtain to keep optics clean
Sensor array
Beam expander
Laser
Lens system
Protective enclosure with tough, easily cleaned optical window
Spray cone
Protective window - kept clean by applying high electrical potential
Protective enclosure with tough, easily cleaned optical window
a
Aerosol spray nozzle
Spray motion
Successive laser pulses samples these planes in the moving spray cone
Camera
Fan-shaped beam
Strobed laser, fitted with cylindrical lens
Illuminated disc shows cross-section of the cone at moment of laser pulse
b

Fig. 40.57

Analysing the internal structure of an aerosol spray. (a) Analysing aerosol droplet/particle sizes using diffraction. (b) Multi-pulse laser provides a series of cross-section samples through the spray cone. Equipment protection enclosures are not shown; these are similar to those illustrated above

TYPICAL APPLICATIONS Detecting faults in bottles, jars, tumblers, other types of glassware and clear plastics.
EQUIPMENT Back-lighting unit.
CAMERA Array or line-scan.
MOTION Static (array camera) or continuous linear/circular motion (line-scan camera).
MATERIALS Transparent/transluscent materials.
REMARKS The optical arrangement is the same as that shown for Method 17 (array camera), or Method 18 (line-scan camera). Certain features, such as embossing and cracks, may be

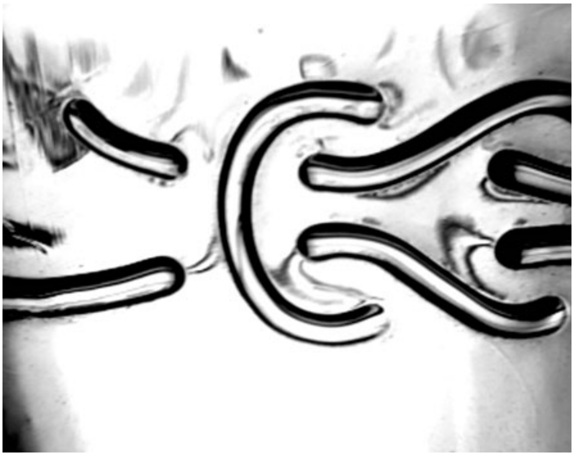


Fig. 40.58
Embossed glass bottle

seen better by this technique than by dark-field illumination (Methods 20 and 21). Visibility of a physical protusion (e.g., embossing) depends on its orientation (Does it project towards the camera, or away from it?). Similarly, the visibility of a crack depends upon its angle, relative to the lighting and camera. It is usually best to experiment with back-illumination and dark-field illumination, in order to choose the optimal lighting-viewing method for a given application.

LAYOUT AND ILLUMINATION DIAGRAMS See *Figs. 40.4a* and *40.5a*

SAMPLE IMAGE *Figure 40.58*

40.62 Method 62: Laser Bore Scanner

OBJECTIVES To inspect a blind cylindrical hole.

TYPICAL APPLICATIONS

- Inspecting bores of hydraulics cylinders and combustion cylinders for automobile engines.
- Inspecting/measuring internal (female) screw threads.

EQUIPMENT Laser bore scanner shown figure.

CAMERA None – image is constructed from the (non-standard) video signal generated by the photo-sensor.

MOTION Scanner tube moves at a controlled speed into the bore during scanning.

GEOMETRY Straight cylindrical bore.

MATERIALS Matt or shiny bore surface.

REMARKS The mirror at the end of the scanner can be rotated through 360°, thereby enabling an all-round examination of the side wall of the bore. This mirror can be oriented at 45° (for a normal view of the side-wall), or at some other angle (providing forward or back-ward views). This scanner was devised in the 1970s by SIRA Ltd., Chislehurst, Kent, U.K., for inspecting hydraulics components for automobile brake mechanisms. It was designed to detect scratches and cracks. The brake cylinder was moved linearly and the probe rotated inside it.

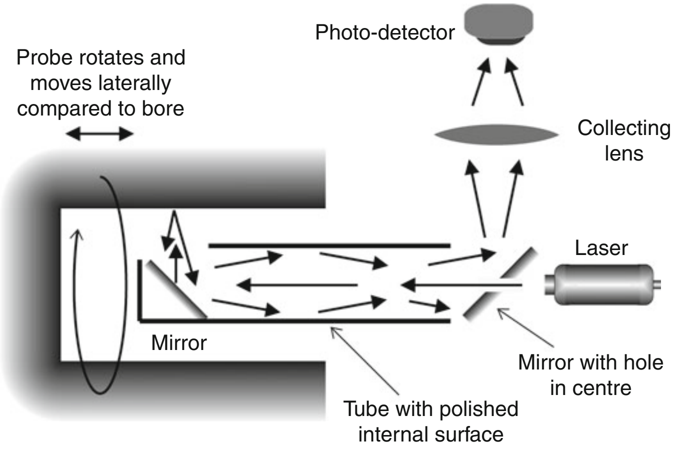


Photo-detector
Probe rotates and moves laterally compared to bore
Collecting lens
Laser
Mirror
Mirror with hole in centre
Tube with polished internal surface

Fig. 40.59
Laser bore scanner

REFERENCES

- Batchelor BG, Hill DA, Hodgson DC (eds) (1985) Automated visual inspection. IFS, Bedford. ISBN 0-903608-68-5. Also published by North Holland, Amsterdam, 1985, ISBN 0-44487577-98
- Batchelor BG, Williams GA (1979) Defect detection on the internal surfaces of hydraulics cylinders for motor vehicles. In: Proceedings of the conference on imaging applications for automated industrial inspection and assembly, Washington, DC, vol 182. SPIE, Bellingham, pp 65–77. ISBN 0-89252-10

LAYOUT DIAGRAM ➋ *Figure 40.59*

40.63 Method 63: Laser Pattern Projector

OBJECTIVE Project a simple pre-defined pattern, such as an array of spots or a grid, onto the widget to provide the basis for high-speed, low-precision measurement of 3D shape.
TYPICAL APPLICATIONS Inspecting objects such as loaves, cakes, pies and other food products, where high precision measurement of surface shape is not required but high speed of operation is.
EQUIPMENT Laser, fitted with special-purpose diffractive optical filter.
REMARKS Standard pattern generators available are:

(1) Multiple parallel stripes (up to about seven stripes).
(2) Single pair of crossed lines.
(3) Grids (about 7 by 7 lines).
(4) Multiple spots (up to about 11*11 array).
(5) Single circle.
(6) Concentric circles (up to about seven).

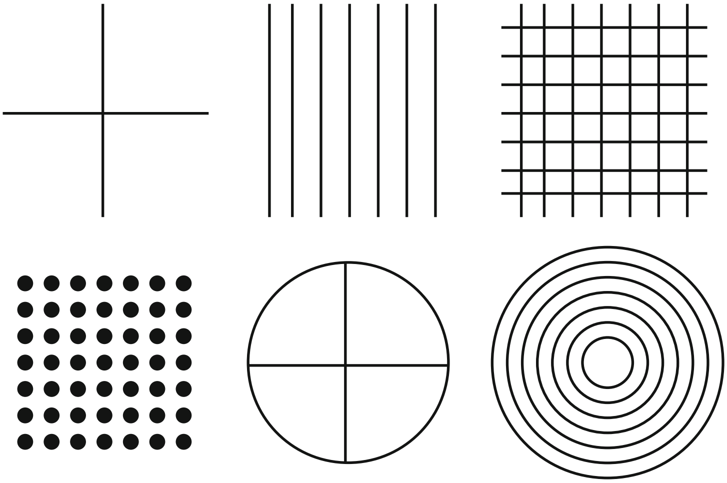


Fig. 40.60
Sample projected patterns generated by diffractive optics fitted to a low-power (diode) laser

These filters are often supplied to fit red solid-state (diode) lasers, although there is no reason why other wavelengths cannot be used. The projector is used in a structured light/triangulation lighting-viewing arrangement (Method 107).
SAMPLE PATTERNS ➲ *Figure 40.60*

40.64 Method 64: Locate Object with Zero Contrast

OBJECTIVE To obtain a silhouette of an object on a continuously moving smooth conveyor belt, when there is low contrast between the object and its background.
TYPICAL APPLICATIONS
(a) Locating con-rods (grey) on a smooth grey conveyor belt.
(b) Examining extrusions (e.g., food products) for gross variations of thickness.
EQUIPMENT Two laser stripe generators, or projectors fitted with rectangular masks.
CAMERA Line scan.
MOTION Continuous linear.
GEOMETRY Any objects whose thickness exceeds a certain minimal amount, determined by the optical set-up.
REMARKS This is the *CONSIGHT* optical arrangement, originally devised for locating con-rods on a smooth conveyor belt. When there is no widget present, the two laser fan-beams converge on the conveyor to form a single line of light, which coincides with the line-scan camera's field of view. When an object is present, the beams no longer converge and form two stripes, displaced to the sides of the camera's field of view. Hence, the camera observes a bright stripe, when no object is present but does not see any light at all when an object in place. Notice that the surface of the conveyor belt must be smooth and flat. There is no need for the object to present any colour or contrast change, compared to the conveyor belt surface.

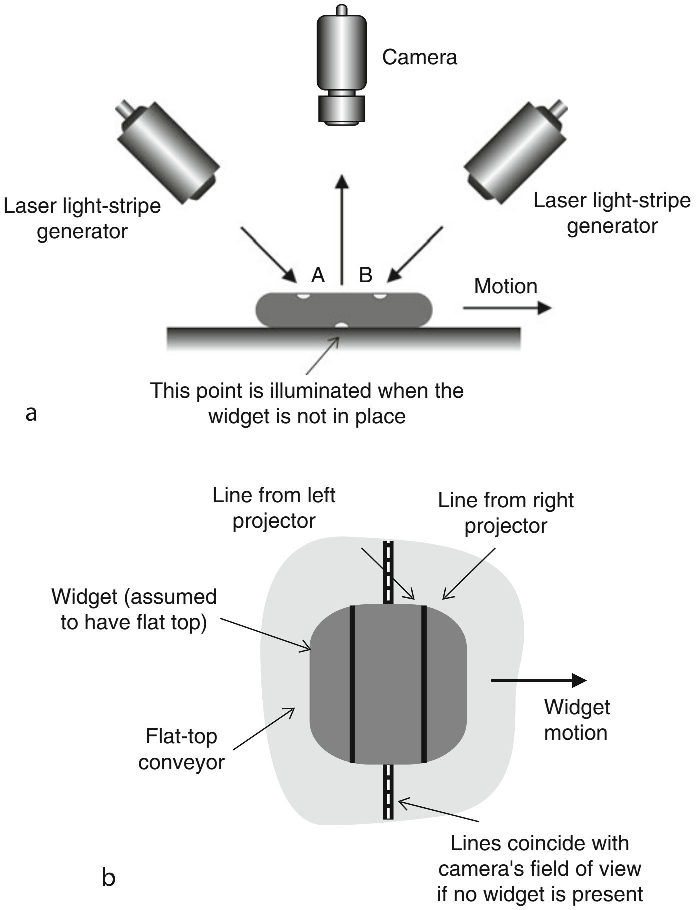

Camera
Laser light-stripe generator
Laser light-stripe generator
A
B
Motion
This point is illuminated when the widget is not in place
a
Line from left projector
Line from right projector
Widget (assumed to have flat top)
Flat-top conveyor
Widget motion
Lines coincide with camera's field of view if no widget is present
b

Fig. 40.61
Locating objects with zero contrast. (a) Optical arrangement. Both laser generate thin fan-shaped beams. (b) The camera receives no light when a widget is present (A flat-topped object is shown here for convenience of illustration)

REFERENCES Baumann RD, Wilmshurst TD (1983) Vision system sorts castings at General Motors Canada. In: Pugh A (ed) Robot vision. IFS, Bedford, pp 11–16
DIAGRAMS ➋ *Figure 40.61*

40.65 Method 65: Long Focus Optical System Using Mirrors

OBJECTIVE To provide a lens with a very long focal length (up to 2,000 mm).
TYPICAL APPLICATIONS

- Inspecting objects, or scenes, from a large distance, for reasons of safety, inaccessibility, etc.

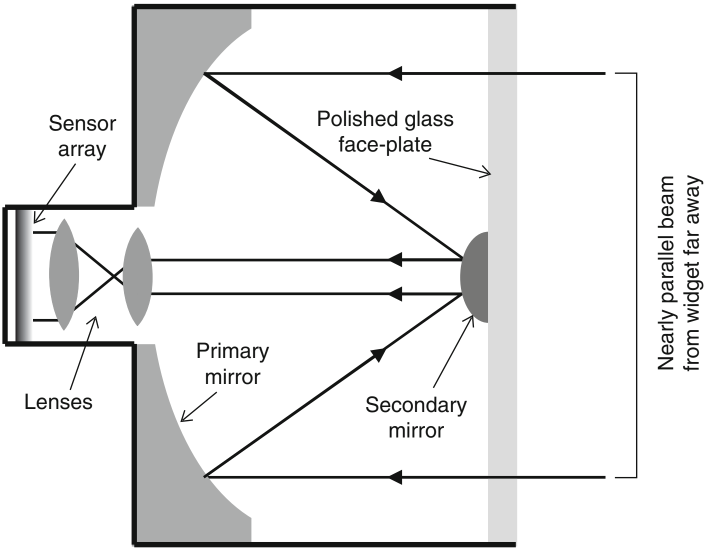

Sensor array
Polished glass face-plate
Nearly parallel beam from widget far away
Lenses
Primary mirror
Secondary mirror

Fig. 40.62
Catadioptric system

- In situ inspection of pipework, electrical insulators on pylons, brickwork, masonry, roofing, etc.
- Examining small objects/features from a large distance.

EQUIPMENT Catadioptic system. This consists of an integrated mirror and lens optical system.

CAMERA Array.

LENS Catadioptic (mirror-lens) system.

MOTION It is important to suppress vibration of the camera, since the large working distance makes this method susceptible to the slightest movement.

GEOMETRY The object or scene to be viewed can be far way away, perhaps as much a 100 m.

REMARKS Catadioptric systems are widely used in astronomical telescopes and by photographers who cherish them because they provide a long-focus, light-weight system. Reflective optical systems are capable of operating over a very wide spectral range (covering UV, VIS and IR) and are immune from certain types of aberration that afflict lens systems, especially chromatic aberration. In their advertising one company suggests that it is possible to view the whiskers of a tiger from a safe distance! Catadioptric systems do not have a wide depth of field and the field of view is much smaller than the focal length. Mis-focussing causes a point source of light to appear as a bright ring. Achieving satisfactory illumination over a large distance may also require a sophisticated mirror-lens system, since standard luminaries do not usually control the lighting angles accurately. A catadioptic system can serve a similar function to a telecentric lens.

REFERENCES Ray SF (1988) Applied photographic optics. Focal Press, London/Boston, p 267. ISBN 0-240-51226-X

LAYOUT DIAGRAM ➒ *Figure 40.62*

40.66 Method 66: Low-Definition Range Information

OBJECTIVE To obtain low-definition, 3-dimensional shape information about a widget, by simultaneously projecting a number of light stripes onto its surface.

TYPICAL APPLICATIONS Low tolerance shape-measurement applications. It is also useful in general mechanical manufacturing, for determining object type and posture. This technique is well suited to the food industry for checking the shapes/sizes of loaves, cakes, fruit, vegetables, etc.

EQUIPMENT Diode or other low-power laser, fitted with a holographic lens that generates a set of fan-shaped beams (When these are projected onto a plane, they generate a set of parallel light stripes).

CAMERA Array type.

MOTION Static. In many applications, the widget should be held in a fixed position relative to the camera.

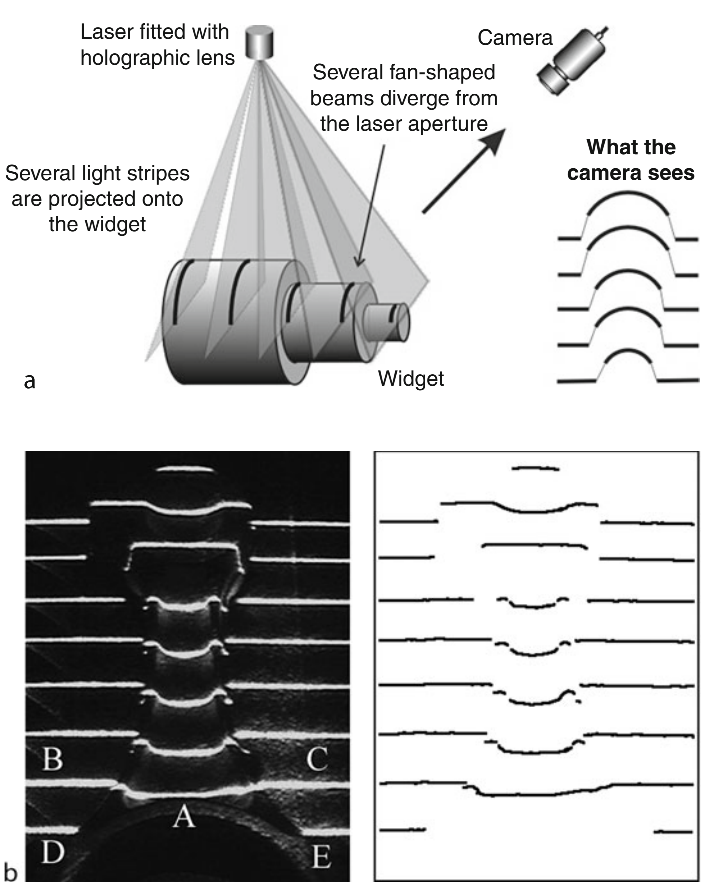
REMARKS This is similar in concept to Method 68 (Also see Method 107). It is able to provide shape information simultaneously (i.e., within one video frame period) about the height profile within several cross-sections. This is achieved by projecting several stripes at once onto the widget. A holographic lens is able to generate a small number (typically up to 12) of parallel light stripes. Hence, it is possible measure the surface height at several positions, without moving the widget. Holographic lenses can also generate cross-lines, arrays of spots, grids and concentric squares or circles (Method 63). Data analysis can be straightforward, involving only a series of simple size checks, or more complicated, requiring the use of Pattern Recognition techniques. The inputs to a neural network, or other classifier could be derived by sampling each height profile at a number of fixed points. In some situations, it may be difficult to identify the different parts of "broken" stripes, in which case we may only be able to monitor deviations from the ideal 3D shape, rather than measuring surface height quantitatively. To resolve ambiguities that might arise when the projected arcs disintegrate and merge (➲ *Fig. 40.63b*), we can use two cameras (See Method 107).

LAYOUT DIAGRAM AND SAMPLE IMAGE ➲ *Figure 40.63*

40.67 Method 67: Magnify One Image Axis

OBJECTIVES

- To make pixels square, so that rotating the image does not alter estimates of linear dimensions or areas, based on simple pixel counting.
- To make the best possible use of the available image resolution.
- To correct for oblique viewing by a camera placed off normal.
- To modify the image warping caused by anamorphic optics.
- To produce parallelogram warping, to make image processing easier. A parallelogram can be transformed into a rectangle, or a rectangle into a square.
- To adjust the virtual aspect-ratio of the photo-detectors in a line-scan sensor. Recall that the resolution across the web is determined by the camera geometry, while the web speed affects the down-web resolution. Cylindrical lenses can make the web speed appear to be greater/smaller, independently of the magnification applied across the web.


Laser fitted with holographic lens
Camera
Several fan-shaped beams diverge from the laser aperture
What the camera sees
Several light stripes are projected onto the widget
a
Widget
b
B
C
A
D
E

◘ Fig. 40.63

Low-definition range measurement using multiple laser stripes. (a) Optical layout. A low-power (diode) laser fitted with a holographic filter, projects a set of fan-shaped beams (typically 5–12) onto the widget. This should be placed in fixed position and orientation relative to the laser and camera. (b) Sample image (simulated using a single laser rotated to eight different angles in turn): automobile con-rod. *Left:* original image. The alignment of arc A with arcs B and C is accidental and is caused by the choice of projection-viewing angles; arc A is actually created by the same fan-beam as arcs D and E. This type of ambiguity places an upper limit on the number of stripes that can be accommodated by this technique. *Right:* Processed image, after filtering, thresholding and skeletonisation

TYPICAL APPLICATIONS

- To inspect long thin objects, such as nails, screws, needles, bottles, confectionary bars, machine parts, carrots, etc. where the aspect ratio may exceed 10:1.
- To correct the geometric distortion produced by a camera viewing a surface at some (known) angle other than 90°.

- To inspect print, paper, plastic film, steel sheet and other web products, as they are moving past a line-scan sensor.

EQUIPMENT One or more cylindrical lenses.

REMARKS A cylindrical lens system can be used to magnify any one axis of an image, relative to the orthogonal axis (This maps a square into a parallelogram, or a circle into an ellipse). Two lenses, mounted orthogonally to each other can magnify both axes independently, by different amounts. Notice that a cylindrical lens produces different focal planes for edges parallel to the stretched and non-stretched axes (Y- and X-axes in ➲ *Fig. 40.64a*). The reason is that rays in the XZ-plane are not deflected by the lens, while those in the YZ-plane are.

REFERENCES Ray SF (1988) Applied photographic optics. Focal Press, London/Boston. ISBN 0-240-51226-X

LAYOUT DIAGRAM AND SAMPLE IMAGE ➲ *Figure 40.64*

40.68 Method 68: Mapping Range by Projecting Multiple Stripes

OBJECTIVE To obtain low-definition 3-dimensional shape information, at high speed (one video frame period).

TYPICAL APPLICATIONS Low tolerance shape-measurement applications, particularly in the food industry. For example, analysing the shape of loaves, cakes, etc.

EQUIPMENT Ruggedised but otherwise standard slide projector. This projects a pattern consisting of a series of parallel stripes (Other patterns, such as arrays of spots, could be projected instead but images created using stripes are particularly easy to analyse). The transparency should be based on a rigid material (glass), rather than plastic or other material which is susceptible to warping.

CAMERA Array.

MOTION Static.

GEOMETRY Fairly smooth and uncomplicated, otherwise there will be some disintegration of the stripes which are difficult to “join together” again by image processing. As the angular separation between the slide projector and the camera is increased, the sensitivity becomes greater but the probability that the projected arcs will “break” also gets larger.

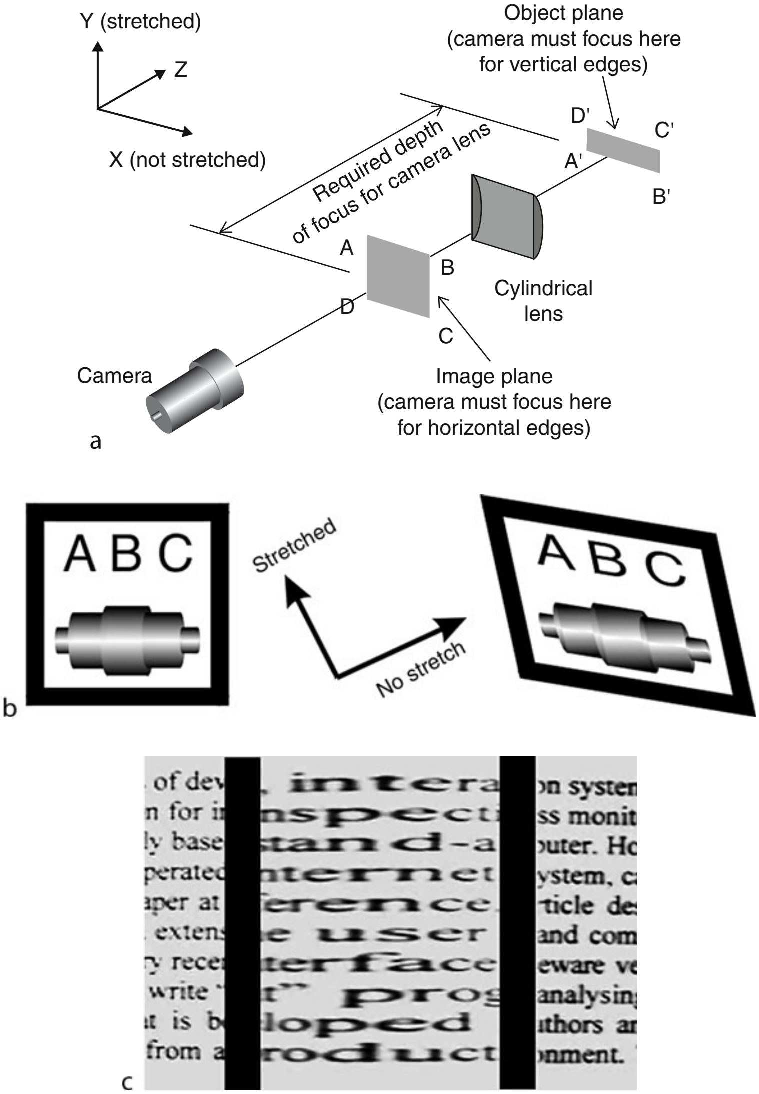
MATERIALS Matt, preferably not very strongly absorbing or reflecting.

REMARKS This is similar in concept to Method 107, but is able to provide shape information simultaneously about a number of cross-sections through the widget. This is achieved by projecting several stripes at once onto it. In some situations, it may be difficult to distinguish different stripes, in which case we may only be able to monitor deviations from the ideal 3D shape, rather than measuring surface height precisely. To avoid problems due to diffraction, the width of the stripes in the transparency must be many times the wavelength of the light used. To ensure that the projected arcs do not disintegrate or merge, due to occlusion, we can use two cameras. The system is then symmetrical about the projector’s optical axis.

LAYOUT DIAGRAM AND SAMPLE IMAGE ➲ *Figure 40.65*

40.69 Method 69: Mapping Range with Polychromatic Light

OBJECTIVE To obtain low-definition height information quickly, as a prelude to forming a range map (or depth map) from an object surface.



■ **Fig. 40.64**

Cylindrical lens. (a) Optical layout. Horizontal magnification = 1, (Hence [AB] = [A′B′]); Vertical magnification = M, where M = [AD]/[D′A′]. Notice that the vertical axis is inverted but the horizontal axis is not. (b) Parallelogram warping using a cylindrical lens. *Left:* Original. *Centre:* Two vectors are needed to characterise the magnification of a cylindrical lens. *Right:* Resulting image. Notice that the process is reversible, so these two images could be labelled in the reverse order. (c) Sample image: *Centre:* View of a printed page through a plastic Fresnel lens. *Sides:* Direct views

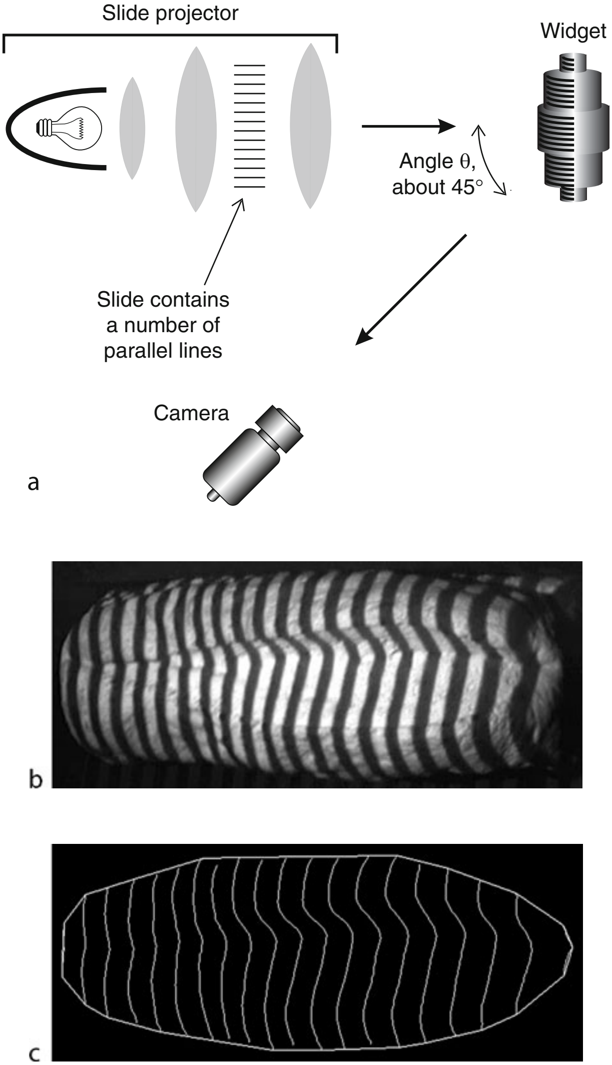


Fig. 40.65

Estimating surface height by projecting light stripes onto an object. (a) Optical arrangement. (b) Sample image: Loaf, original image. (c) Processed image. Each light stripe has been reduced to an arc one pixel wide. This makes subsequent analysis easier

TYPICAL APPLICATION Monitoring shape of loaves, cakes, etc.
EQUIPMENT Various options are available:

(a) Intense collimated source of white light, diffraction grating.
(b) Array of LEDs emitting at different wavelengths.
(c) Linear variable interference filter (At each point along the filter, it passes a narrow band of wavelengths. The wavelength at the centre of the pass-band varies across the face of the filter).

CAMERA RGB or HSI colour camera with array- or line-scan sensor geometry (An HSI camera is easier to use).
MOTION Static (array camera), or continuous linear motion (line-scan).
GEOMETRY Fairly simple; should not have steep-sided pits or cliffs.
MATERIALS Opaque, matt, preferably not strongly coloured, non-fluorescent. Surface, diffraction effects should be negligible.
REMARKS This method uses colour contours, instead of sharp illumination boundaries, in an otherwise conventional structured-lighting arrangement (Methods 107 and 68). If an RGB camera is used, its output signal is digitised and then passed through a look-up table to "calculate" hue. In the LUT, each [R,G,B] triple is associated with a single scalar quantity proportional to hue (H). If an HSI camera is used, its hue channel is used directly, while saturation (S) and intensity (I) are both ignored. By thresholding the hue at various levels, a range of contours can be produced, from which the height can be determined (See the inset in *Fig. 40.66a*). The geometry is the same as for Methods 107 and 68; colour contours simply replace the illumination edges.) The optical layout for a system based on an array-scan camera is shown in *Fig. 40.66a*. A "rainbow" is projected onto the widget surface. The *hue* of the light leaving a given point, P, on the object surface is not affected at all by the spectral-reflection characteristic at P, which is illuminated with only monochromatic light (Assuming that fluorescence effects are negligible, there is no light at P with any other wavelength). A standard slide projector, fitted with a variable-absorption filter (e.g., gelatine or coloured-glass) produces polychromic illumination. In this case, the measured values for hue, as well as the intensity, of the reflected light are affected by variations in the widgets' spectral reflectivity characteristic. The technique outlined in *Fig. 40.66a* requires a high-intensity light source and a highly sensitive camera, to obtain good accuracy. For this reason, it is unlikely to be suitable for inspecting large objects. An alternative scheme, based on a line-scan camera (*Fig. 40.66b*), makes more efficient use of the available light and hence is capable of being used on larger and darker objects. Among the techniques available for projecting a rainbow pattern are:

(a) A diffraction grating and an intense source of collimated white light that has a very small output aperture (*Fig. 40.66c*).
(b) An array of LEDs emitting light at different wavelengths spanning the VIS wave-band (*Fig. 40.66d*).
(c) Linear, variable band-pass interference filter, used in conjunction with a white-light projector (*Fig. 40.66e*).

REFERENCES

- Batchelor BG, Whelan PF (1993) Generalisation procedures for colour recognition. In: Proceedings of the SPIE conference on machine vision applications, architectures and systems integration II, Boston, 7–8 Sept 1993. SPIE- The International Society for Optical Engineering, vol 2064, pp 36–46. ISBN 0-8194-1329-1
- Schott *Veril* linear variable interference filters, optics and optical instruments catalog 2002. Edmund Optics. www.edmundoptics.co.uk. Accessed Nov 2002

LAYOUT DIAGRAMS *Figure 40.66*

40.70 Method 70: Measuring Range by Projecting Patterns

OBJECTIVE To project a sequence of patterns of light onto the widget so its 3D shape can be calculated.

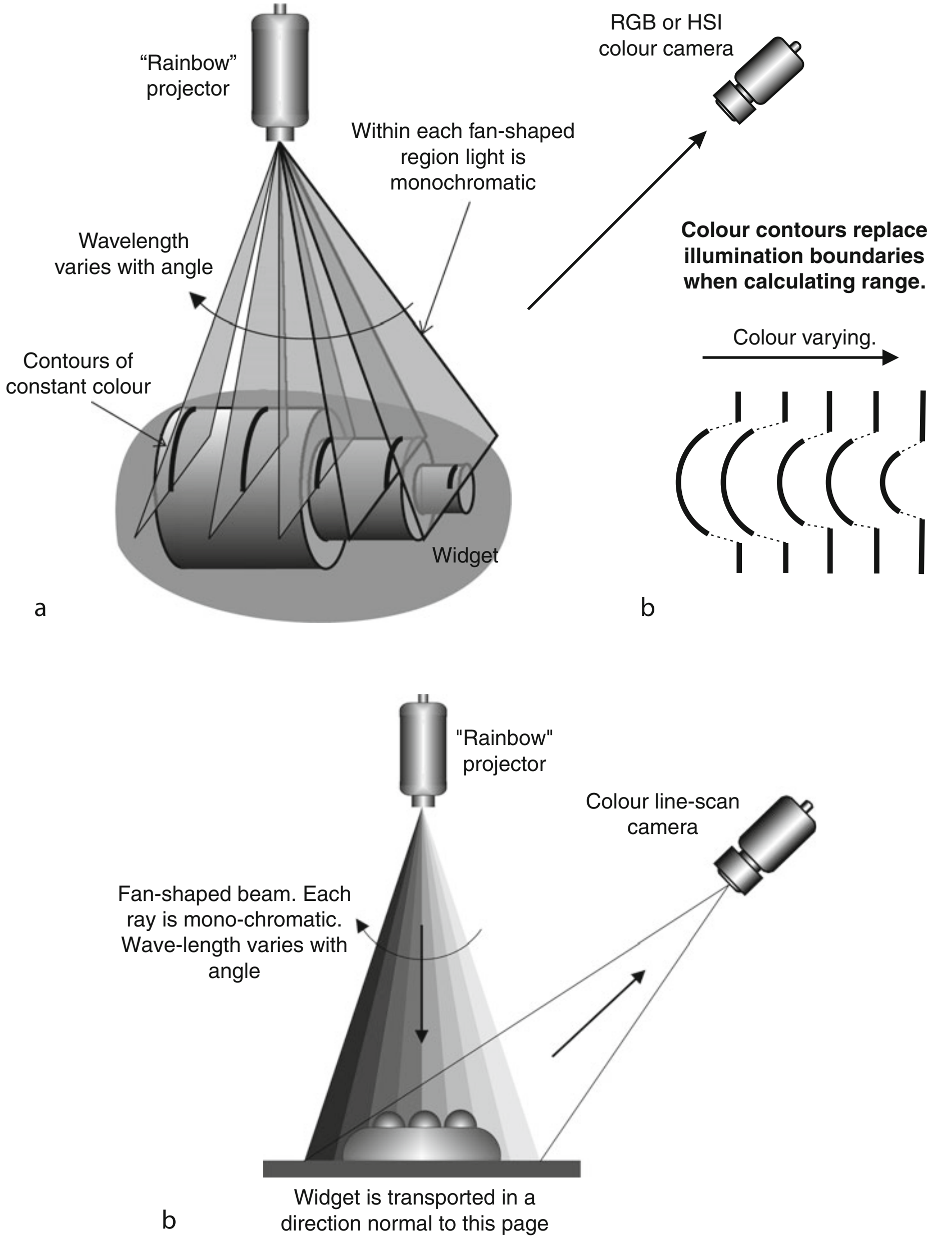

"Rainbow" projector
RGB or HSI colour camera
Within each fan-shaped region light is monochromatic
Wavelength varies with angle
Colour contours replace illumination boundaries when calculating range.
Colour varying.
Contours of constant colour
Widget
a
b
"Rainbow" projector
Colour line-scan camera
Fan-shaped beam. Each ray is mono-chromatic. Wave-length varies with angle
Widget is transported in a direction normal to this page
b

Fig. 40.66
(Continued)

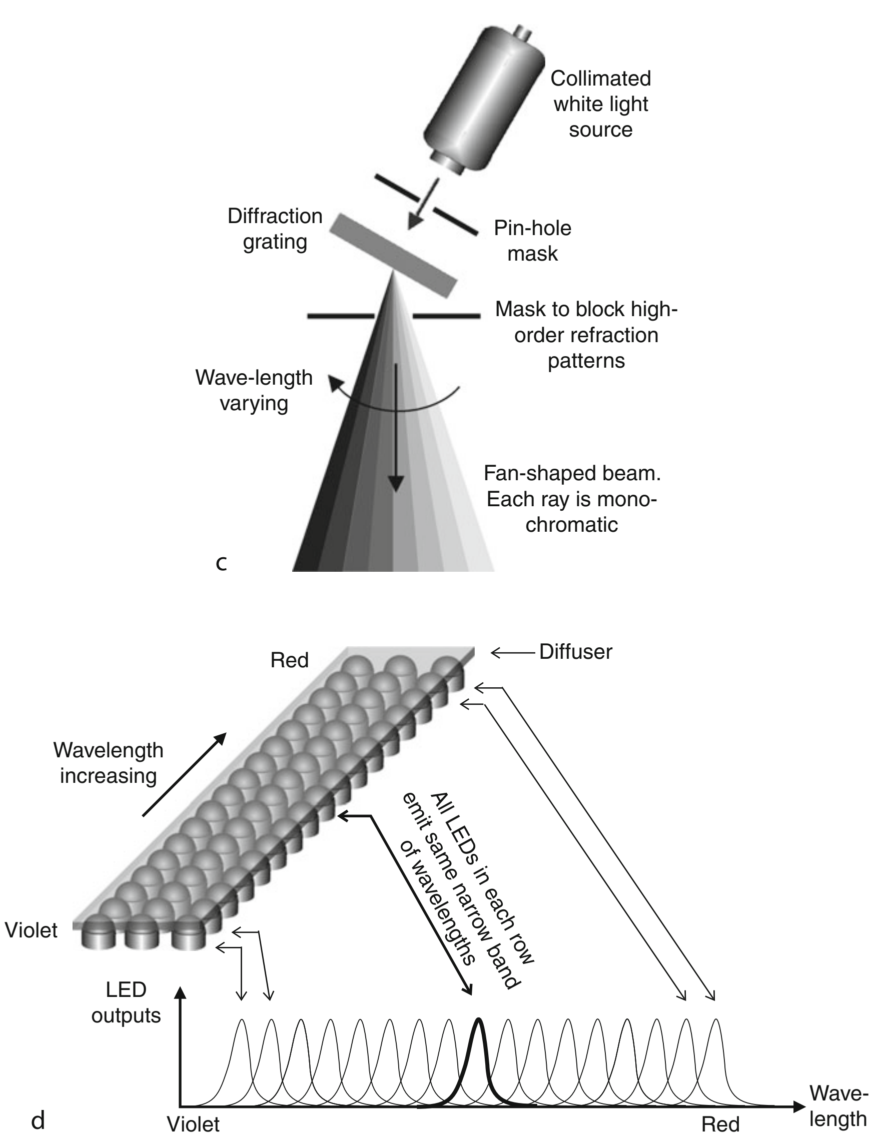

Collimated white light source
Pin-hole mask
Diffraction grating
Mask to block high-order refraction patterns
Wave-length varying
Fan-shaped beam. Each ray is mono-chromatic
c
Red
Diffuser
Wavelength increasing
All LEDs in each row emit same narrow band of wavelengths
Violet
LED outputs
d
Violet
Red
Wave-length

Fig. 40.66
(Continued)

TYPICAL APPLICATIONS
(a) Low-precision measurement and inspection.
(b) Measuring range for inspection and low-precision measurement.

EQUIPMENT Programmable spatial light modulator, such as:

(a) Liquid-crystal light valve.
(b) Digital Micromirror Device (DMD chip, or "mirror chip").

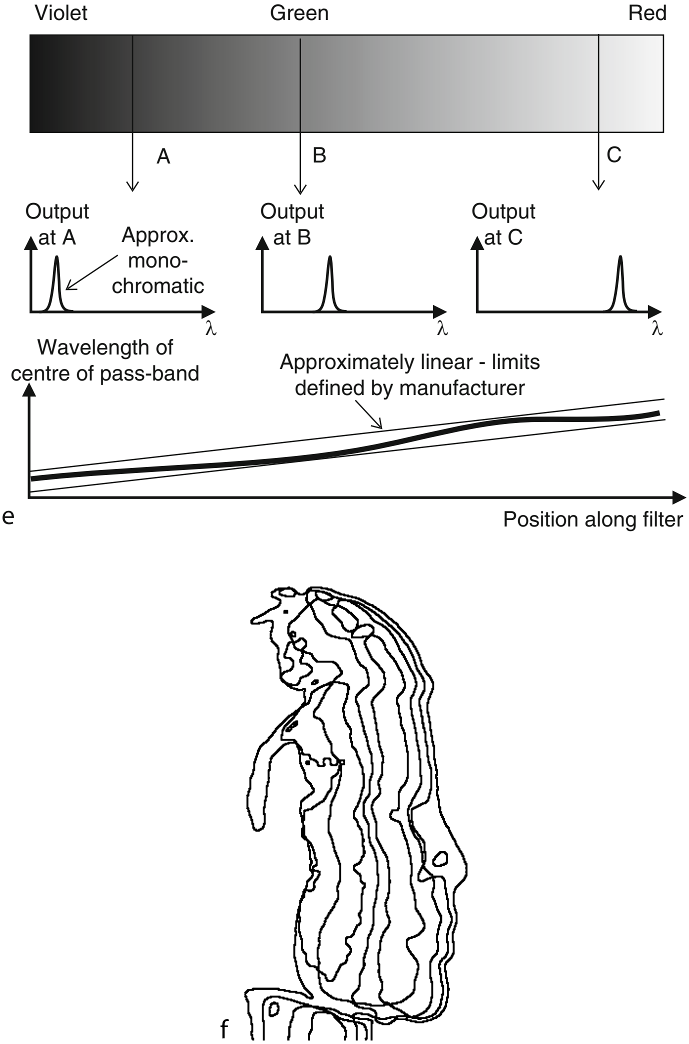


Fig. 40.66

Generating a range map by projecting a colour pattern. (a) Using an array-scan camera. While it was necessary to draw distinct fan-shaped beams and colour contours here, the projector actually produces a continuously variable coloured pyramid of light. Colour contours can provide information about range in the same way as the projected light stripes in Method 66. (b) Using a line-scan camera. (c) Using a diffraction grating to decompose white light. (d) Using an array of LEDs to create a ''rainbow'' pattern. (e) Variable band-pass pass filter. The central wave-length of the pass-band is a linear function of distance along the filter. (f) Sample image: Colour contours derived from a cardboard model of a human head. The ''rainbow'' was generated by a slide projector fitted with a single-stripe mask and a glass prism

CAMERA Array scan.
MOTION Static during several video frame periods (up to about eight).
REMARKS Liquid-crystal display projectors are now common-place in lecture rooms. However, it may be necessary to replace the lens, to accommodate small widgets and to reduce the working distance.

A DMD chip consists of an array of tiny mirrors (normally square and typically 13 × 13 μm^2 to 17 × 17 μm^2), each of which can be tilted through a small angle (10°–12°), thousands of times per second by applying a voltage pulse sequence. A typical DMD chip has a resolution of 1280 × 1024 pixels and the mirror array of mirrors is about 20 × 20 mm^2. An intense, continuous beam of white, coloured or monochromatic light is projected onto the mirror array. When one of the micromirrors is in the *on* state, light shone onto it is reflected towards the target (i.e., the widget) but in the *off* state, the light is reflected onto an absorbing surface, By switching the mirrors with a variable *on:off* ratio, a multi-level image can be projected. Temporal integration of the eye/camera is essential in creating an impression of shading. It is important, therefore, that the mirror's period of vibration is much smaller than the camera's light-integration period, to avoid strobing effects. The DMD chip is used in the manner shown in ➲ *Fig. 40.67b*. Projecting patterns of light onto a widget provides a useful means of inferring its 3D shape. In Methods 68 and 69, a fixed illumination pattern is used and a single image analysed, whereas the two schemes outlined in ➲ *Fig. 40.67* use dynamic, time-varying illumination and generate sequences of images.

To see how this facility can be used to measure range, consider a path *P* through the optical system so that it passes through the *(U, V)* pixel in the spatial light modulator (Notice that if we can identify *P*, we can also locate *(U, V)*). As time progresses, several rays are transmitted, in turn, along path P. The "intensities" of these rays will be denoted by $R_1, R_2, \ldots, R_n$; if ray *i* has zero intensity, then $R_i = 0$ (The spatial light modulator's *(U, V)*-pixel is switched *off*), while rays with non-zero intensity will be denoted thus: $R_i = 1$. The spatial light modulator allows us to generate a different binary sequence $(R_1, R_2, \ldots, R_n)$ for each optical path, thereby providing a unique identity for *P*. The camera observes the *sequence* of brightness values for each point in the image plane. Analysis of this bit pattern reveals the identify of the illumination path and hence *(U, V)*, thus providing the information needed to calculate the widget-sensor distance by applying formula given in Method 107. While this method relies on identifying the optical path digitally, Method 69 does so by varying the wavelength.

REFERENCES

- Efron U (1995) Spatial light modulator technology: materials, devices and applications. Marcel Decker, New York. ISBN: 0824791088
- Digital Micromirror Device (DMD chip). Texas Instruments, http://www.dlp.com/dlp_technology/dlp_technology_products.asp

LAYOUT DIAGRAMS ➲ *Figure 40.67*

40.71 Method 71: Measuring Thickness of a Thin Film

OBJECTIVES To visualise/measure the thickness of a very thin transparent film. The film might be a distinct entity, or the space between two surfaces of near optical precision.
TYPICAL APPLICATIONS Measuring very thin coatings (thickness of a few microns, μm).
EQUIPMENT Point source of white or monochromatic light.
CAMERA Colour array-scan, or line-scan.

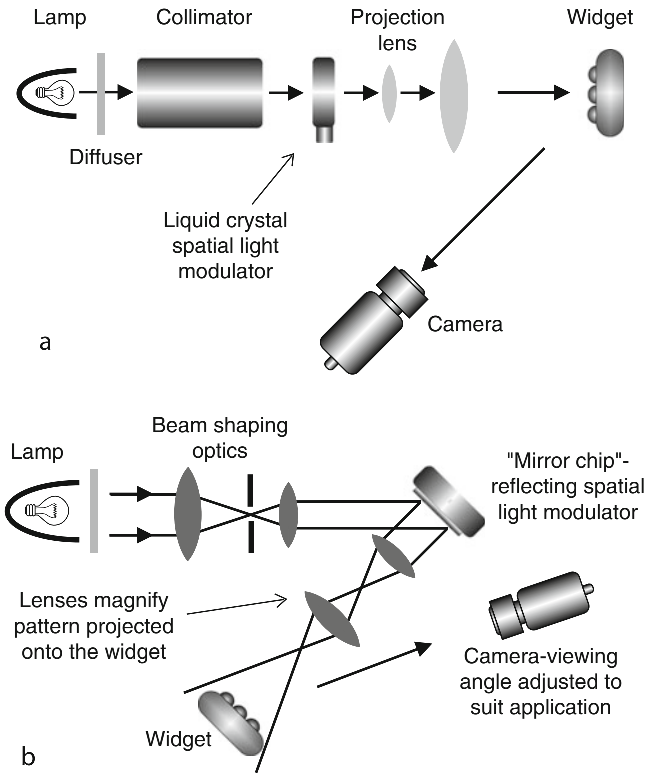

Lamp
Collimator
Projection lens
Widget
Diffuser
Liquid crystal spatial light modulator
Camera
a
Beam shaping optics
Lamp
"Mirror chip"- reflecting spatial light modulator
Lenses magnify pattern projected onto the widget
Camera-viewing angle adjusted to suit application
Widget
b

Fig. 40.67

Program-controlled projection of patterned light. The spatial light modulator is a slightly modified form of computer display (a) Liquid crystal light valve. (b) Digital Micromirror Device

MOTION Static or linear motion.
MATERIALS Very thin transparent film, or coating overlaying a reflective surface.
REMARKS This method creates interference patterns similar to those visible on a film of petrol on a wet road. If monochromatic light is used, the interference fringes vary in brightness, rather than colour. The method can be adapted to visualise the surface height changes on optical quality surfaces. A thin film can be created by placing a glass plate, or lens, on a reference flat surface. Clear oil might be inserted between them. The air film thickness will vary with the undulations of the surface of the test surface. The interference fringes produced in this way represent contours of height difference.
LAYOUT DIAGRAM ➋ *Figure 40.68*

40.72 Method 72: Measuring Refractive Index of a Thin Film

OBJECTIVE Measure film thickness, or refractive index of a thin film.
EQUIPMENT Right-angle prism with a high refractive index.
CAMERA Line scan.

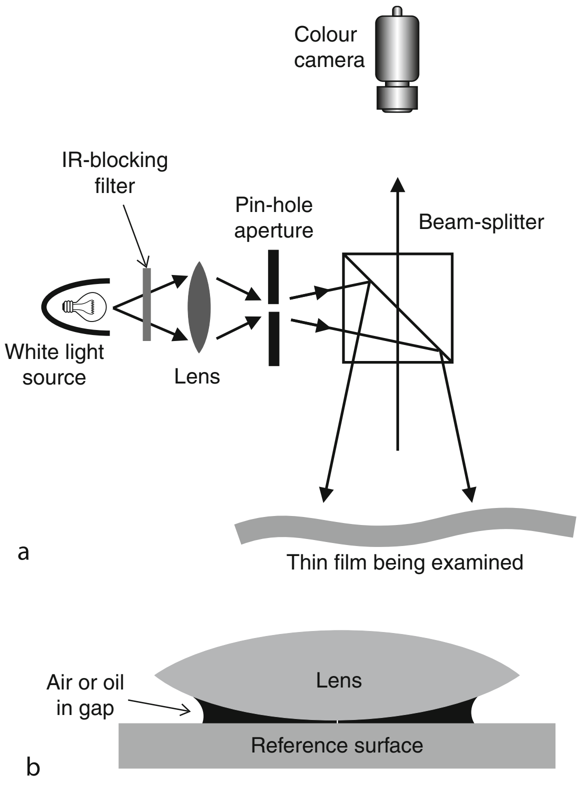


Fig. 40.68
Measuring the thickness of a thin transparent film or coating. (a) Optical arrangement. Produces coloured interference patterns. (b) Newton's rings, produced by a thin "film" of air between a flat surface and a spherical glass lens, can be used to measure the curvature of the test surface

LENS Short focus.
MOTION Static.
GEOMETRY Thin film.
MATERIALS Transparent.
REMARKS The prism is placed in direct contact with the film. Hence, its surface can become dirty and scratched. In some applications, a thin film of water, clear oil some other suitable liquid will help to maintain good optical coupling whilst providing a degree of lubrication. The displacement, *d* of a single ray of light travelling through a film of thickness *t*, with an angle of incidence equal to *A* is given by the formula:

$$t.sin(A).(1 - cos(A))$$

If the film thickness is known beforehand, the refractive index can be measured. Alternatively, if the refractive index is known, the thickness can be measured.
LAYOUT DIAGRAM ➔ *Figure 40.69*

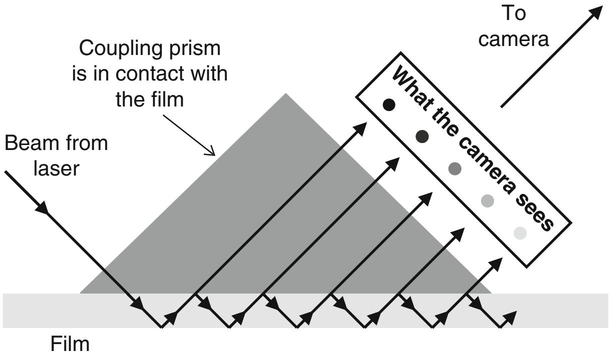


Fig. 40.69
Measuring the thickness of a thin film by observing multiple internal reflections

40.73 Method 73: Micro-Louvres Suppress Ambient Light

OBJECTIVE To prevent ambient light from affecting the image quality, whilst allowing a person to view the inspection area by eye.

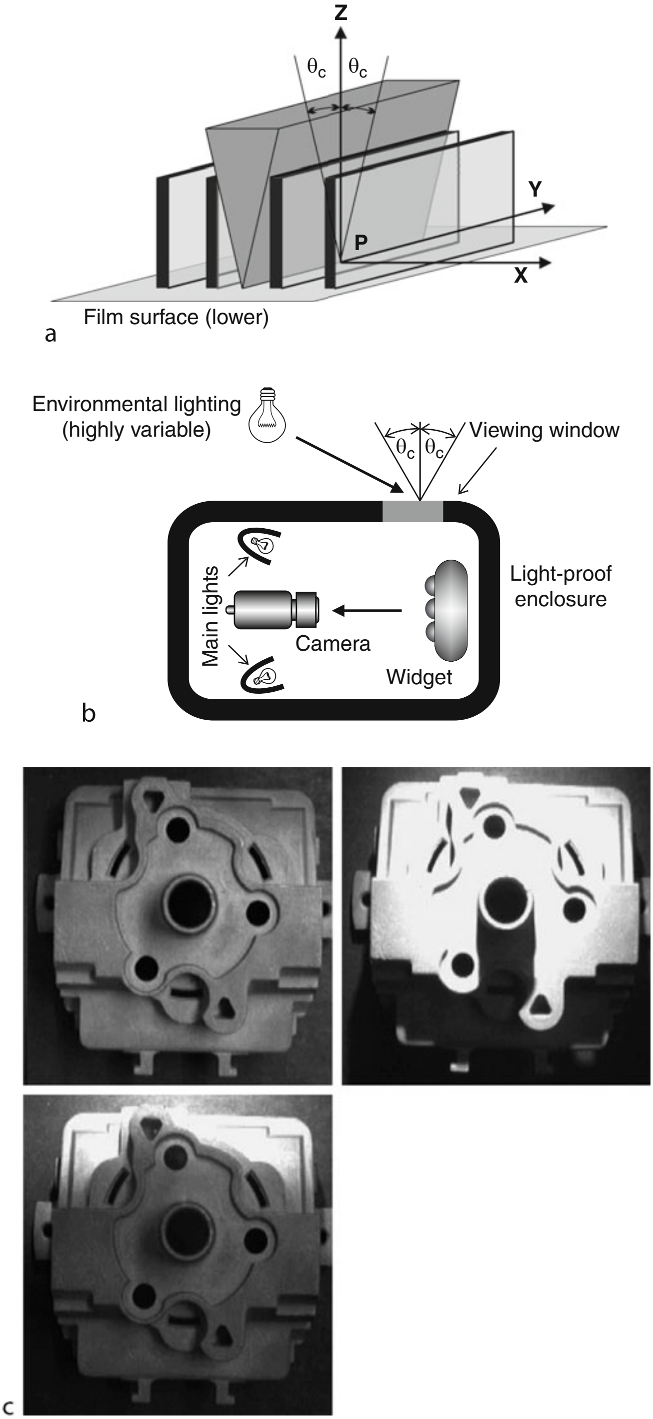
TYPICAL APPLICATIONS Inspecting flexible and other awkward objects that are likely to jam the widget-transport mechanism. Using the arrangement shown in *Fig. 40.70b*, a person can visually monitor widgets moving, without disturbing the inspection process.

EQUIPMENT A light-proof enclosure with a window consisting of *Light-control Film* (product of the 3M Company). The structure and properties of this film are explained in *Fig. 40.70a*. The film is oriented so that it blocks light from the most prominent ambient source but allows a person to view inside the inspection cell.

REMARKS Light Control Film is often used to ensure privacy of viewing for confidential computer displays. People facing the screen directly can see the image unhindered, while an unauthorised person looking obliquely can only see a grey rectangle. Light-control film blocks light over a wide range of wavelengths, since it relies on physical barriers to light transmission: micro-louvres, consisting of parallel strips of opaque material embedded in clear acetate or polycarbonate. The action is similar to that of Venetian blinds. The film transmits all light rays whose angle of incidence, within the XZ-plane, is smaller than some critical angle, θ_c. A ray of light is transmitted if it falls within a wedge (See *Fig. 40.70a*). The light-pass volume becomes a pyramid, if we overlay two films with their (X, Y)-axes interchanged. Light-control film is normally placed in the illumination path, not in the camera's line of sight, since it reduces image quality significantly and can generate Moiré patterns with the camera's photo-detector matrix. Light-control film can be used to construct a collimator. Use a single sheet to collimate one axis only. Use two sheets, aligned orthogonally to one another, to collimate the beam in both axes.

REFERENCES Vikuiti™ Light Control Film, 3M Co. http://solutions.3m.com/wps/portal/3M/en_US/VikuitiHome/. Accessed 5 Sep 2011

LAYOUT DIAGRAM AND SAMPLE IMAGES *Figure 40.70*



◘ Fig. 40.70
(Continued)

40.74 Method 74: Microscopic Fourier Analysis of Transparent Film

OBJECTIVE To analyse a thin transparent film for microscopic surface defects, such as scratches.

TYPICAL APPLICATIONS Inspecting optical–quality transparent film, sheeting, etc. (e.g., acetate film used with overhead projectors, microfiche film, etc.).

EQUIPMENT Laser, beam expander. Converging lens.

LENS Macro lens, capable of close-up viewing of a small area.

CAMERA Array scan with a wide dynamic range.

MOTION Static.

MATERIALS Thin sheet of transparent material.

REMARKS Projecting a coherent beam of light though a transparent film generates the Fourier Transform of any surface defects, such as scratches. The Fourier transform is visible as the far-field interference pattern. To shorten the optical path, a simple lens can be used (See *Fig. 40.71*). This effect can be demonstrated simply, by shining the beam from a laser pointer through a sheet of OHP film. A similar result can be obtained using a fine fabric, a 50 mm slide, or the edge of a screw with a fine-pitch thread. Viewing distant street lights through a gauze curtain produces the same effect. Since the Fourier transform is generated optically in real-time, image processing software need only be applied to undemanding operations: sampling the intensity at a few pre-defined points, or simple fixed-position image comparison. Care must be taken to ensure that the prospective customer for a vision system understands that the Fourier Transform, not a direct image of the surface features, is generated by this method. He/she should appreciate, for example, that a series of equally spaced horizontal scratches generates a set of bright spots lying along a *vertical* line. The Fourier transform often generates an image with a very wide dynamic range, in excess of that accommodated by a standard video camera. For this reason, a camera with a logarithmic response characteristic is highly desirable.

REFERENCES

- Butters JN (1985) Coherent optical methods. In Batchelor BG, Hill DA, Hodgson DC (eds) Pattern recognition: ideas in practice. Plenum, New York/London. ISBN 0-903608-68-5, Ch. 8

Fig. 40.70

Light-control film. (a) A set of thin, opaque, parallel slats, placed very close together forms a crude, single-axis collimator. Any ray of light impinging on point P and originating within the grey wedge-shaped region (extended indefinitely along the Y- and Z-axes) will pass through the film, which covers the XY-plane. Other rays will be blocked. (b) Blocking ambient light from a single, fixed direction, whilst allowing viewing of the widget in situ. (c) Sample images: Zinc die-casting, black back-ground. Multi-source front lighting. *Top-left:* No additional lighting. *Top-right:* Brilliant ambient lighting, at a large angle of incidence, greatly reduces the image quality. *Bottom-left:* Light-control film blocks the ambient light, improving the image quality. The film does not provide an effective block at the *top* of the image, where the angle of incidence is less than the critical angle, θ_c (This can be avoided by altering the inspection cell geometry). If the film is rotated by 90°, it cannot block the unwanted rays

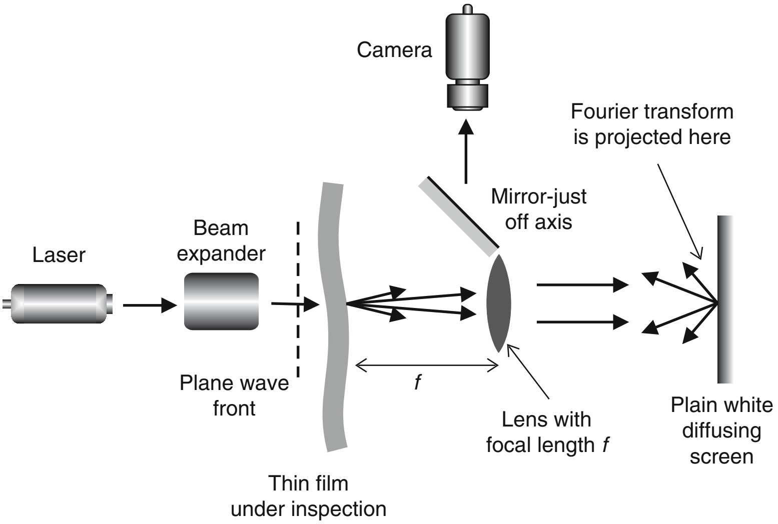

Camera
Fourier transform is projected here
Mirror-just off axis
Beam expander
Laser
Plane wave front
f
Lens with focal length f
Plain white diffusing screen
Thin film under inspection

Fig. 40.71
Fourier analysis of a transparent medium. The horizontal axis has been compressed for ease of illustration. Placing the camera slightly off-axis does not introduce significant errors, which can, in any case, be corrected by software. This optical arrangement prevents the beam from the laser ever shining directly into the camera and damaging it

- Ullman JR (1978) Optical pattern recognition techniques. In: Batchelor BG (ed) Pattern recognition: ideas in practice. Plenum, New York/London. ISBN 0-306-31020-1

LAYOUT DIAGRAM *Figure 40.71*

40.75 Method 75: Motorised Filter Wheel

OBJECTIVE To provide a means of switching optical filters and polarisers under program control.

TYPICAL APPLICATIONS

- Flexible inspection cell.
- System prototyping.
- Working in an unpredictable optical environment.

EQUIPMENT Metal disc, fitted with various filters, such as:

- UV blocking filters.
- IR blocking filters.
- Colour-selection filters
- Neutral density filters.
- Polarisers.

REMARKS One commercial device allows the following filters to be fitted:

- 8 filters, size 1″, or

- 6 filters, size 1.25″, or
- 4 filters, size 1.5″

The unit is 6″ diameter, by 0.65″ deep. Another filter wheel, holds one of the following sets of filters:

- 16 filters, size 1″
- 12 filters, size 1.25″
- 8 filters, size 2″

This device is 10″ diameter by 1.3″ deep. It can hold square or round filters. Control is exercised by a USB or RS232 port. Since two units can be mounted back-to-back to combine filters. For example, it is possible to mount a series of low-pass filters on one wheel and high-pass filters on the other. This combination implements up to 64 band-pass filters that can be selected by computer. Also see Method 40.

REFERENCE

- Batchelor BG, Steel AK (1985) A flexible inspection cell. In: Proceedings of the international conference on automation in manufacturing, part 4: automated vision systems, Singapore. Singapore Exhibition Services, pp 108–134. Also in Proceedings of the 5th international conference on robot vision and sensory control, Amsterdam, 1985. IFS, Bedford, pp 449–468
- Motorized Filter Wheels, Edmund Optics. http://www.edmundoptics.com/. Accessed 5 Sep 2011

40.76 Method 76: Multi-arm Image Conduit

OBJECTIVE To provide a split-image facility, so that several views of a complex object, or scene, can be combined into a single video frame. All sub-images are sampled at the same instant and may be rotated (by twisting the image conduit). In addition, the magnification may be different for each sub-image, which may be derived from an awkward viewing location, or a hostile environment.

TYPICAL APPLICATIONS Viewing complex objects and scenes in which there are several "interesting" areas, with large "uninteresting" areas that need not be examined.

EQUIPMENT Several lengths of fibre-optic image conduit. Flexible bundles may be used instead but this option is more expensive.

CAMERA Array.

LENS The main camera lens is replaced with the multi-arm conduit/bundle. Each conduit rod has its own objective lens.

MOTION Static.

REMARKS Since miniature cameras are cheap nowadays, this method is less important than it once was. The chief advantage is that the same camera is used to view all sub-images, thereby achieving accurate colour and intensity matching from one sub-image to another. If a frame-transfer camera is used, the sub-images are sampled at the same moment, thereby allowing accurate synchronisation of activity within the different sub-images.

LAYOUT DIAGRAM ➋ *Figure 40.72*

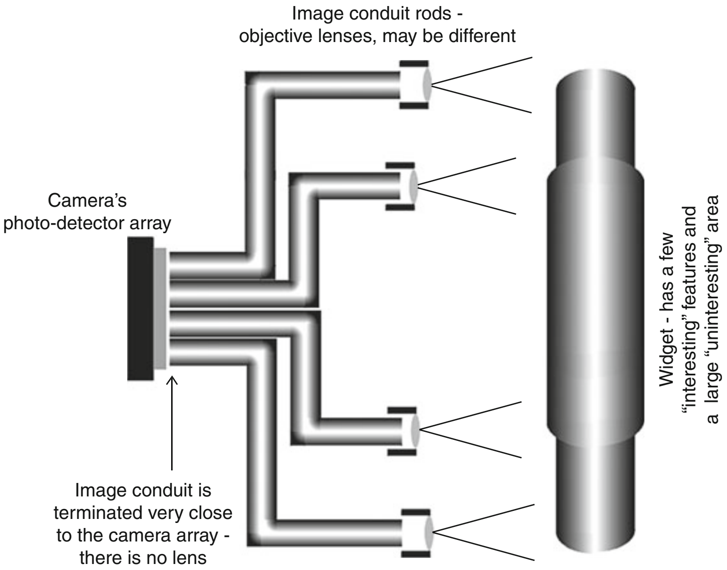


Fig. 40.72
Multi-arm fibre-optic image conduit, used to split the camera's field of view. The widget must be presented for inspection in fixed position and orientation

40.77 Method 77: Multi-Camera Specular and Near Specular Illumination

OBJECTIVES To highlight shallow surface features by providing oblique illumination, with oblique and normal viewing.

TYPICAL APPLICATIONS Inspecting nominally flat shiny surfaces, such as metal sheet, plastic film, sheet glass, etc. The surface to be inspected may be polished, painted or lacquered.

EQUIPMENT Up to four cameras, plus one of the following:

- Back light unit with slit mask placed in front
- Fluorescent tube, with a high frequency power source
- Fibre optic line illuminator
- Array of LEDs fitted with diffuser

CAMERA Line scan.

MOTION Linear motion.

GEOMETRY Nominally flat, smooth with minute pits, scratches, dimples, pimples, small particles of foreign material adhering to the surface.

MATERIALS Metal sheet, plastic film, glass, glossy paper/card, etc.

REMARKS Up to four cameras can be used simultaneously. These generate four images: "below specular" (BS in *Fig. 40.73b*), "specular" (S), "above specular" (AS) and "normal" (N). They are registered perfectly along the Y axis (i.e., along the web) and alignment along the

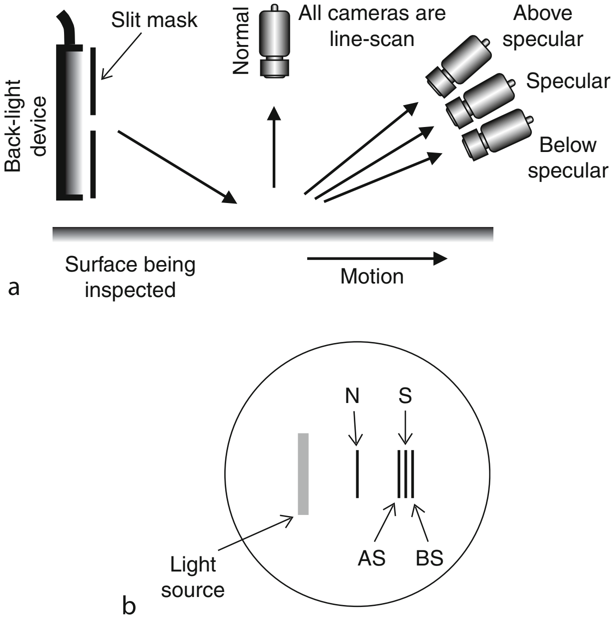


Fig. 40.73
Multi-camera grazing illumination. (a) Optical Layout. (b) Illumination diagrams. A - normal viewing; AS - above specular; S - specular; BS - below specular

Y axis (i.e., across the web) is straightforward. Hence, it is possible to obtain four well-registered images, which can be analysed individually or combined by suitable software. Highly sensitive discrimination of surface irregularities can be obtained by using a collimated light beam, in which all rays are parallel and lie within a narrow sheet. This can be achieved by projecting a collimated light beam through a slit aperture (The slit must be wide enough to avoid producing significant diffraction effects).
DIAGRAMS AND SAMPLE IMAGE ➋ *Figure 40.73*

40.78 Method 78: Multi-camera Structured Lighting

OBJECTIVE To obtain 3D shape information and generate an "all round" surface relief map, by simultaneously scanning the top and two sides of an object.
TYPICAL APPLICATION

- Analysing the shape of baked goods: loaves, pies, cakes, etc.
- "All round" inspection of extrusions, tubing, cables, etc.

EQUIPMENT Three or more lasers, fitted with holographic or cylindrical lenses, to generate fan-shaped beams.
CAMERA Three or more array cameras. To obtain precise shape measurements, digital cameras should be used (Vertical and horizontal video clocks should be locked to the scanning of the photo-detector array).
LENS Three or more lenses, probably with similar focal lengths.

MOTION Continuous linear motion.
GEOMETRY Should not be very complicated, to avoid occlusion and sub-sequent fragmentation of the range map.
MATERIALS Matt not reflective material, preferably not very dark.
REMARKS Aligning the lasers is straightforward: simply, rotate them by hand, until their beams coincide, projecting a single bright line onto any plane. The cameras view the widget from different vantage points. Hence, their images must be "warped," by applying simple geometric transformations, before they can be combined to create a composite cross-section contour of the widget. In order to align and register these images, careful calibration is necessary. This relies on a target consisting of a square board, with four high-contrast markers (e.g., LEDs), arranged at its corners (➋ *Fig. 40.74b*). The markers provide four reference points, from which we can calculate the appropriate geometric transformations. A different warping function is required for each vantage point. In each case, the objective is to map a trapezium into a square. The following image "warping" functions are needed:

(a) Linear rescaling and shifting of the horizontal axis (Cameras 2 and 3)
(b) Linear rescaling and shifting of the vertical axis (Camera 1)
(c) Trapezium-to-square mapping (Cameras 1–3)
(d) Non-linear stretching along one axis (Vertical axis for camera 1; Horizontal axis for cameras 2–3)

The last of these is only required if the cameras are placed close to the object. Once these functions have been performed, three "warped" images are obtained, each one containing a bright arc. These transformed images are then superimposed, with the markers aligned. This creates an image containing a single continuous bright curve, tracing the top and side edge contours of a vertical cross-section though the widget.
LAYOUT DIAGRAM ➋ *Figure 40.74*

40.79 Method 79: Multiple Views of a Widget

OBJECTIVE To combine several views of a widget, obtained under different lighting conditions thereby allowing features, or combinations of features, to be detected in a way that it is not possible from a single view.
EQUIPMENT Computer-controlled lighting system, with two or more sources, which may provide light of different colours and polarisation.
CAMERA Array.
MOTION NONE The widget must be absolutely stationary compared to the camera. Image acquisition requires two, or more, frame intervals.
REMARKS This method is suitable for viewing complex components and assemblies that have a non-homogenous surface finish (e.g., black rubber and polished metal). It is possible to combine edge contours and internal features (e.g., cracks, splits, pitting, etc.), which may be critical in certain parts of an object but not in others. The precise layout of the optical sub-system may vary from one application to another; the simple optical arrangement in ➋ *Fig. 40.75* is just one example that illustrates the general principle. Individual light sources can vary in position, polarisation and colour. Multi-lamp/multi-colour lighting systems are potentially very useful. Since the whole visible-light spectrum (VIS) can be covered by a series of LEDs whose wavelengths vary in 0.01 µm steps, it is possible to tailor a multi-colour lighting

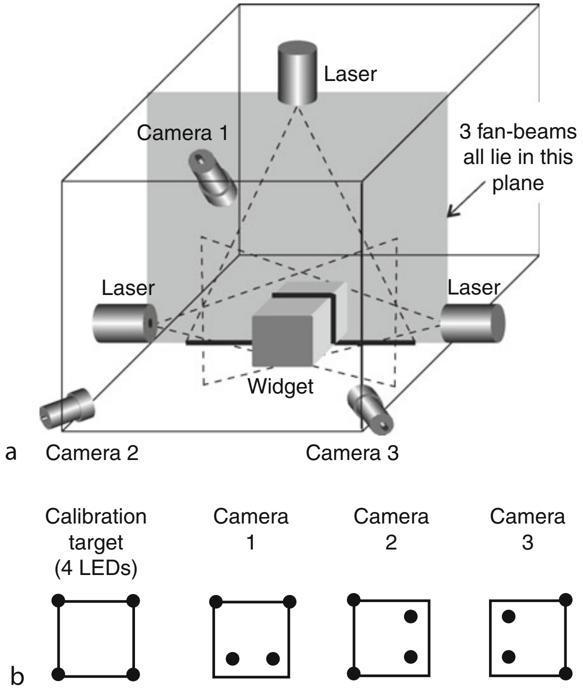


Fig. 40.74
Multi-camera Structured Lighting. (a) Optical layout. (b) Calibration target and images obtained by the three cameras

system to suit each individual application. A crude spectral analysis of a complex 2- or 3-dimensional scene can be performed using a series of switched LEDs. Holograms produce diffraction patterns which change their colour as the position of a white light source changes. This effect can be useful for authenticating holographic security tags. The general principle of combining two or more images, derived under different lighting conditions, has not been exploited to its full potential. There is considerable scope for further research in both the design of the lighting system and the processing of multi-view images.

LAYOUT DIAGRAM AND SAMPLE IMAGES *Figure 40.75*

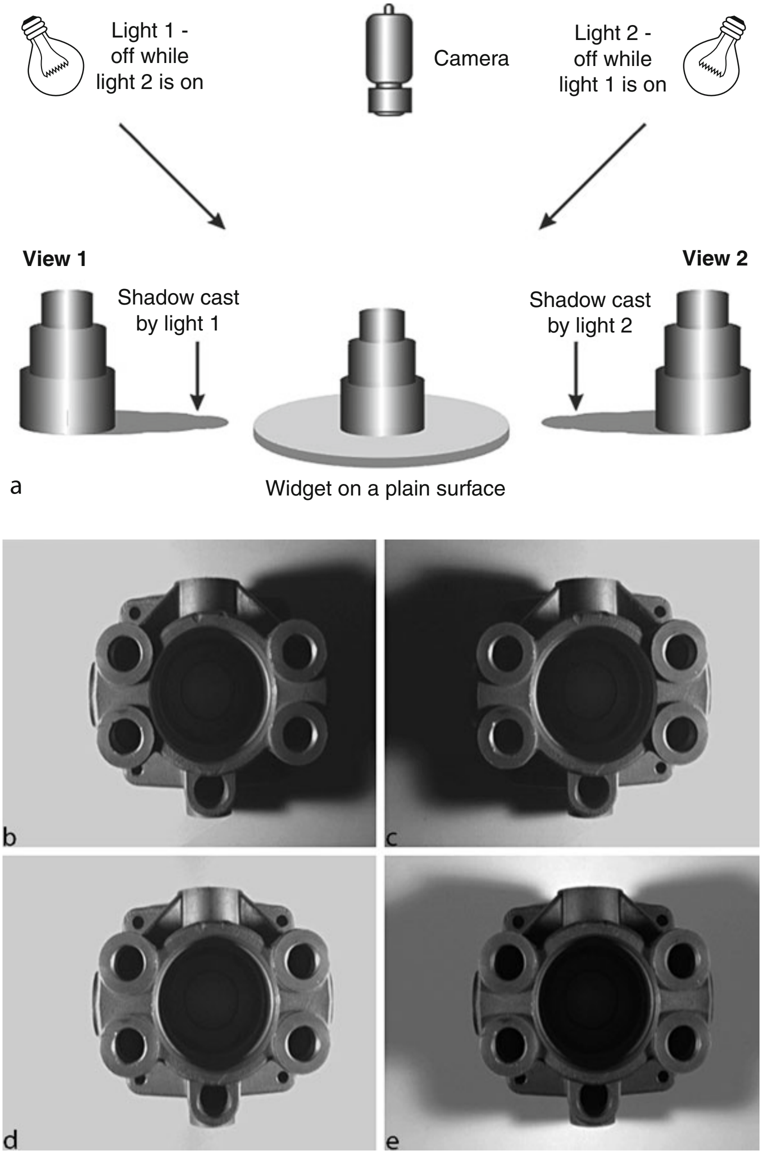
40.80 Method 80: Multi-spectral Image Sensing

OBJECTIVE To obtain images from a number of different spectral bands in the near-UV, VIS and near-IR. Performing a coarse spectral analysis at high speed.

TYPICAL APPLICATIONS Inspecting coloured packaging, printing and other multi-coloured objects.

EQUIPMENT Several matched image sensor arrays and filters, beam-splitters and mirrors or prisms.

REMARKS Three different optical arrangements are shown in *Fig. 40.76*. In each case, the optical system must be carefully aligned, to ensure that the images are properly registered.



■ **Fig. 40.75**
Images derived with lighting from different directions (a) Optical layout. (b) Sample image: hydraulics manifold, lit from the left. (c) The same component lit from the right. (d) Images (b) and (c) combined by computing the maximum intensity between corresponding pairs of pixels. (e) Lighting simultaneously from both sides casts shadows and produces an image that is far more difficult to analyse

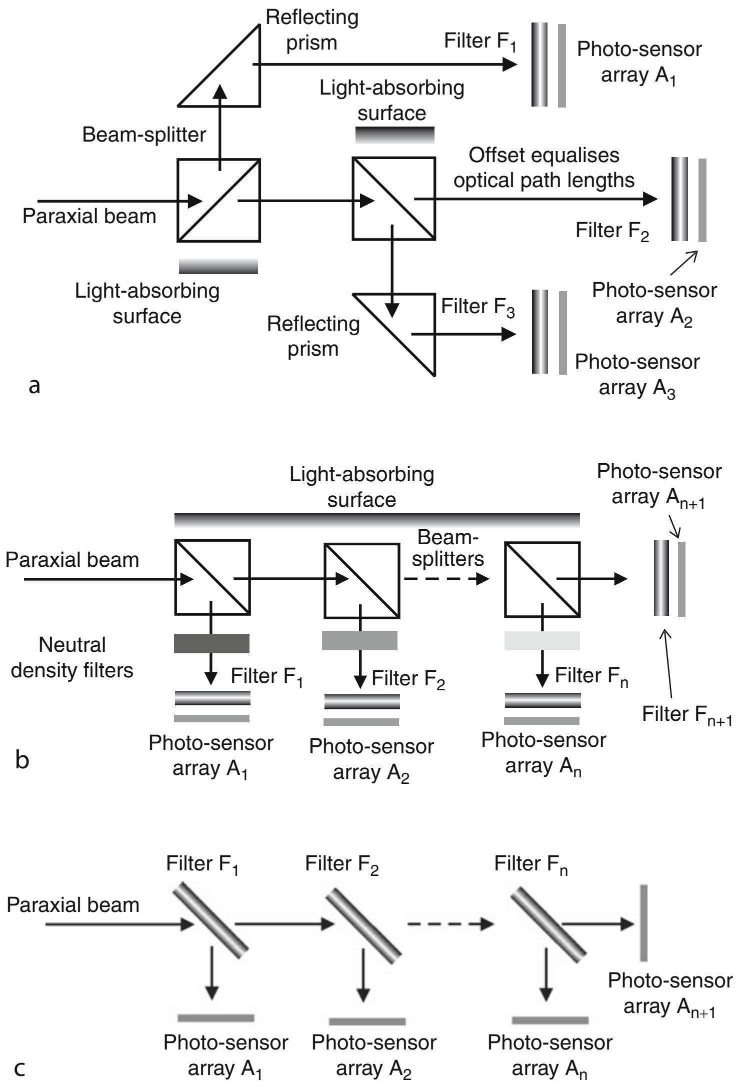


Fig. 40.76

Multi-spectral image sensing. These three diagrams are not drawn to scale; the path lengths should be equal, to ensure that the images will all be in focus and of the same size. (a) Using 50%/50% beam-splitters and filters that absorb, rather than reflect, ''unwanted'' light. The optical arrangement shown here can be extended to provide more than three colour-separation channels. Filter F_2 and sensor array A_2 are deliberately offset to equalise the optical magnification. (b) Alternative arrangement. Since each beam splitter has an (theoretical) insertion loss of 50%, the intensity decreases exponentially as the beam travels from left to right. This limits the overall length of the beam-splitter chain and can be corrected, for short chains, by the neutral density filters. (c) Using dichroic filter-mirrors. The order of the filters along the chain should be chosen to maximise the beam intensity. This does not suffer from the exponential loss of beam power that the previous arrangement does

These arrangements are theoretically capable of operating over a wide spectrum: near-UV, VIS and near-IR wave-band. However, bear in mind that solid-state sensors are most sensitive in the red/near-IR part of the spectrum and that they are much less so in the UV waveband. Also, remember that narrow band-pass filters do not transmit much light. As a result, high illumination intensity is often required. Using the optical arrangement in ➋ *Fig. 40.76b* or *c*, it is possible to examine a narrow spectral band (i.e., colour analysis) of a scene. For example, it is feasible to analyse up to about seven bands within a wave-length range of say 0.1 μm (Conventional colour cameras analyse just three channels over a much broader spectral range.). This is useful for precise measurement of colour variations, in natural products and food-stuffs, where subtle colour changes occur.
LAYOUT DIAGRAMS ➋ *Figure 40.76*

40.81 Method 81: Multi-View Stereo Imaging

OBJECTIVE To provide several views of an object, so that range data can be obtained.
TYPICAL APPLICATIONS Robot vision, automated assembly, automated vehicle guidance.
EQUIPMENT Two camera mounted on a rigid bar. The cameras' optical axes are arranged to converge at the desired working distance.
CAMERA Array.
MOTION Static.
GEOMETRY Preferably containing a number of step-like edges or holes.
REMARKS The analysis of range data from a pair of stereo images is a complex subject and the reader is referred to ➋ Chap. 15 for a more comprehensive discussion. Ideally, the scene being viewed should contain a large number of easily identifiable features that are visible to both cameras. These might be sharp "step" edges, holes, spots or linear features. They may be geometric (i.e., part of the 3D shape of the widget), or surface features (e.g., printed, etched, or punched), If few such features exist, it may be advantageous to project a pattern, such as a grid, an array of parallel lines, or spots onto the widget surface. Even a single point projected onto the widget by a diode laser (➋ *Fig. 40.77c*), can be useful in calibrating and aligning the optical system. Multiple lines or spots can be projected using a laser fitted with a diffraction lens. A projected "random" pattern might also be used. It is important, however, that we choose a projected pattern that allows easy alignment of the images from the various cameras; a truly random pattern might be difficult to analyse one that is more structured. The exact position of the projector is unimportant, as it is the disparity between the images derived from the various cameras that allows us to calculate the range. When we are inspecting a widget, we are normally trying to confirm that it conforms to the specification; For the purposes of *verification*, as distinct from *recognition*, we may use any convenient and easily analysed projected pattern.
REFERENCES Marshall AD, Martin RR (1992) Computer vision, models and inspection. World Scientific, Singapore. ISBN 9810207727
LAYOUT DIAGRAMS ➋ *Figure 40.77*

40.82 Method 82: Normal Viewing of Spherical Surface

OBJECTIVE To view part of a spherical surface normally.
TYPICAL APPLICATIONS Inspecting surfaces of ball bearings.

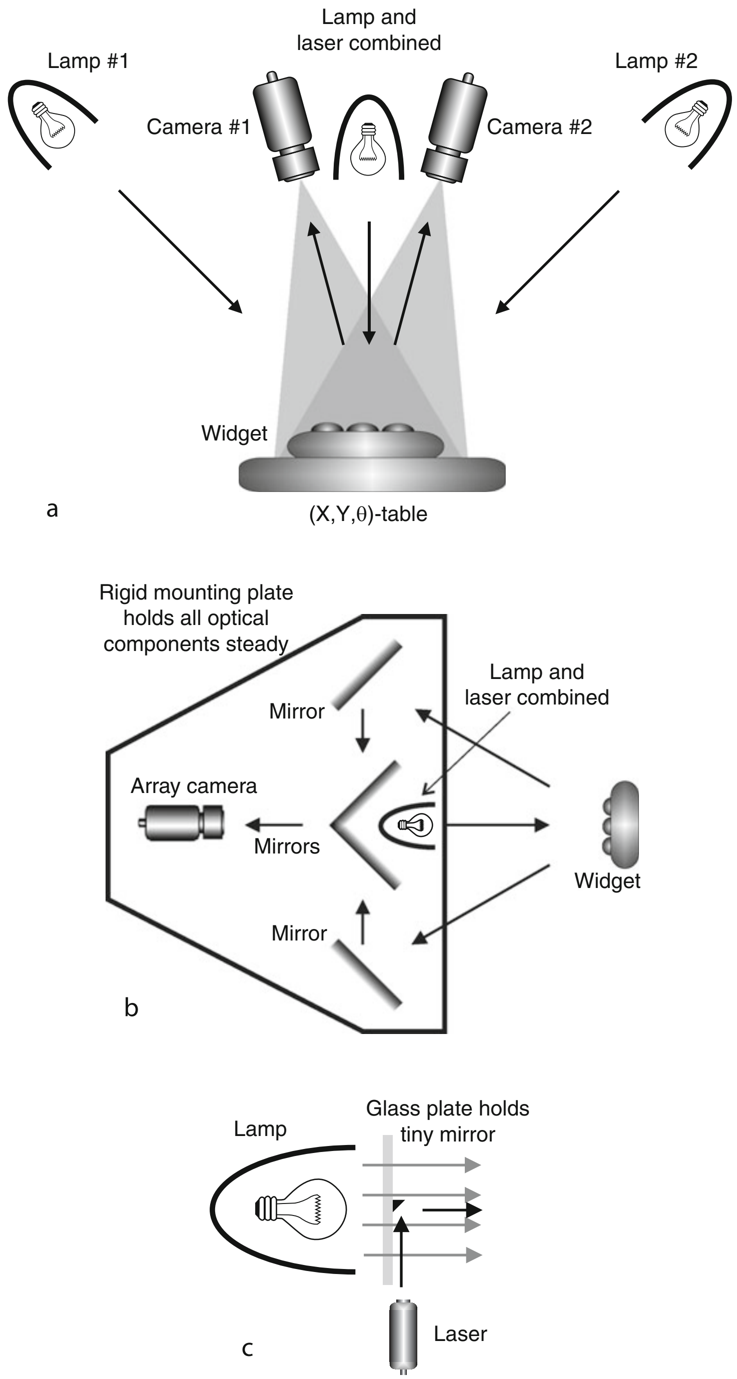


Fig. 40.77
Stereo imaging. (a) Using two cameras. (b) Using a single camera and image splitter. (c) Co-axial illumination by a lamp and laser

OPTICS
(1) Standard lens
(2) Two parabolic mirrors
(3) Elliptical mirror

CAMERA Array.

MOTION Static.

GEOMETRY Spherical.

MATERIALS Brightly polished (glinting).

REMARKS In the arrangements shown in ➲ *Fig. 40.78a* and *b*, the illumination and viewing are both normal to the widget surface. Hence, specular reflection is guaranteed over the whole field of view. Surface imperfections are therefore seen easily, if they alter either the angle or intensity of the reflected beam. Hence, imperfections of the spherical geometry and surface staining are both visible. The arrangement in ➲ *Fig. 40.78c* provides normal viewing but does not enable normal illumination to be achieved. Similar arrangements to these can be devised using cylindrical lenses and mirrors. A line-scan camera could then be used and the object rotated to scan all round the widget. None of these methods allows the whole surface to be viewed at the same time. They work best if the widget is small compared to the focal length of the lens/mirrors.

LAYOUT DIAGRAMS ➲ *Figure 40.78*

40.83 Method 83: Objects on a Conveyor

OBJECTIVE To inspect widgets transported past the camera on a continuously moving conveyor.

TYPICAL APPLICATIONS Inspecting discrete objects: piece parts and assemblies, one at a time.

EQUIPMENT Conveyor belt. Strobed lighting, or a camera fitted with a fast electronic shutter. Proximity sensor (Used to detect when a widget is in front of the camera). *Accept/reject* device (Used to deflect faulty widgets into a bin). A concurrent processor is ideally suited to this type of inspection application.

CAMERA Progressive scan, array type (Images can be digitised at any moment).

MOTION Continuous or indexed linear motion.

REMARKS The belt speed need not be known precisely. This method is not particularly sensitive to vibration. The camera can be mounted either above or beside the conveyor. Three cameras can used simultaneously, to inspect the top and sides of the widget. Strobed lighting can induce epileptic fits and migraine attacks, even in people who are not knowingly vulnerable. Moreover, moving objects can appear stationary under strobed lighting. Hence, care must be taken to stop bright flashing light, generated by the inspection system, from being visible to people standing nearby. A camera fitted with a fast electronic shutter is intrinsically safer but it requires continuous, high intensity lighting, often creating excessive heat. A LED-photo-transistor proximity sensor can be used to signal the arrival of widgets, as they reach the camera. However, it is important that the light source associated with the proximity sensor (often an IR LED) does not spoil the image (➲ *Fig. 40.78b*). A concurrent processor is ideal for inspecting objects on a conveyor, since it can simultaneously analyse images derived from several widgets, as they travel from the camera towards the *accept/reject* device. Such a system can have a high *throughput rate* (i.e., number of widgets inspected per unit time), even though

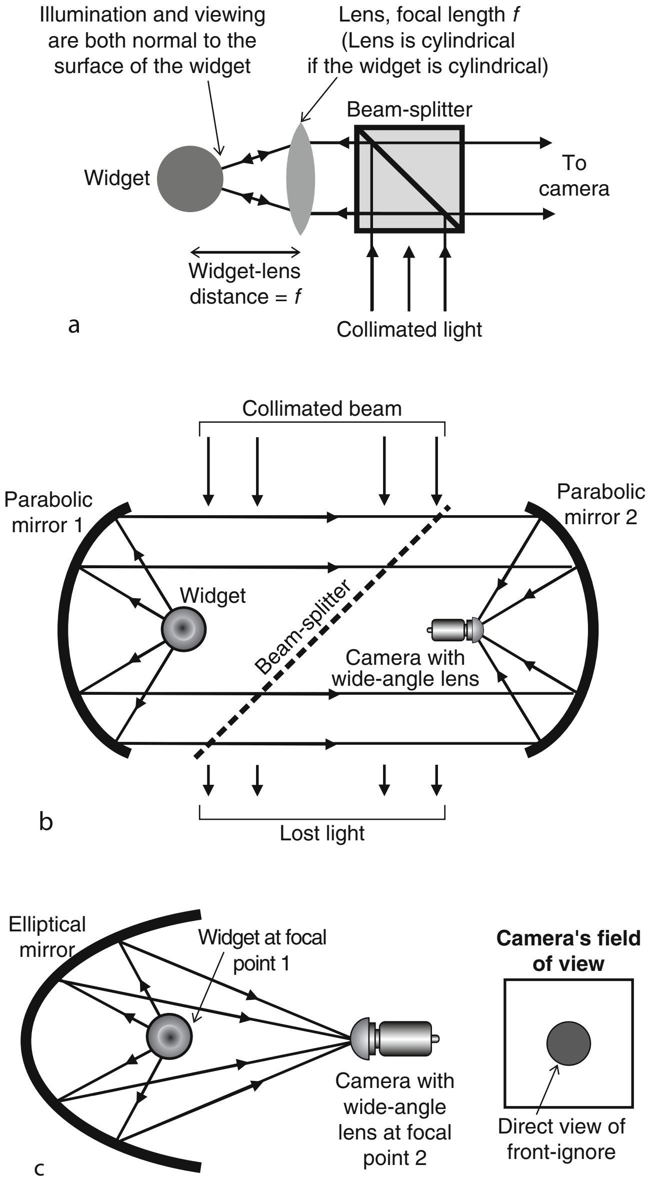

Illumination and viewing are both normal to the surface of the widget
Lens, focal length f (Lens is cylindrical if the widget is cylindrical)
Beam-splitter
Widget
To camera
Widget-lens distance = f
Collimated light
a
Collimated beam
Parabolic mirror 1
Parabolic mirror 2
Widget
Beam-splitter
Camera with wide-angle lens
b
Lost light
Elliptical mirror
Widget at focal point 1
Camera's field of view
Camera with wide-angle lens at focal point 2
Direct view of front-ignore
c

Fig. 40.78
Viewing a spherical surface normally. (a) Optical layout using a single lens. In practice, a compound lens would probably be used, to obtain a greater coverage of the surface, by providing a wider viewing angle. (b) Optical layout using two parabolic mirrors. The widget and camera are placed at the focal points of the mirrors. The symmetrical reflectors do not introduce any geometric distortion. Some rays reflected from mirror 1 are blocked by the widget on their way towards mirror 2. Hence, there is a small ''dead'' region in the centre of the image, (c) Optical layout using an elliptical mirror. This arrangement does not preserve the geometry of the widget. Obtaining uniform normal lighting is problematical

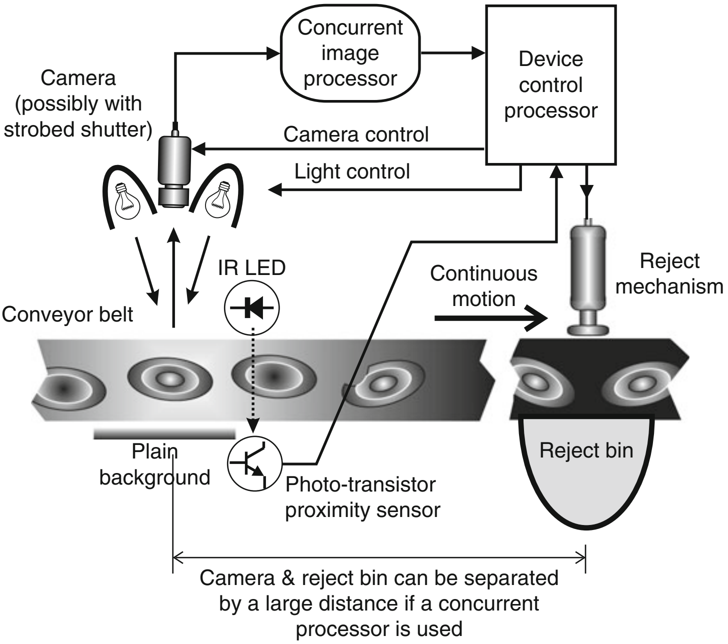

Concurrent image processor
Device control processor
Camera (possibly with strobed shutter)
Camera control
Light control
IR LED
Continuous motion
Reject mechanism
Conveyor belt
Plain background
Photo-transistor proximity sensor
Reject bin
Camera & reject bin can be separated by a large distance if a concurrent processor is used

Fig. 40.79
Inspecting objects on a continuously moving conveyor. If strobed lighting is used, the region immediately around the camera and lamps must be enclosed within a light-proof screen (not shown). IR light from the proximity sensor can cause reflections. Solutions: Make main lights brighter; use an optical filter to block IR entering the camera; switch off the IR LED during image capture

its *latency* is high (Latency is the time taken for a widget to travel along the conveyor, from the camera to the *accept/reject* mechanism). During the latency period, the conveyor belt effectively “stores” widgets until the *accept/reject* decision has been computed. This method can be slightly modified, to examine web materials that have a strongly defined pattern (e.g., high-contrast printed materials). Image capture is synchronised by a simple non-imaging sensor observing registration features (fiduciary marks) on the edge of the web.

REFERENCES Batchelor BG, Waltz FM (2001) Intelligent machine vision: techniques, implementation and applications. Springer, Berlin. ISBN 3-540-76224-8

LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.79*

40.84 Method 84: Observing Bubbles in Clear Liquid

OBJECTIVES To view small air bubbles rising in a clear liquid, such as water, oil, etc.

TYPICAL APPLICATIONS

- Checking for air leaks in pipes, bellows. joints, etc.
- Visual determination of f in carbonated liquids (lemonade, mineral water, beer, champagne), etc.

- Measuring liquid viscosity, by deliberately producing bubbles that rise naturally due to gravity and then measuring their sizes and terminal velocities.

EQUIPMENT Water/oil bath with transparent walls or optical windows.

CAMERA Array or line scan.

MOTION

- Array scan: static
- Line-scan: scanning relies on bubble motion

GEOMETRY Bubbles.

MATERIALS Liquid in the bath should not be cloudy or absorb too much light.

REMARKS ➲ *Figure 40.80* shows several possible lighting-viewing methods including dark-field and back-illumination, for array-scan and line-scan sensors. Clearly, it is necessary to choose the best combination for each application, in order to achieve the greatest possible image contrast. When using a line-scan camera, the bubbles should be allowed to reach terminal velocity before they are examined (This will probably require thermostatic control of the bath temperature to limit viscosity variations). Notice that the technique can be used to measure viscosity. When an array-scan camera is used, it may be necessary to use stroboscopic illumination, to "freeze" bubble motion. It may also be necessary to use multiple strobe flashes within a single video frame, or digitise multiple images, to measure terminal velocity.

LAYOUT DIAGRAM ➲ *Figure 40.80*

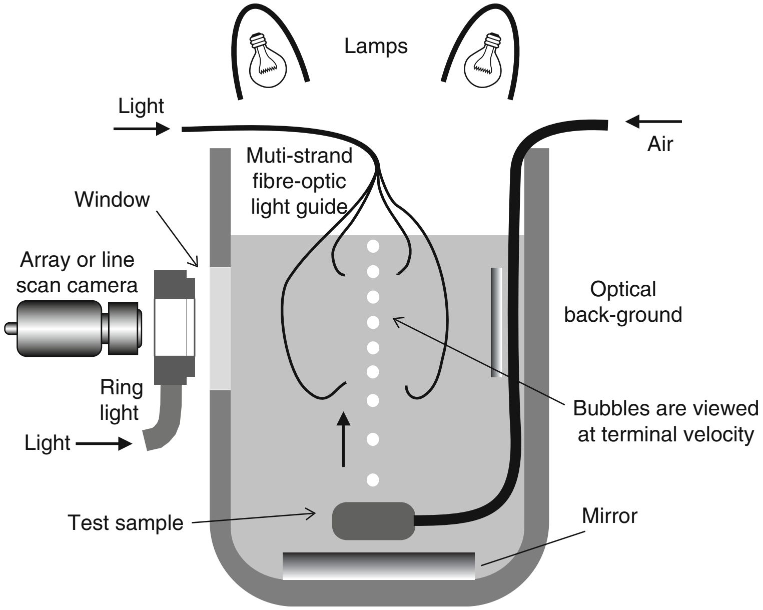

Lamps
Light
Air
Muti-strand fibre-optic light guide
Window
Array or line scan camera
Optical back-ground
Ring light
Bubbles are viewed at terminal velocity
Light
Test sample
Mirror

Fig. 40.80

Observing air bubbles moving in a clear liquid. The top lamps provide flood lighting and are used in conjunction with the mirror at the bottom. The fibre-optic bundle can be arranged to produce very precise projection of the light beam for side or dark-field illumination. The ring light produces front illumination, while the optical background may be bright, dark, or bright with a black patch (for dark-field illumination)

40.85 Method 85: Omni-Directional Illumination

OBJECTIVES

- To provide uniform, all-round (omni-directional) illumination
- To eliminate shadows
- To reduce the effects of glinting
- To see inside deep blind holes

TYPICAL APPLICATIONS

- Viewing complex parts and assemblies, fibrous surfaces, etc. without casting shadows.
- Viewing bright reflecting surfaces without glinting.

EQUIPMENT Circular illumination source: circular fluorescent tube (driven by a high-frequency supply to avoid strobing), fibre-optic ring light, circular array of small bulbs, or LEDs. Hemispherical diffuser, painted matt white or stippled inside, with small viewing aperture at the centre. A cylindrical baffle prevents direct illumination of the widget.

CAMERA Array.

MOTION Static.

GEOMETRY Complicated shapes. Deep holes.

MATERIALS Brightly polished metal, shiny plastic, porous, fibrous.

REMARKS The omni-directional light source is similar to the "light tent" that has been used by photographers for many years when recording images from shiny objects, such as polished silver hollow-ware. Since there are no shadows, step-like features produce very little contrast. It is possible to see inside quite deep holes that would appear dark with standard (oblique) illumination. The effects of specular reflection are slight. (Glinting occurs but does not cause problems, because the hemispherical diffusing surface is of nearly uniform brightness.) Changes in the reflectance and colour of an object with a diffusing surface are clearly visible. The cloudy day illuminator is a proprietry omni-directional source with an adjustable secondary light source and beam-splitter, which compensate for the dark viewing orifice.

REFERENCES

- The use of a hemispherical dome, without the compensating auxiliary source, was first described by SIRA Institute, England, www.sira.co.uk
- NERLITE™ Cloudy day illuminator. RVSI/Northeast Robotics, URL http://www.nerlite.com/. Accessed 5 Sep 2011

LAYOUT DIAGRAM AND SAMPLE IMAGE ➲ *Figure 40.81*

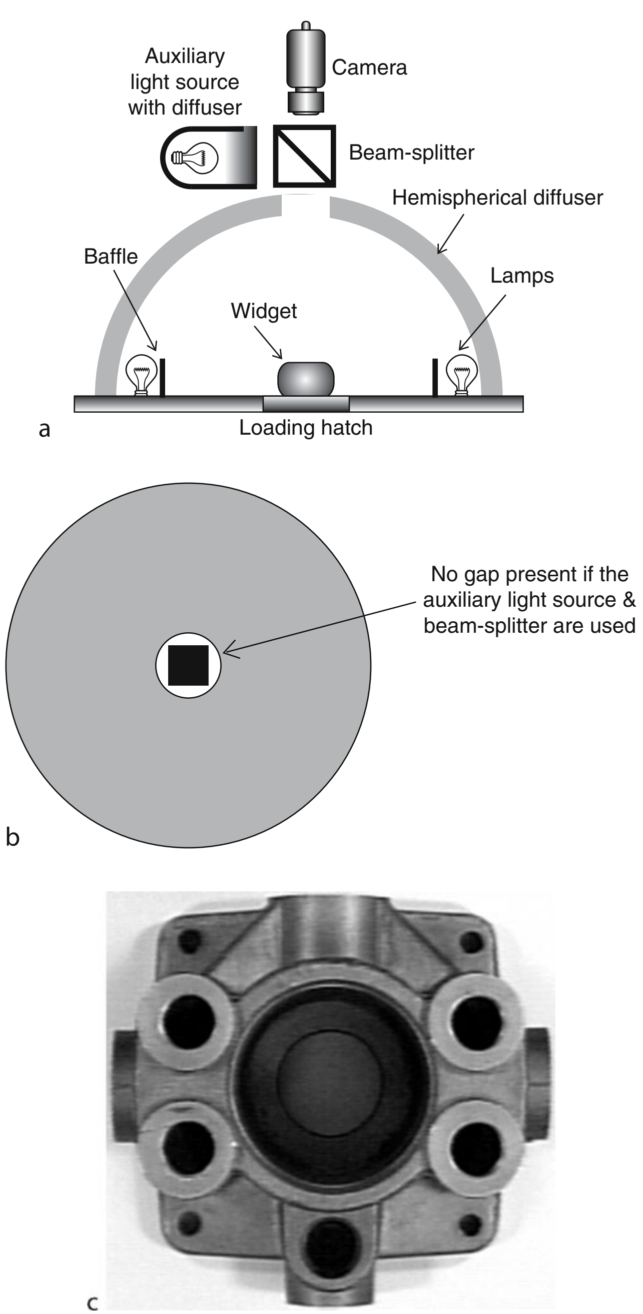
40.86 Method 86: Optically Tooled Bowl Feeder

OBJECTIVE To examine objects as they leave a vibratory bowl feeder, determining their orientation and performing a simple, non-dimensional, check on shape.

TYPICAL APPLICATIONS Parts recognition, determining orientation of parts prior to assembly, handling small components with flexible parts (e.g., brushes used in small electric motors/generators. Since these consist of a graphite block with a copper wire "tail," they are difficult to handle).

EQUIPMENT Vibratory bowl feeder. Illumination can be provided in one of several ways:

(a) Fibre optic device,
(b) Linear LED array or



◘ Fig. 40.81
Omni-directional lighting. (a) Optical set-up of the cloudy day illuminator (cross-section) The omni-directional light source was originally described without the auxiliary light source and beam-splitter. (b) Illumination diagram. (c) Sample image: light-grey aluminium casting, about 120 × 120 mm^2. Step-like features (82 mm) do not cast significant shadows and hence are barely visible. The large circular well is 78 mm deep and its diameter is 56 mm

(c) Laser fitted with a cylindrical lens, generating a fan-shaped beam,

(d) Using a collimator and slit aperture, as shown below. Image acquisition, achieved using a fibre optic ribbon or conduit, allows the camera to be located out of harm's way.

CAMERA Line scan.

LENS None needed – uses fibre optic image conduit instead.

MOTION Motion is provided as parts leave the bowl feeder. Movement is obtained using bowl feeder, slide-way (outside bowl), or using a special pusher.

GEOMETRY Only simple non-interlocking shapes can be separated by a bowl feeder.

MATERIALS Hard metal and plastic parts that are smooth and non-sticky and will not be damaged by the action of the bowl feeder.

REMARKS Vibratory bowl feeders are used to feed small machine parts. As the widgets arrive at the top of the bowl, they reach a slide-way and slip downwards, accelerated by gravity, past the sensing head illustrated in ➋ *Fig. 40.80*. A bowl feeder is not always able to separate widgets in different poses (e.g., travelling "head first" or "tail first") Simple guide rails can achieve some success but they are prone to jamming and are not able to separate widgets in different postures if the differences are subtle. This is precisely the type of situation in which the optically tooled bowl feeder is useful. It may be necessary to use active handling methods (e.g., a pick-and-place robot) to normalise the posture. Alternatively, widgets that are travelling "tail first" might simply be returned to the bowl feeder for recycling. In this case, only widgets travelling "head first" are machined or assembled. Since the speed of travel along the slide-way is uncertain it is not possible to use this method for precise dimensional measurement of widget length (i.e., measured along the slide-way). However, this optical arrangement is also useful for inspecting objects on a conveyor belt, in which case the speed may be known quite precisely.

LAYOUT DIAGRAM ➋ *Figure 40.82*

40.87 Method 87: Passive De-speckling of Laser

OBJECTIVES To remove, or reduce, the speckle seen in an image when using a laser as a source of high-purity monochromatic illumination.

TYPICAL APPLICATIONS Any application in which laser light is useful for illumination but where image averaging, or active de-speckling, is undesirable.

EQUIPMENT One of the following:

- Integrating sphere
- Hollow sphere cut from a suitable diffusing material
- Metal/plastic sphere coated internally with a diffusing paint
- Table-tennis ball painted externally with a matt white paint to improve diffusion (This is a cheap alternative that may suffice for non demanding applications and initial testing purposes)

CAMERA Any.

LENS Any.

REMARKS Light enters the sphere via a small hole and undergoes multiple reflections before leaving through the exit port. The latter should be located at 90° to the input port. These reflections greatly reduce the spatial coherence of the beam and hence diminishes the interference

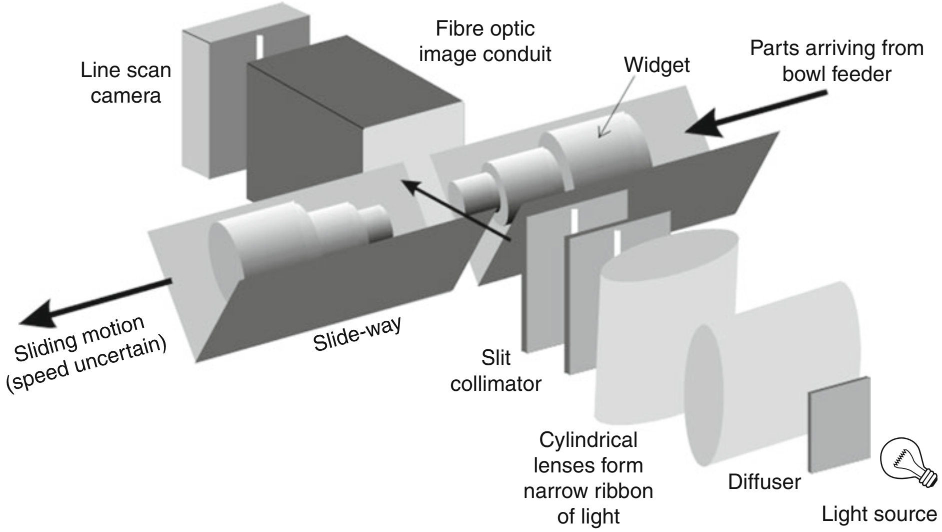

Line scan camera
Fibre optic image conduit
Widget
Parts arriving from bowl feeder
Sliding motion (speed uncertain)
Slide-way
Slit collimator
Cylindrical lenses form narrow ribbon of light
Diffuser
Light source

Fig. 40.82
Inspecting small objects on a slide-way leading from a bowl feeder. Lighting is provided by a single lamp and diffuser. Two cylindrical (rod) lenses and several narrow slits aligned with each other form a collimated "ribbon" of light aligned vertically and traversing the slide-way. A short length of image conduit receives the light and feeds it directly onto the camera array; there is no imaging lens. This keeps the camera away from danger and provides a small robust imaging system

effects that produce speckle. This method works well with sources that have a short coherence length, such as diode lasers. It works less well with long-coherence length devices such as gas lasers. The efficiency of an integrating sphere is low; it is proportional to the ratio of the area of the output port to that of the sphere. On the other hand, a small aperture ensures effective de-speckling. Thus, for a given port size the sphere should be as small as possible. As this ratio increases, however, the de-speckling effect decreases. Alternative methods for de-speckling rely on physical movement of a diffuser or beam deflector. Deliberate high-frequency variation of the laser drive current can be used to reduce speckle further, by causing mode hopping. Artefacts due to laser speckle can also be attenuated by averaging several video frames (The widget must, of course, be stationary and the laser light level constant throughout image acquisition).

REFERENCES

- Labsphere, North Sutton. www.labsphere.com. Accessed 5 Sep 2011 (Visit this site for details of integrating spheres and *Spectralon* light diffusing material)
- Luster S, Light Works, Toledo. www.lw4u.com. Accessed 5 Sep 2011

LAYOUT DIAGRAM ➜ *Figure 40.83*

40.88 Method 88: Pipe or Hollow Cone of Light

OBJECTIVES To provide a "thick-walled" pipe or cone of light.

TYPICAL APPLICATIONS Inspecting deep circular holes, objects with circular symmetry: rims of bottles, jars, other containers.

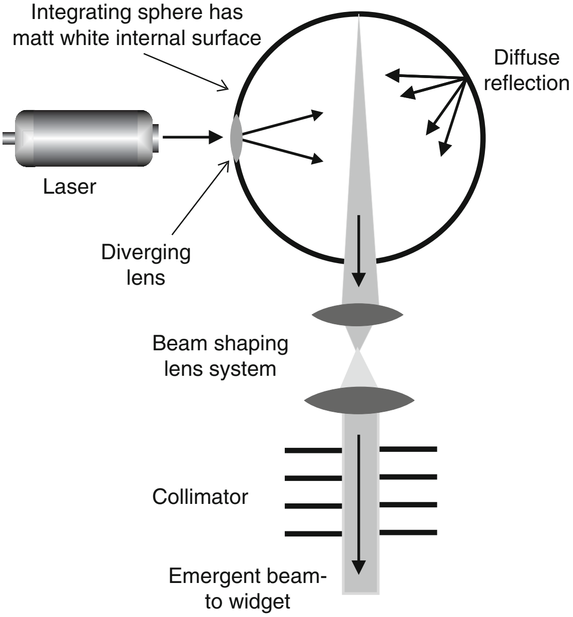


Fig. 40.83
Passive de-speckling of laser radiation

EQUIPMENT

- For a pipe of light: Two “axicons” (An axicon is a conical lens, i.e., a shallow cone of transparent material. It has one plane and one curved surface)
- For a hollow cone of light: One axicon and one lens

CAMERA Array or circular scan.

MOTION Static.

GEOMETRY Object with spherical, circular or radial symmetry.

REMARKS The optical arrangement in *Fig. 40.84a* generates a “pipe” of light. If it is set up properly, this produces a hollow “pipe” of light, with parallel sides. The arrangement in *Fig. 40.84b* produces a hollow “cone” of light. In front of the “focal point,” the cone is converging. It diverges thereafter. When projected onto a plane surface, these two methods generate a bright annulus, with a dark centre. Using one or two axicons makes more efficient use of the available light than merely applying a circular mask to a collimated beam.

REFERENCES http://www.optics.arizona.edu/OPTI696/2005/axicon_Proteep.pdf. Accessed 5 Sep 2011

LAYOUT DIAGRAM *Figure 40.84*

40.89 Method 89: Polar-Coordinate Scan of Annulus

OBJECTIVE To provide a high-resolution view of an annular target. The central disc is ignored and a polar-coordinate map of the intensities within the annulus is created.

TYPICAL APPLICATIONS Inspecting ring-shaped objects, such as

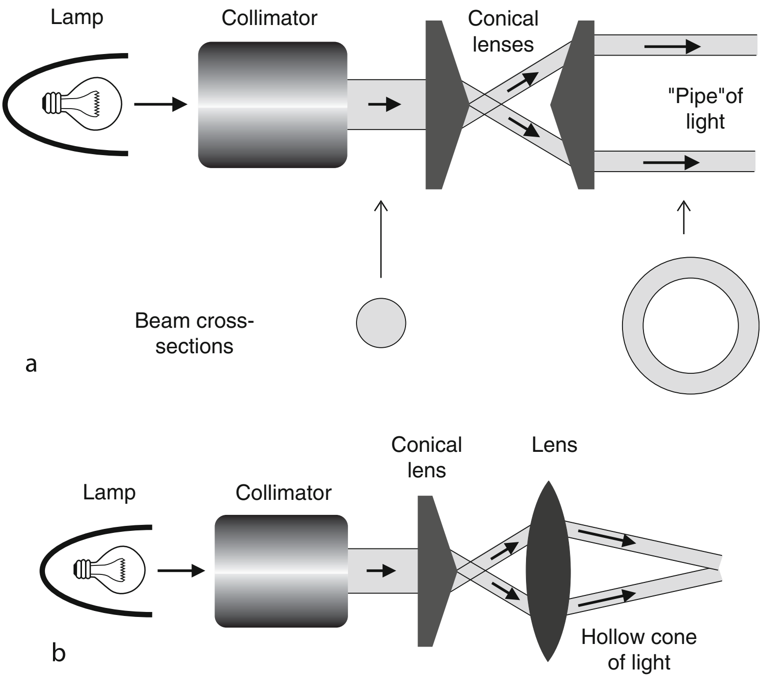


Fig. 40.84

Pipe and cone of light. (a) Two axicons generating a pipe of light. (b) One axicon and a lens produce a hollow cone of light

- Rims of bottles/jars and their lids
- O-rings
- Wheels
- Rims of holes and cylindrical objects

EQUIPMENT Dove prism, plus a simple assembly consisting of two mirrors mounted rigidly together. The latter is rotated at twice the speed of the Dove prism in the opposite direction (A 3-mirror assembly can perform the same function as a Dove prism and does not introduce spherical and chromatic aberrations *Fig. 40.85b*).

CAMERA Line scan.

LENS Telecentric lens (The 3-mirror assembly does not introduce aberrations to the same extent as the Dove prism and does not require a telecentric lens).

MOTION The widget and camera are both stationary. The prism and two-mirror assembly are rotated mechanically.

REMARKS The opto-mechanical scanner illustrated in *Fig. 40.85a* effectively performs a polar-to-Cartesian coordinate mapping of the widget surface; rotation of the two-mirror assembly scans an annulus, while the Dove Prism maintains constant image orientation. An added benefit of using a telecentric lens is that the distance between the mirrors can be adjusted without changing the magnification or focus. This allows a simple variable-radius scanner to be built. While the Dove prism is mechanically stable and easy to mount on a rotating shaft, it

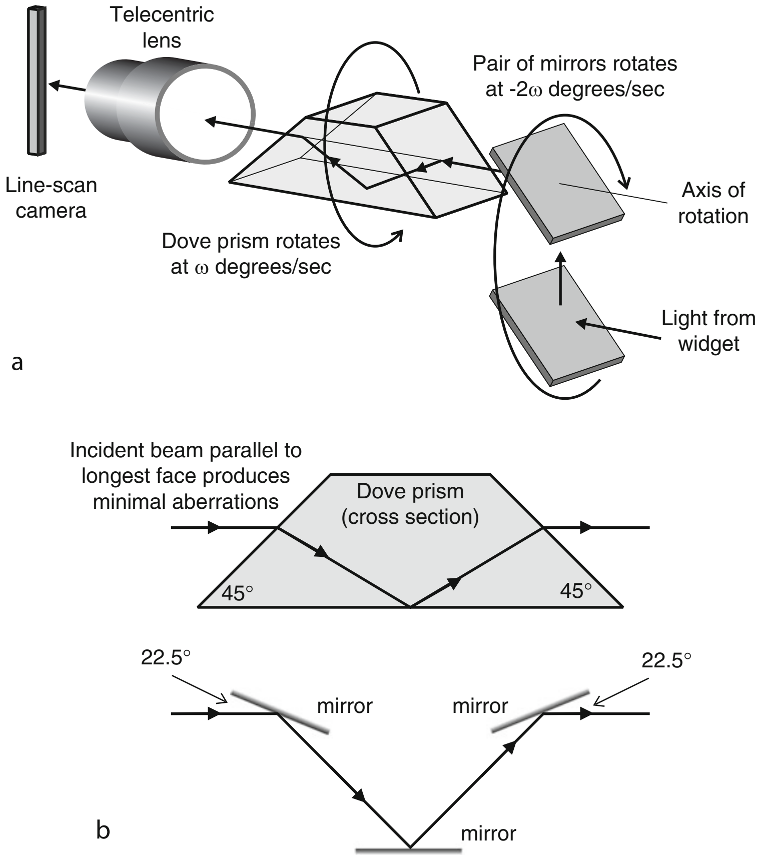

Telecentric lens
Line-scan camera
Pair of mirrors rotates at -2ω degrees/sec
Axis of rotation
Dove prism rotates at ω degrees/sec
Light from widget
a
Incident beam parallel to longest face produces minimal aberrations
Dove prism (cross section)
45°
45°
22.5°
22.5°
mirror
mirror
mirror
b

Fig. 40.85

Circular scanning of an annulus. (a) Optical arrangement. (b) 3 mirrors performing the same function as a Dove prism

does introduce spherical and chromatic aberrations. These can be minimised by ensuring that incident rays are parallel to its long face, using a telecentric lens.

REFERENCES Light Works, URL. http://www.lw4u.com. Accessed 5 Sep 2011

LAYOUT DIAGRAM ➋ *Figure 40.85*

40.90 Method 90: Projected Array of Spots

OBJECTIVE To obtain qualitative, low-precision information about the geometry of an object surface.

TYPICAL APPLICATIONS

- 3-dimensional shape analysis of machine parts, plastic mouldings, die castings, food products (e.g., loaves, cakes, meat pies, etc.)
- Metal and plastic extrusions

EQUIPMENT Industrial (i.e., ruggedised) slide projector. This projects light through a transparency having a regular or irregular, 1- or 2-dimensional array of spots.
CAMERA Array-scan or line-scan.
MOTION Static (for array-scan camera) or linear motion (line-scan camera).
GEOMETRY Arbitrary but known.
MATERIALS Matt, non reflective (Small local areas which are brightly reflecting can be tolerated).
REMARKS When this method was originally developed, a slide containing a regular array of bright spots of light was projected onto the widget (➲ *Fig. 40.86*). This creates a regular array of spots on a plane, although on general, non-planar surfaces, it produces a non-periodic array. The resulting digital image may be difficult to analyse, particularly if there are steep-sided cliff-like features present. The can cause some spots to be become "lost" due to occlusion. This may be acceptable, if the ideal form of the widget is fixed and is known with some precision. In this case, the inspection procedure consists of simply checking that there is a spot present in each position where a spot has been encountered before and that there are no spots visible elsewhere. However, when it is possible to anticipate the shape of the widget, it is usually preferable to arrange the spots so that the camera sees a regular pattern of bright points on the surface of a well-formed widget (➲ *Fig. 40.86b*). The resulting image is easier to analyse. A 1-dimensional version of this optical arrangement, based on a line-scan camera, can be used to good effect, when inspecting extrusions.
REFERENCES

- Will PM, Pennington KS, Grid coding preprocessing technique for robot and machine vision. In: Proceedings of the 2nd joint conference on artificial intelligence. British Computer Society, London, pp 66–70
- Parks JR (1978) Industrial sensory devices. In: Batchelor BG (ed) Pattern recognition: ideas in practice. Plenum, London/New York. ISBN 0-306-31020-1

LAYOUT DIAGRAMS ➲ *Figure 40.86*

40.91 Method 91: Protecting the Camera Using a Periscope

OBJECTIVE To fold the optical path, so that the camera can be situated in a convenient, safe position, well away from the widget.
TYPICAL APPLICATIONS

- Monitoring operations such as casting, grinding, cutting, welding, etc. that create a lot of flying debris
- Inspecting very hot, steaming, or spluttering widgets
- Inspecting objects within an explosive, toxic or radioactive atmosphere, where people cannot work safely
- Placing the camera in a convenient position, where it is easy to service or adjust
- Inspecting objects that are immersed in water, oil, etc. (➲ *Fig. 40.87b*)

EQUIPMENT Plane mirrors, preferably front silvered. Notice that 45°/45°/90° prisms may be used in lieu of mirrors.
CAMERA Array or line-scan.
MOTION Static (array-scan camera) or linear motion (line-scan camera).

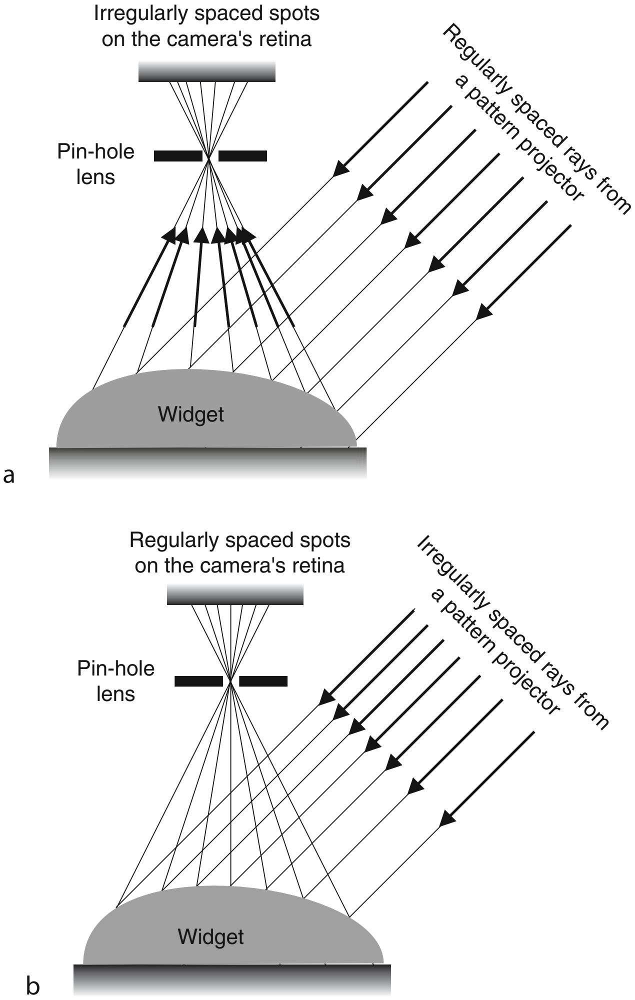


Fig. 40.86

Projecting an array of spots. (a) A regular array produces a non-periodic pattern of bright points on the camera's retina. A pin-hole lens is shown here for simplicity of illustration. In practice, a normal glass or plastic lens would be used. (b) If the form of the surface is known, an irregular array of projected spots can be used to create a periodic pattern of bright points on the object surface. The idea implicit in these two diagrams can be extended into 2-dimensions

REMARKS The periscope is a versatile device. By adding extra mirrors, it can be used to examine objects that are displaced both vertically and horizontally. It can allow viewing within a confined space, where access for servicing/adjusting the camera would be impossible.

LAYOUT DIAGRAM ➋ *Figure 40.87*

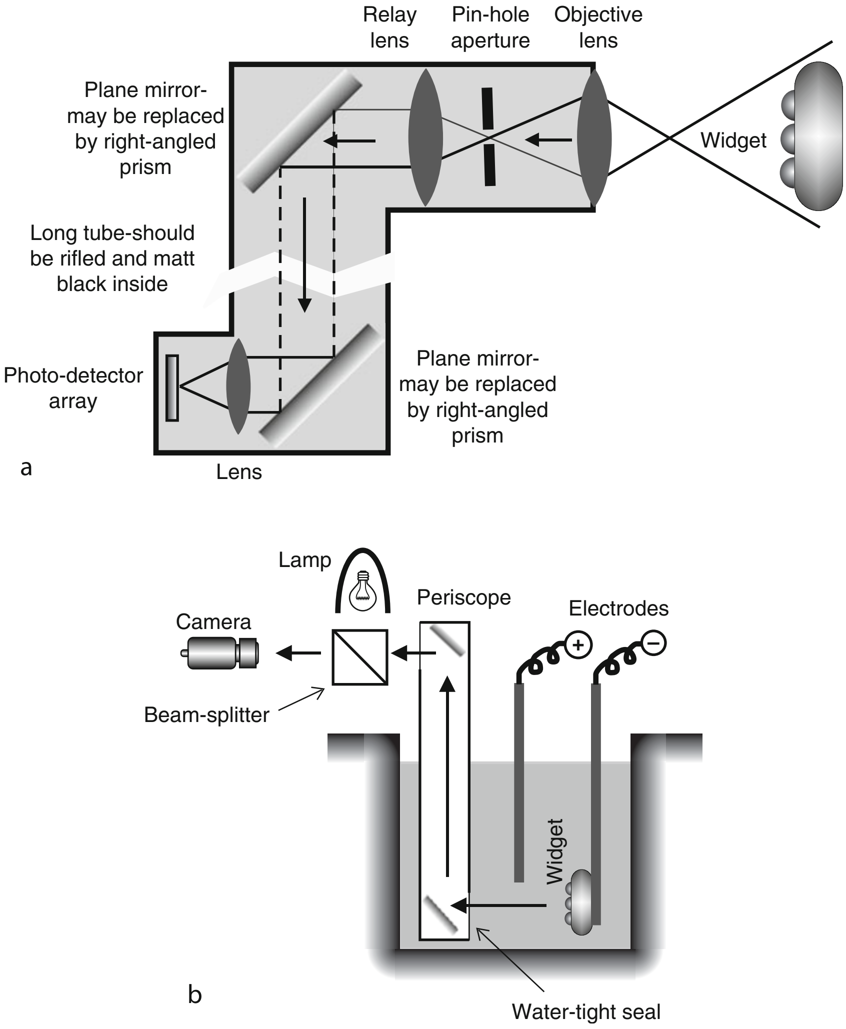

Relay lens
Pin-hole aperture
Objective lens
Plane mirror-may be replaced by right-angled prism
Widget
Long tube-should be rifled and matt black inside
Photo-detector array
Plane mirror-may be replaced by right-angled prism
a
Lens
Lamp
Periscope
Electrodes
Camera
+
−
Beam-splitter
Widget
b
Water-tight seal

Fig. 40.87
Periscope. (a) Optical arrangement. (b) Examining an object within a plating bath. The lighting arrangement shown here avoids immersion and is suitable for bulky high-voltage lamps. This is just one of many possibilities; LEDs and fibre optics, located within the periscope, could be used instead

40.92 Method 92: Protecting the Camera with Fibre Optics

OBJECTIVE To remove a camera from a dangerous environment, by transmitting the image along a coherent fibre optic image bundle directly onto the camera. The fibre optic bundle may be flexible, or rigid bundle (called *image conduit*), or in the form of a flexible ribbon.

TYPICAL APPLICATIONS Inspecting objects in a hazardous environment, which may be very hot, very cold, corrosive, explosive, toxic, biologically contaminated, radioactive, or in an intense x-ray, UV or microwave beam, or be at a high electrical potential.
EQUIPMENT Fibre optic ribbon, flexible fibre optic bundle, or image conduit.
CAMERA Line-scan, or array-scan.
MOTION Static, or continuous linear/rotary movement.
REMARKS Optical fibre ribbon can be purchased, or made in the laboratory, using the following process. Wind a neat coil of optical fibre onto a large drum. Secure the fibre to the drum at two points, by placing two lengths of adhesive tape along the coil and close together. Cut the coil between these points. The ribbon can then be removed from the drum. The ends are then encapsulated in epoxy resin, machined flat and polished. A flexible fibre-optic bundle can be used with an array camera. If flexibility is not important, image conduit may be used instead (This is a rigid rod-like assembly of optical fibres that preserves image integrity. It can be bent to shape after heating in a gas flame.). Specialised fibre-optic devices are available commercially and perform a variety of useful functions, including quarter-twist, 90° bend, change of image size (magnification/minification), change of aspect ratio. The sensor array may be interfaced directly to the fibre bundle, as in ➲ *Fig. 40.88*, or an intermediate lens may be added. Care must be taken to avoid creating Moiré effects, by matching the pitch of the fibre matrix to that of the camera array.
REFERENCES

- Cronshaw AJ (1979) Software techniques for an optically tooled bowl feeder. In: IEE conference on trends in on-line computer control techniques. Institution of Electrical Engineers, London. IEE conf. pub. 172, pp 145–150
- Image Conduits, Edmund Optics, URL http://www.edmundoptics.com/. Accessed 5 Sep 2011

LAYOUT DIAGRAMS ➲ *Figure 40.88*

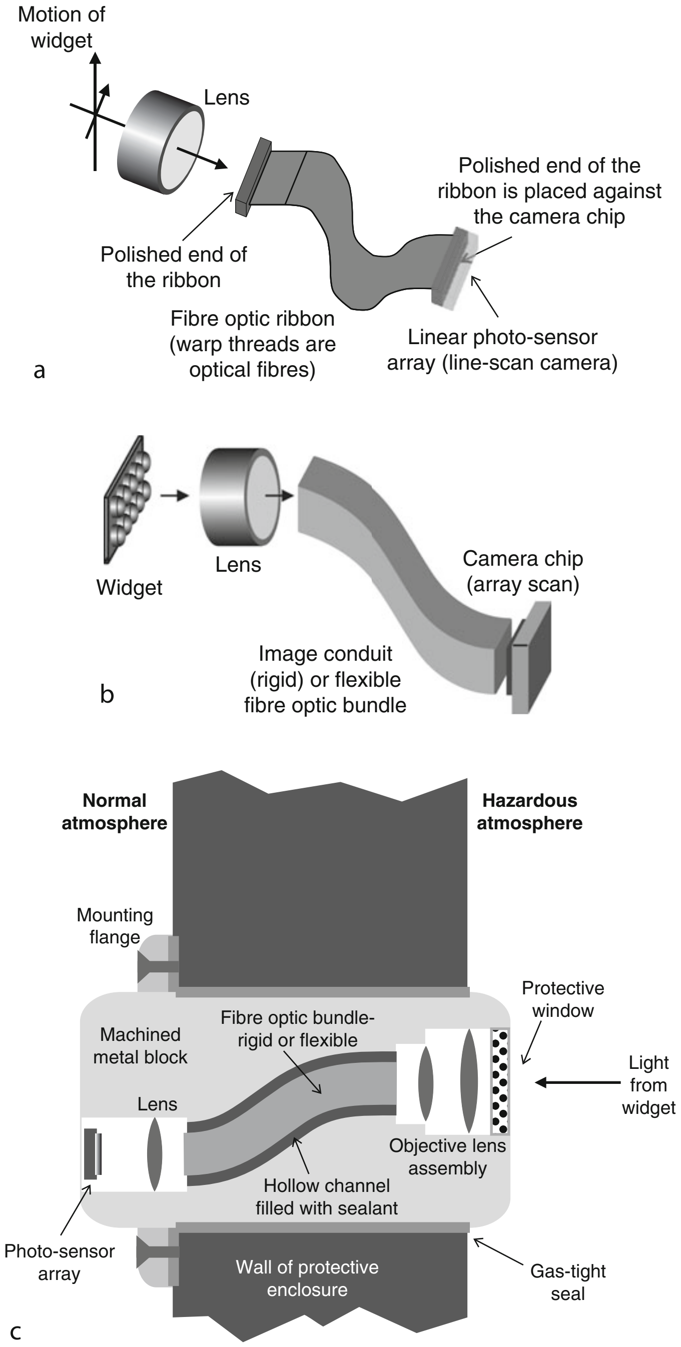
40.93 Method 93: Protecting the Camera

OBJECTIVE To protect the camera and lens from damage due to

- Hot/cold air
- Condensation
- Infra-red radiation
- Fine airborne pollution (dust, mist, fumes, smoke, aerosols, paint, coating and cleaning chemicals, etc.)
- Flying particulate debris (saw-dust, swarf, grinding sparks, etc.)
- Flying droplets (water, oil, coolant, adhesive, paint, etc.)
- Scheduled cleaning (particularly in food and pharmaceutical factories)
- Tampering, theft, sabotage
- Impact (both accidental and deliberate)
- Blast from explosive processes
- X-rays and nuclear radiation

TYPICAL APPLICATIONS

- Inspecting hot steel slabs and sheet, loaves and cakes in the baking oven, etc.
- Controlling/monitoring the cutting of timber and metal, injection moulding


Motion of widget
Lens
Polished end of the ribbon is placed against the camera chip
Polished end of the ribbon
Fibre optic ribbon (warp threads are optical fibres)
Linear photo-sensor array (line-scan camera)
a
Widget
Lens
Camera chip (array scan)
Image conduit (rigid) or flexible fibre optic bundle
b
Normal atmosphere
Hazardous atmosphere
Mounting flange
Protective window
Fibre optic bundle- rigid or flexible
Machined metal block
Light from widget
Lens
Objective lens assembly
Hollow channel filled with sealant
Photo-sensor array
Wall of protective enclosure
Gas-tight seal
c

◘ Fig. 40.88

Protecting the camera using a coherent fibre optic bundle. (a) Using a line-scan camera and fibre optic ribbon. (b) Array camera and image conduit or coherent flexible fibre optic bundle. (c) Building a camera into the wall of a protective enclosure

- Measuring aerosol spray cone
- Controlling painting, spraying, etc.
- Remote visual control in areas providing radiation, toxic or biological hazards

EQUIPMENT *General* Rugged box with a tough optical-quality window at the front end and providing a broad range of environmental protection measures. Internal surfaces of the enclosure should be matt black. Not all of the features shown in ➲ *Fig. 40.89a* are needed in every application.

Radiation Lead box. Radiation resistant optics are needed.

CAMERA

- Array- or line-scan camera
- Tube camera and valve electronics may be needed in very radio-active areas, as solid-state devices are quickly damages

REMARKS Heat dissipation by the camera electronics should be considered when designing an enclosure, to avoid overheating. Similar environmental protection techniques are appropriate for protecting small illumination sources (e.g., LEDs) Notice too that the product must also be protected from the camera and lights, to ensure parts (e.g., glass lenses, screws, etc.) do not fall off and contaminate critical products (e.g., food/pharmaceuticals). In very radio-active areas, a tube camera may be required, since solid-state electronics is susceptible to radiation damage. Special types of glass and insulation may be needed.

LAYOUT DIAGRAM ➲ *Figure 40.89*

40.94 Method 94: Range Measurement Using Talbot Fringes

OBJECTIVE To measure 3D surface shape by projecting Talbot fringes onto the widget surface.

TYPICAL APPLICATIONS Inspecting small objects which have a simple geometry.

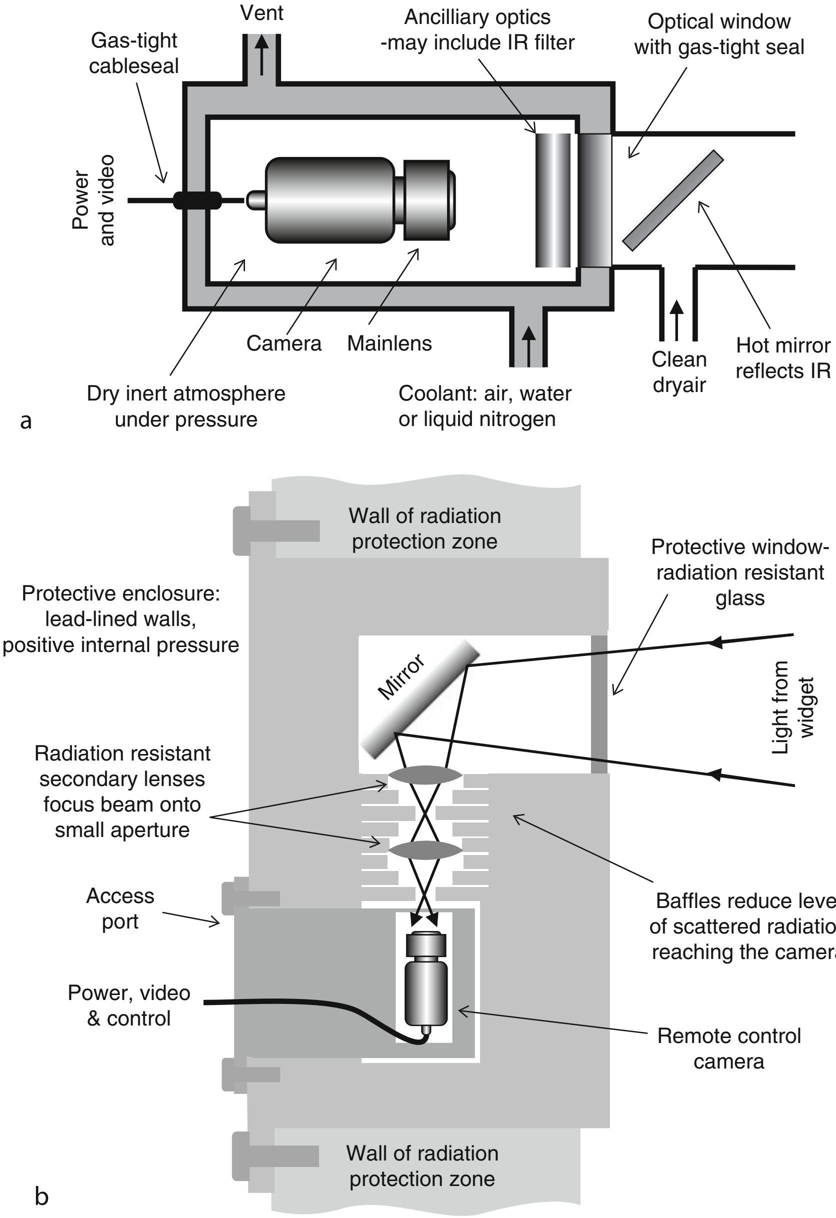
EQUIPMENT Laser, optical grating.

CAMERA Array.

MOTION Static.

MATERIALS The widget must have uniform reflectivity. To achieve this, it may be necessary to coat it. The developer spray used for dye-penetrant crack detection is ideal, as it can be applied as an aerosol spray. Talc ($Mg_3Si_4O_{10}(OH)_2$) or ammonium chloride (NH_4Cl) may be used for this purpose.

REMARKS When a coherent beam of light is transmitted through a coarse grating, a series of sharply focussed images of the grating (*Talbot images*) is generated at regular intervals along the optical axis. This effect was first reported in 1836 by Henry Fox Talbot but was then forgotten until it was rediscovered by Lord Rayleigh in 1881. The spacing between these images, the so-called *Talbot period*, is given by A2/λ, where A is the slit spacing of the grating and λ is the wavelength of the light. This is based on the assumption that A >> λ. For red light with a wavelength of 632.8 nm (the wavelength of light produced by a HeNe laser) and a grating with 50 slits per inch (a = 0.508 mm) the Talbot period is 407.8 mm. See ➲ *Fig. 40.90*. For accurate range measurement, the object should be uniformly reflective over its whole surface. The range is measured by estimating the local contrast in the digital image captured from the camera. Large objects in which the variation in height exceeds the Talbot period cannot be measured directly, since adjacent fringes become confused.



▪ Fig. 40.89

(a) Combination of techniques for protecting the camera and lens. Even if no air purge is used, a long thin tube around/in front of the lens provides an effective dust trap (However, this may affect both camera placement and cause vignetting). (b) Protecting the camera from nuclear radiation and x-rays

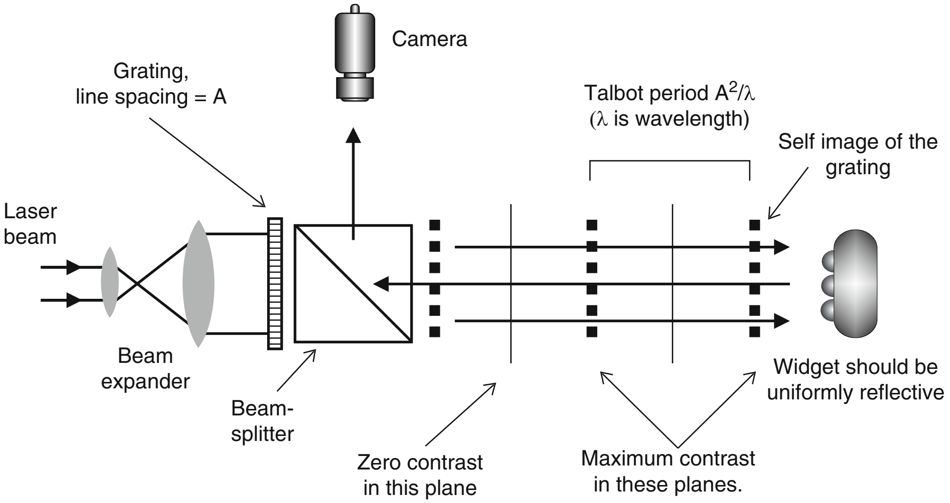

Camera
Grating,
line spacing = A
Talbot period A²/λ
(λ is wavelength)
Self image of the grating
Laser beam
Beam expander
Beam-splitter
Zero contrast in this plane
Maximum contrast in these planes.
Widget should be uniformly reflective

Fig. 40.90

Talbot Effect. Alternating positive and negative self-images of the grating are created at intervals of A^2/λ. Planes of zero contrast lie in between. The beam-splitter provides coaxial illumination and viewing and hence avoids obscuration

REFERENCES

- Berry M, Marzoli I, Scheich W, Quantum carpets, carpets of light. http://physicsweb.org/article/world/14/6/7/1/#17. Accessed 6 Sep 2011
- Ambrosini D, A bibliography about the Talbot-Lau Effect. http://dau.ing.univaq.it/omhat/Ref/talbib.htm. Accessed 6 Sep 2011
- Besl PJ (1988) Active, optical range imaging systems. Machine Vision Appl 1:127–152
- Flawfinder Developer Spray, Rocol, Leeds. www.rocol.com. Accessed 6 Sep 2011

LAYOUT DIAGRAM ➜ *Figure 40.90*

40.95 Method 95: Reading Printing in Photo-Chromic Ink

OBJECTIVE To obtain an image of text or a data code (e.g., bar code, or dot matrix code) printed with photo-chromic (PC) ink.

TYPICAL APPLICATIONS There are numerous potential applications in security and document/product authentification.

EQUIPMENT

- Ultra-violet flash lamp (to switch the colour of the PC ink)
- Visible (VIS) flash lamp (to switch the ink back to its original colour)
- Low intensity visible lamp (for viewing)

CAMERA Colour array scan.

MOTION Static.

MATERIALS Avoid red/pink surfaces for red-transparent PC ink.

REMARKS Photo-chromic ink changes colour when it is irradiated. Typical changes are as follows:

- Ultra-violet – colour changes to pink-red
- Visible light (VIS) – colour changes to transparent

Other colour combinations are possible. In its transparent state, PC ink is difficult, but not impossible, to see by eye. Switching between colour states is very fast and can easily be accomplished during the inter-line scan interval of a video signal. (Rapid colour switching occurs as a result of a change in the *geometry* of an organic ring-shaped molecule; there is no chemical change. The reader should not confuse the PC ink being described here with the more familiar photo-chromic spectacles, in which the colour change is much slower. Moreover, PC ink has two stable colour states, unlike PC spectacles, which have one stable state and one quasi-stable state.) To isolate photo-chromic ink from normal ink, we might use the following procedure:

(a) Flash the UV lamp. Digitise an image (Call this *I1*.)
(b) Flash the high-intensity VIS lamp. Digitise an image (Call this *I2*.)
(c) Subtract images *I1* and *I2*
(d) Threshold the difference image

This is effective in highlighting the PC ink and obscuring the normal ink. PC ink can be applied using a variety of printing methods, including ink-jet printers. It can therefore be applied under program control. When programmable PC printing is combined with high-level encryption and optical character recognition (OCR) software, it is possible to provide a message, or product, with a high degree of security. The PC ink effectively authenticates the document, while its message is protected by the encryption process. People must be protected during colour switching, by the use of appropriate shielding. (Ultra-violet light can damage eyes and skin, while a high intensity VIS flash can induce epileptic fits and migraine episodes.)

REFERENCES

- Heller HG (1991) Photochromics for the future. In: Miller LS, Mullin JB (eds) Electronic materials, from silicon to organics. Plenum, New York
- Batchelor BG, Stephens NM (1993) Image analysis of photochromic ink for security applications. In: Proceedings of the SPIE conference on intelligent robots and computer vision XII algorithms and techniques, Boston, 7–10 Sept 1993. SPIE- The International Society for Optical Engineering, vol 2055, pp 310–323. ISBN 0-8194-1320-8

LAYOUT DIAGRAM ➋ *Figure 40.91*

40.96 Method 96: Real-Time Opto-Electronic Filtering (High-Pass)

OBJECTIVE To generate digital images that have been pre-processed by high-pass filtering.
EQUIPMENT Two matched, aligned cameras,. One camera is slightly mis-focussed. The other is sharply focussed. Their output signals are subtracted electronically, either in analogue or digital circuitry.
CAMERA Array.
LENS Special arrangement.
MOTION Static.

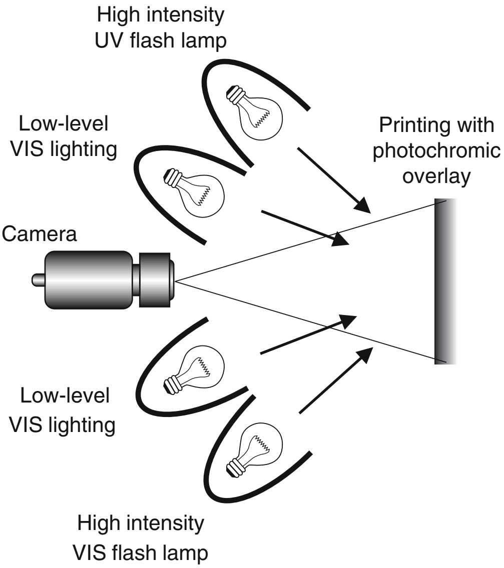


Fig. 40.91
Reading Photo-chromic ink

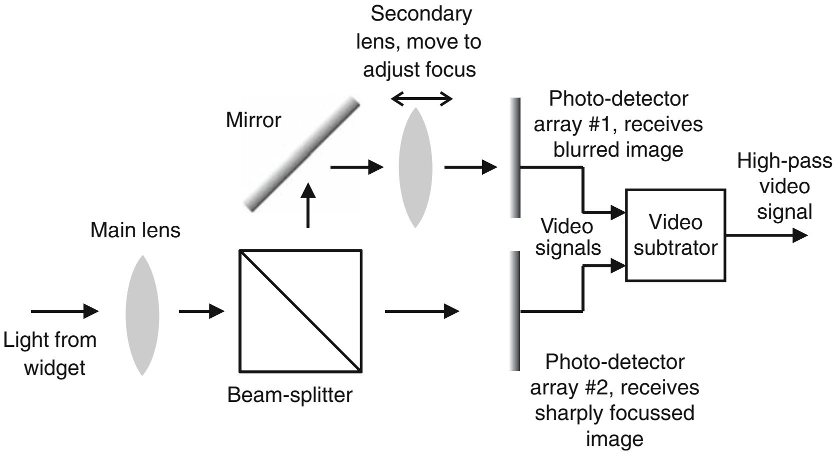


Fig. 92
Layout for opto-electronic filtering

REMARKS This produces results that are similar to those generated photographically, by unsharp masking. High-pass filtering emphasises local intensity changes. Small spots and narrow streaks are emphasised. Edges are also emphasised but less so than by a digital edge-detection operator.

LAYOUT DIAGRAM ➋ *Figure 40.92*

40.97 Method 97: Ring of Light Projecting Inwards

OBJECTIVE To provide a ring of lamps which projects light inwards, thereby generating all round grazing illumination.

TYPICAL APPLICATIONS

- Inspecting coins and similar items with engraving and/or embossing on a flat surface
- Highlighting chamfers on machined/moulded components

EQUIPMENT Ring of small light sources, such as LEDs or optical fibres. Alternatively, a fibre-optic ring light may be used in conjunction with a conical mirror (➋ *Fig. 40.93d*).

CAMERA Array.

MOTION Static.

GEOMETRY Flat, with minor depressions or raised areas.

MATERIALS Shiny materials.

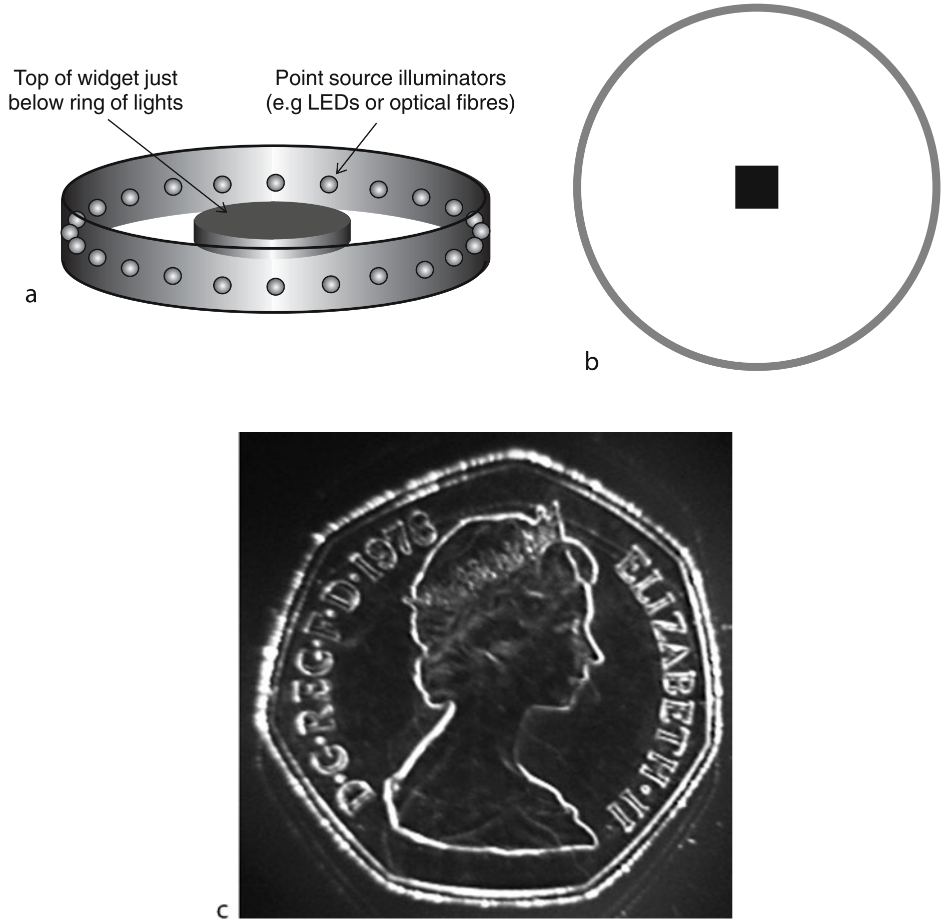

Top of widget just below ring of lights
Point source illuminators (e.g LEDs or optical fibres)
a
b
c
D·G·REG·F·D·1978
ELIZABETH·II

Fig. 40.93
(Continued)

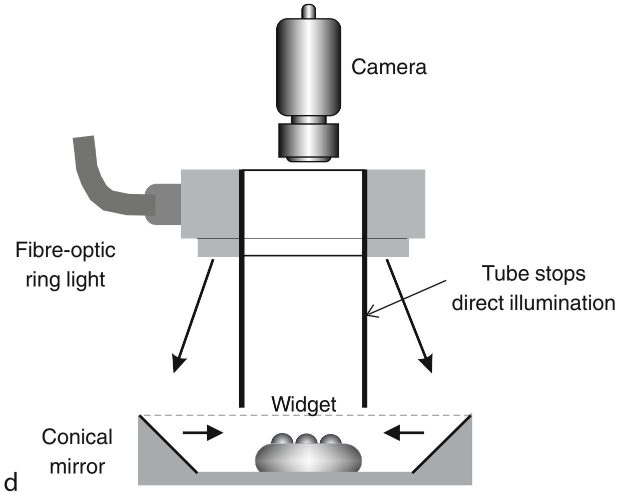


Fig. 40.93
Ring of light projecting inwards. (a) Optical layout. (b) Illumination diagram. (c) Sample image: Coin (British 50 pence). A multi-point source (i.e., a ring of optical fibres) was used here. (d) Using a fibre-optic ring light and a conical mirror

REMARKS The technique works best if the widget is held so that its flat surface is horizontal and in a fixed vertical position. Commercially available low-angle ring lights are sometimes described as "dark field lights."
LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.93*

40.98 Method 98: Rotate Image Using a Dove Prism

OBJECTIVE To rotate an image by a fixed amount, to make subsequent image analysis easier.
TYPICAL APPLICATIONS Looking for lines of known orientation. Analysing texture, where the pattern has a known form, with dominant features at fixed angles. If θ is the orientation of the prism, this method will emphasise linear features at angles 2.θ and 90° + 2θ.
EQUIPMENT Dove prism. A conventional 45°/45°/90° prism will suffice but this is larger and heavier than a Dove prism.
CAMERA Array or line-scan camera, as appropriate for the application.
LENS Standard lens fitted to camera. It may be necessary to fit an extra lens in front of the prism.
MOTION Static or moving, as appropriate for the application.
REMARKS Optical image rotation is usually preferred to software, for the following reasons:

- The optical system does not degrade the image significantly, whereas rotating a digital image produces ragged edges, due to quantisation effects
- Optical systems have a far higher resolution
- There is no increase in the computational load. This makes the optical system faster than one based on software

- The optical system has minimal running costs
- The rotation can be altered by simply turning the prism by hand. This allows easy real-time interaction during the initial alignment and calibration of the system

Notice that the image is rotated by an angle 2.θ where θ is the orientation of the Dove prism. The advent of miniature cameras makes this method less important than hitherto.

A variable angle of rotation can be achieved, using a motorised Dove prism. Hence, polar-coordinate raster scanning can be implemented, by rotating the Dove prism continuously and using a line-scan camera ➋ *Fig. 40.94c*. This is particularly useful for viewing objects that possess several concentric circular features, are radially symmetric or have a "wheel-spoke" appearance. A fused fibre optic bundle (image conduit) can also rotate an image. Image conduit is made with various standard rotations: ±90° and 180°. Other values can be made on request. The conduit can be bent into an L- S-, or U-shape and is a very robust means of transmitting an image. It is also able to function in a smoky or fume-laden atmosphere. The image rotation can be performed by a 3-mirror arrangement, sometimes called an Abbe K-mirror.

REFERENCE Lighting and Imaging Machine Vision, Schott. http://www.schott.com/. Accessed 6 Sep 2011

LAYOUT DIAGRAM ➋ *Figure 40.94*

40.99 Method 99: Silhouette from Multiple Shadows

OBJECTIVE To obtain a high-contrast silhouette of an object by combining several views, obtained using lighting from different directions.

TYPICAL APPLICATIONS This method can be useful for inspection applications, where there is limited rear access and there is little or no contrast between the widget and its background.

EQUIPMENT Computer controlled lights.

CAMERA Array.

MOTION Static over several video frames. Vibration cannot be tolerated.

MATERIALS Virtually any opaque material.

REMARKS This method is suitable for use in those situations where back-lighting is impossible and there is little, or no contrast, between the widget and its back-ground. If oblique lighting is used from a single point source, the widget casts a shadow that is offset to one side. This increases the contrast around just part of the edge. However, that part of the edge nearer the light source still has low contrast. If the light source is now moved around the widget, a different part of the silhouette will be well defined. In practice, it may be necessary to provide illumination from several different directions (at least four) in turn. For each lighting direction, a digital image is captured. These images are then combined using a simple image processing algorithm: Let *a(i,j,k)* be the intensity at the point *[i,j]* in image *k*. Then, we generate an image *{b(i,j)}* by computing

$$b(i,j) = MIN(a(i,j,1), a(i,j,2), a(i,j,3), \ldots)$$

for all *[i,j]*. In this way, it is often possible to create a continuous high-contrast edge in the combined image, *{b(i,j)}*, that no single view could achieve. Notice that the inclined lighting causes errors in estimating the position of the outer edge of the widget (➋ *Fig. 40.95b*). Their

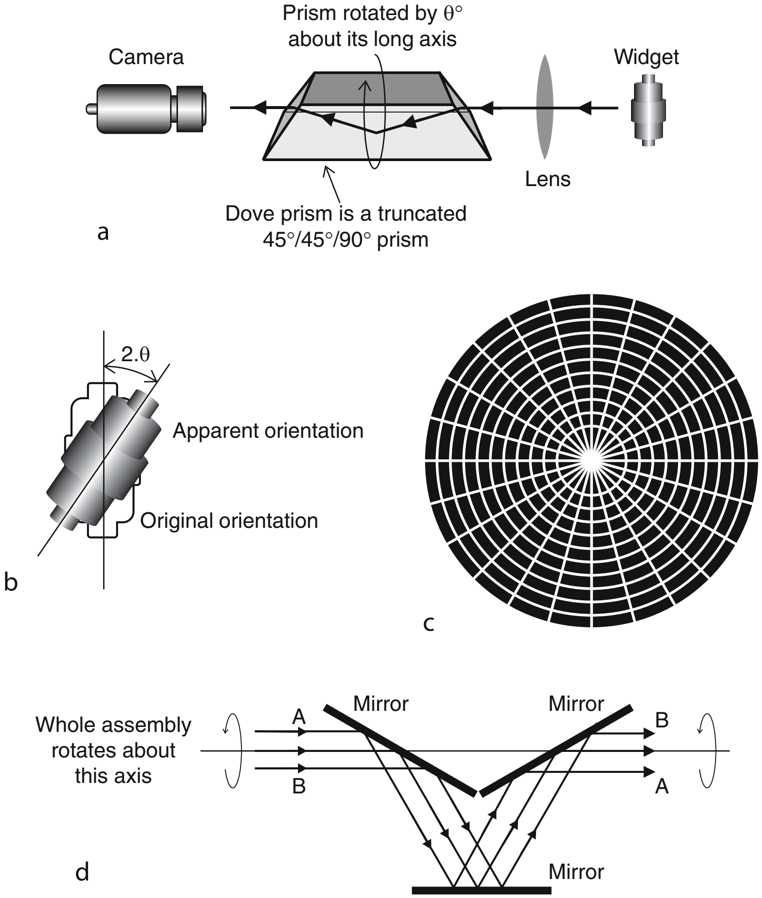

Prism rotated by θ° about its long axis
Camera
Widget
Lens
Dove prism is a truncated 45°/45°/90° prism
a
2.θ
Apparent orientation
Original orientation
b
c
Whole assembly rotates about this axis
Mirror
Mirror
A
B
B
A
Mirror
d

Fig. 40.94

Using a Dove prism to rotate an image by a fixed amount. (a) Optical arrangement. (b) The camera sees an image that is rotated by 2θ, where θ is the orientation of the Dove prism. (c) Digitisation pattern obtained by rotating the Dove prism continuously and using a line-scan camera. (d) Abbe K-mirror

extent depends upon the 3D shape of the widget near the edge. For example, chamfers could cause an under-estimate of the size of the widget. For this reason, it may be necessary to place the lamps close to the camera, to ensure that the lighting is nearly normal.
LAYOUT DIAGRAM ➜ *Figure 40.95*

40.100 Method 100: Silhouette in Restricted Space

OBJECTIVES To view an opaque, object in silhouette, in situations where there is limited rear access.
TYPICAL APPLICATIONS In-process inspection of objects as they are being machined.

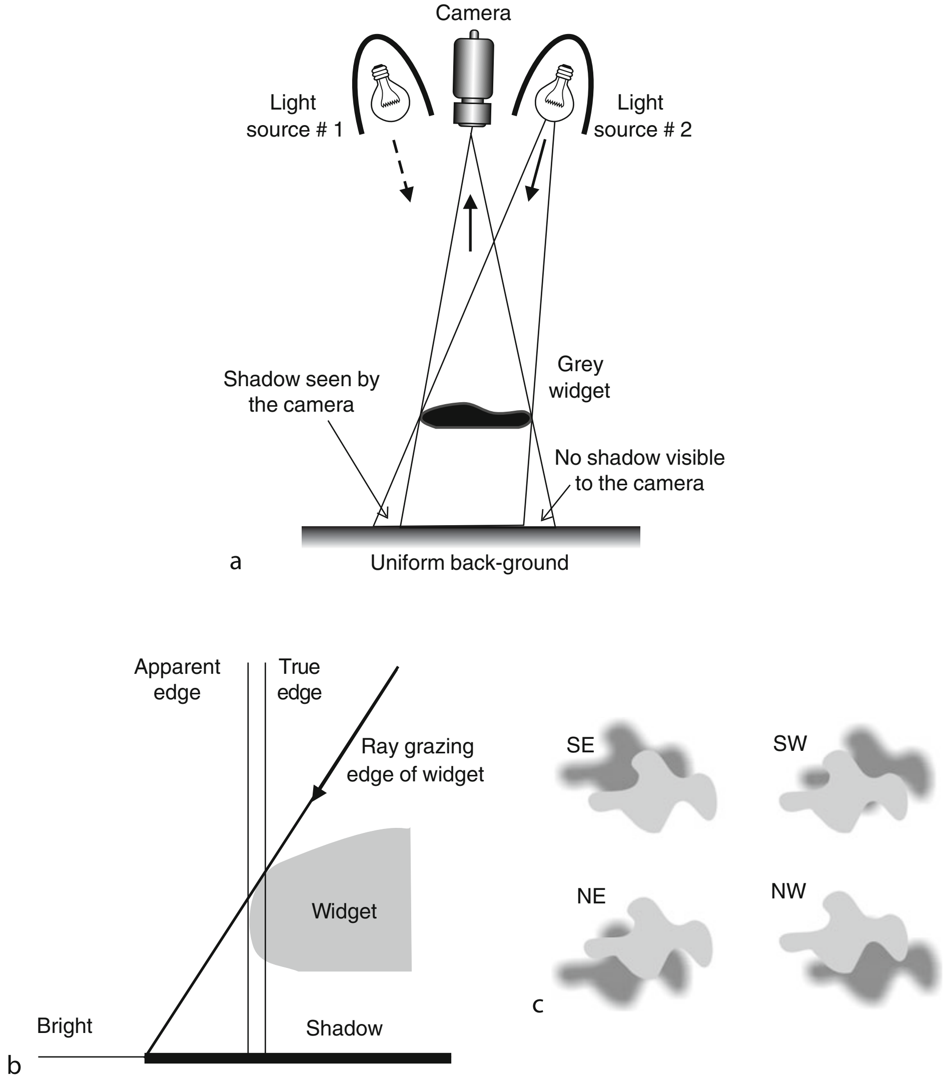


Fig. 40.95
Obtaining a silhouette of an opaque widget by creating multiple shadows, in sequence. (a) Optical layout. Only one lamp is lit at any given moment. (b) Inclined lighting causes errors in estimating the position of the outer edge of the widget. (c) Individual shadows increase the contrast around only part of the outer edge. These images are combined by simple image processing to produce a high-contrast edge all round the widget

EQUIPMENT Diffusing filter, beam-splitter, retro-reflective film.
CAMERA Array- or line-scan.
MOTION Static or linear/circular motion.
MATERIALS Matt, non-shiny.

REMARKS In normal room lighting, a retro-reflector appears to be light grey in colour. However, when a source of light is placed very close to the line between the eye and the retro-reflector, it shines brilliantly: far brighter than a sheet of white paper. Hence, in this optical arrangement, the retro-reflector provides a bright background for the widget. To obtain an accurate silhouette of the widget, the primary illumination should be from a diffuse source; an accurately collimated beam may miss the camera entirely after it has been reflected. If the widget is close to the retro-reflector during dirty operations, such as milling, spraying, etc. there is a danger that it will quickly become dirty, thereby reducing its effectiveness. In this case, the retro-reflector should be protected behind a clear glass screen that can easily be wiped clean. (Retro-reflective film cannot be cleaned so easily, without damaging its surface.) Retro-reflective material can be purchased from automobile spares retailers. Flat panel units with LED or cold cathode illumination are alternatives.

REFERENCES Scotchlite (retro-reflective film, product of 3M Company, St. Paul). Wikipedia, http://en.wikipedia.org/wiki/Scotchlite. Accessed 4 Feb 2011

LAYOUT DIAGRAM ➋ *Figure 40.96*

40.101 Method 101: Size Measurement in Free Fall

OBJECTIVE To measure size and shape parameters of a small object, such a drop of fluid, in free fall.

TYPICAL APPLICATIONS

(a) Measuring and counting wall/oil drops emerging from a nozzle

(b) Calibrating fluid control valves

(c) Checking/measuring the flow of cutting fluid during the machining of metal components

EQUIPMENT

- Optical proximity sensor, to indicate when the object is in front of the camera
- Strobed light box, with a dark patch at its centre, to provide dark-field illumination

CAMERA Asynchronous (progressive scan) camera, capable of digitising an image immediately, the trigger signal is received from the proximity sensor.

MOTION Free fall.

MATERIALS Opaque or transparent solid. Clear or opaque fluid.

REMARKS The secondary (LED) light source, within the proximity sensor, must not be allowed to reduce image contrast, by illuminating the drop of oil/water during image capture. Hence, it may be necessary to place an IR-blocking filter immediately in front of the camera lens. Alternatively the LED may be switched off during image acquisition, just before the back-light strobe is triggered. Strobing is needed, because the drops are travelling quite fast: a small drop of oil/water that has previously been in free fall, through a distance of 100 mm, is travelling at about 1.5 ms^{-1} and hence travels about 60 mm during a CCIR video frame-period. Compare this with the movement during the strobe flash (typically, 0.01–0.1 ms). During this time interval, the drop falls by about 0.015–0.15 mm. This motion effectively limits the vertical resolution that can be achieved by this method. If possible, the drop should have reached its terminal velocity before it arrives within the camera's field of view. This ensures that the drop is in standard, known position, thereby allowing the maximum possible spatial resolution to be achieved.

LAYOUT DIAGRAM ➋ *Figure 40.97*

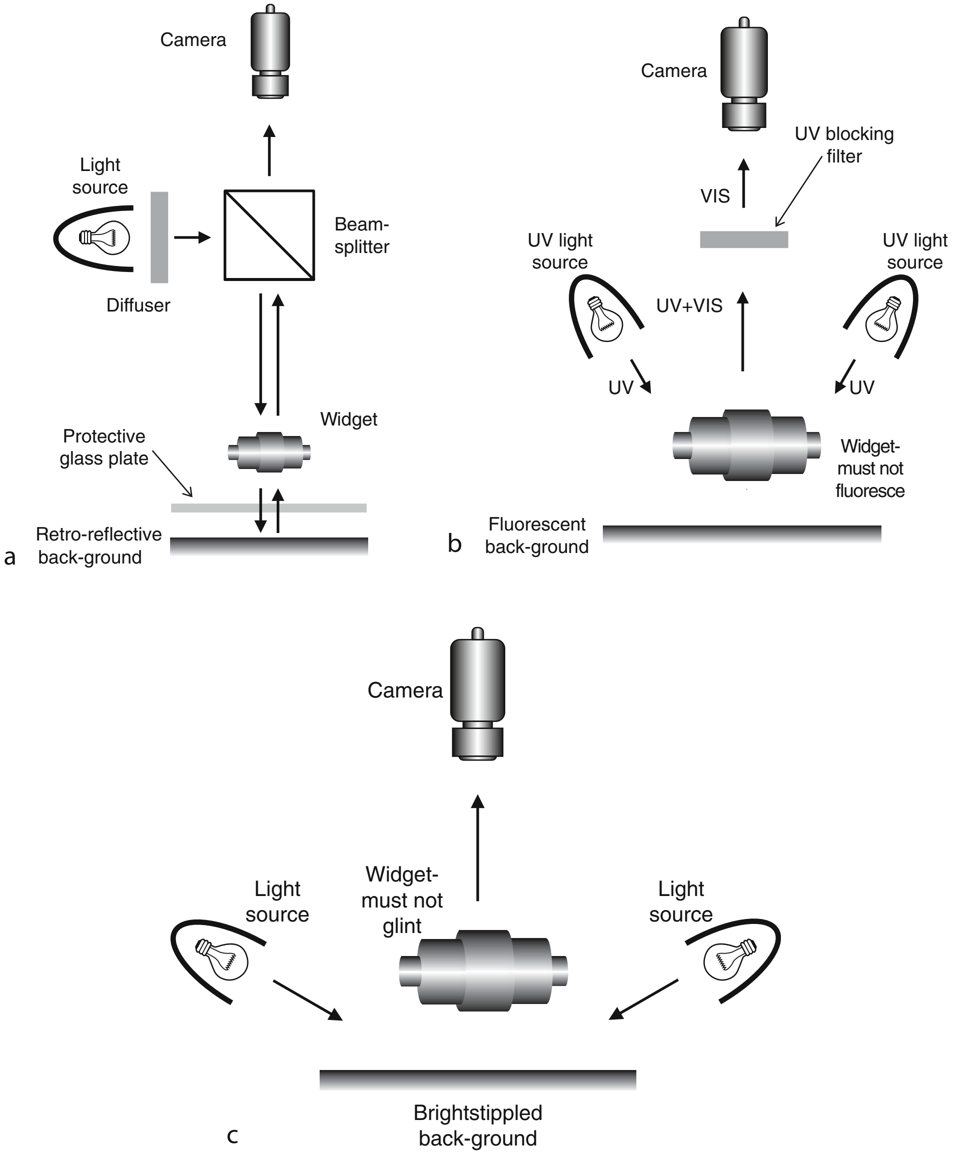


Fig. 40.96
(Continued)

40.102 Method 102: Split Field Lighting and Viewing

OBJECTIVES To obtain two or more views of an object, using a single camera. For example we may view the front and back of an simultaneously, or the ends of a long-thin object.
TYPICAL APPLICATIONS

- Inspecting complex objects

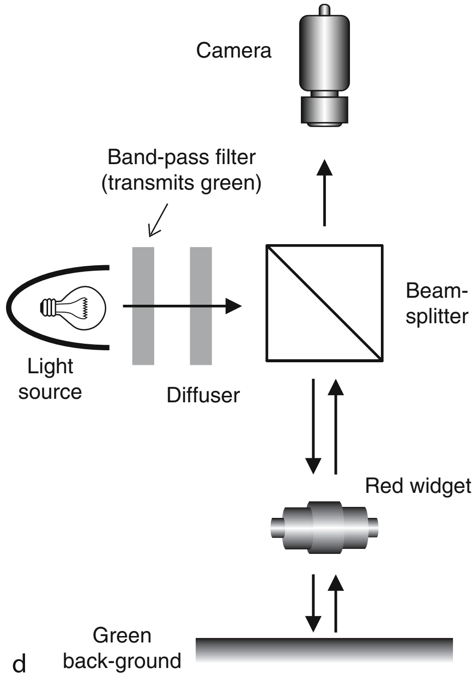


Fig. 40.96
Inspecting an object silhouette in a restricted space (a) Retro-reflective back-ground. (b) Fluoresent back-groud. (c) Stippled glinting back-ground with low-angle illumination. (d) Coloured illumination and background

- Inspecting long thin objects
- Robot vision (viewing front and back of an object before picking it up)

EQUIPMENT

- Front-silvered (first-surface) plane mirrors. (Prisms could be used instead.)
- Colour filters
- Beam splitter

CAMERA Array.

MOTION Static.

REMARKS Mirrors provide a low-cost method of splitting the camera's field of view. Using the optical arrangement in *Fig. 40.98*, the field of view is split into two. However, illumination can be difficult, since a lighting arrangement that optimises image quality for one half-field may reduce the contrast in the other. Two possible illumination methods are shown *Fig. 40.98*, while omni-directional lighting offers another option. Since video cameras are cheap nowadays, it may be preferable to use two separate cameras instead of the split-field method illustrated shown here. The video images can be combined electronically, or in software.

REFERENCE Adjustable field splitter, multi-arm articulated periscope. Light Works, LLC, Toledo. US Patent 5,975,710

LAYOUT DIAGRAM *Figure 40.98*

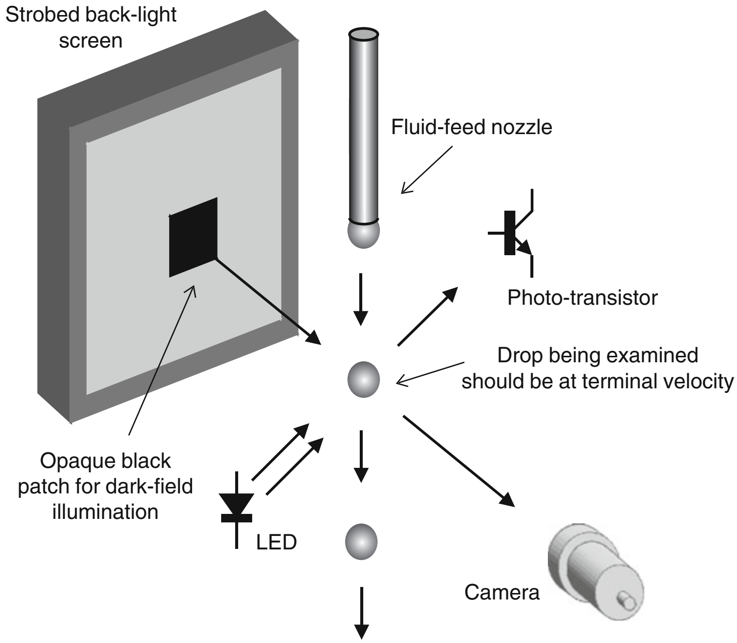

Strobed back-light screen
Fluid-feed nozzle
Photo-transistor
Drop being examined should be at terminal velocity
Opaque black patch for dark-field illumination
LED
Camera

Fig. 40.97
Using strobed lighting to capture an image from a drop in free fall

40.103 Method 103: Stabilised Lighting

OBJECTIVE To provide high-stability lighting that remains nearly constant despite changes in lamp efficiency. Changes in brightness may occur as a result of ageing, power-line voltage fluctuations, and tarnishing/contamination of optical surfaces.

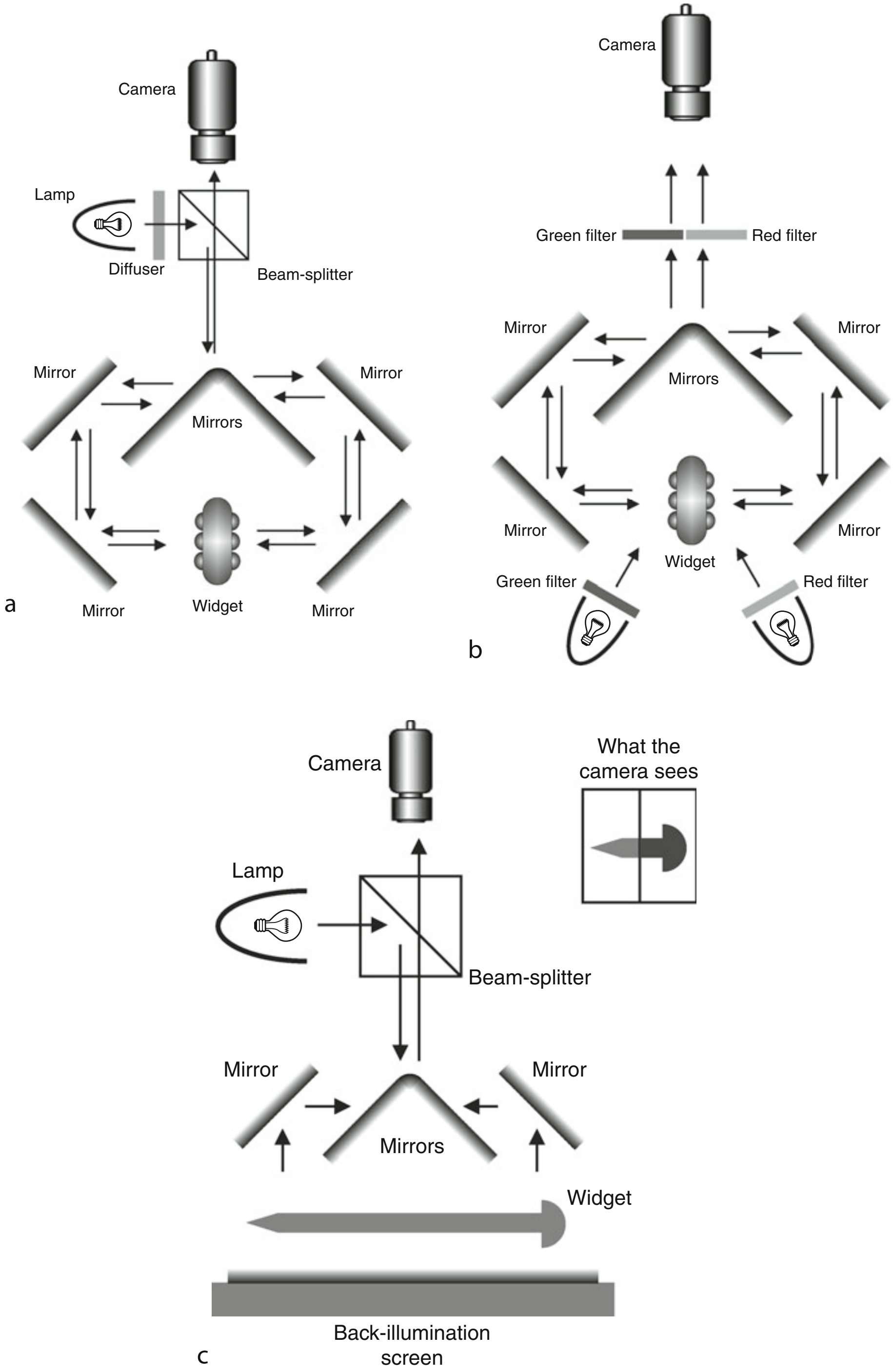
TYPICAL APPLICATIONS

- Nearly all applications benefit from the use of well-controlled lighting, since it is possible then to make the best possible use of the camera's limited dynamic range
- High-resolution print and web inspection, where simple thresholding is used at the beginning of the processing cycle, to reduce the image to binary form and thereby achieve the highest possible throughput rate
- Measuring the sizes of opaque objects using simple thresholding and pixel counting. (This is a standard arrangement for simple, low-cost vision systems.)

EQUIPMENT This depends upon which of the methods described below is used.

- Photo-sensor, capable of measuring light levels accurately
- Feedback control amplifier
- Voltage controlled power amplifier
- Filament lamps, or LED array

REMARKS Many proprietary lighting modules now incorporate some form of feed-back control. Normally, the output light level is measured just before the luminaire aperture. (Method A in *Fig. 40.99*) However, this does not directly control the light level received by the camera, which is the critical parameter. Hence, it is usually better to monitor the light levels



◘ Fig. 40.98
(Continued)

falling on the widget or its back-ground. Several methods for controlling the light level are possible:

1. Sensor is placed close to the light source, possibly within the luminaire, and measures the light output from the lamps
2. Sensor is placed close to the background and measures light scattered by it
3. A sensor is embedded within the back-ground and measures light received directly from the lamps
4. Sensor observes the back-ground from afar and measures light scattered by it
5. Feed-back signal is derived from the optical image received by the camera, which may be viewing either of the following:
   (a) A standard calibration pattern, occasionally placed in the camera's field of view
   (b) A calibration pattern permanently attached to the back-ground. This might, for example, be a simple "chess-board" target

These all have different merits. The final choice about which method is best for a given application depends upon the level of contamination (dust and fumes) that exists around the inspection area, the precision expected of the inspection process and the availability of staff to perform regular calibration checks. 5(b) is closest to the ideal calibration method, since it is available continuously and monitors what is most important: the light reaching the camera. Standard incandescent lamps change colour when the filament temperature varies. LED brightness can be varied using pulse-width modulation but this requires careful synchronisation with the camera, to avoid producing hum bars. Discharge and fluorescent lamps are rather difficult to control.

LAYOUT DIAGRAM ➋ *Figure 40.99*

40.104 Method 104: Steerable Image Sensor

OBJECTIVES

- To perform rapid pan-and-tilt sampling of a large/complex scene
- To observe a moving widget without introducing significant motion blur
- To view a static widget using a stationary line-scan sensor

TYPICAL APPLICATIONS

- Situations where a single expensive camera (e.g., low-light, IR or UV sensor) must be used for more than one function
- Surveillance, using a single static camera to monitor activity within a large scene
- Tracking individual widgets as they are transported along a conveyor line
- Motion analysis

Fig. 40.98

Viewing the front and back of an object at the same time. (a) Coaxial illumination and viewing. The right-hand side of the optical system provides back-illumination for the widget as it appears in the left half-field and front-illumination for the right-hand field. This method is only effective if the reflectivity of the widget is high. (b) Colour filters stop this back-illumination effect. This method is not effective for vividly coloured objects; the widget should have neutral shading (i.e., grey). (c) Viewing the ends of a long thin object

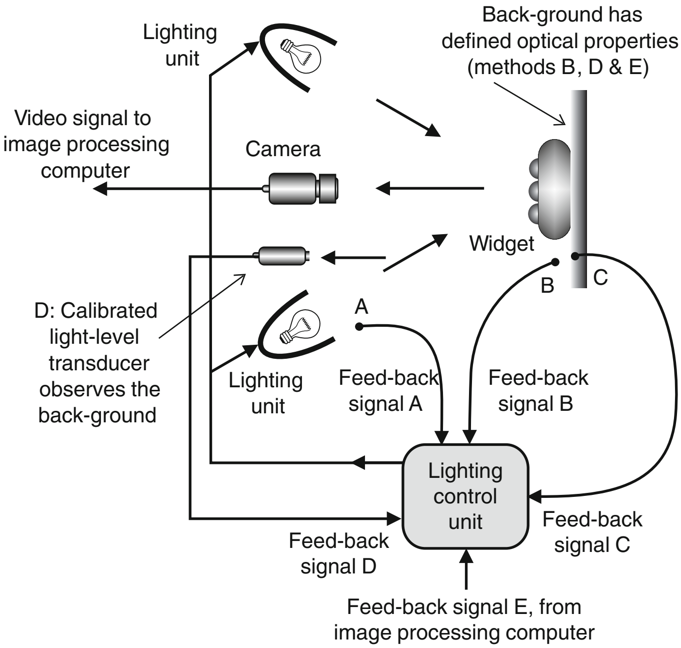

Lighting unit
Back-ground has defined optical properties (methods B, D & E)
Video signal to image processing computer
Camera
Widget
D: Calibrated light-level transducer observes the back-ground
A
B
C
Lighting unit
Feed-back signal A
Feed-back signal B
Lighting control unit
Feed-back signal C
Feed-back signal D
Feed-back signal E, from image processing computer

Fig. 40.99
Methods for monitoring and adjusting lamp brightness

- Eliminating motion blur
- Monitoring the motion of a robotic vehicle
- Monitoring a long thin scene (e.g., checking a walk-way for debris)
- Warehousing: checking rows of items on shelves

EQUIPMENT One or two small first-surface mirrors mounted on galvanometers.

CAMERA Array or line-scan, depending on the application.

MOTION Static or continuous linear motion, depending on the application.

REMARKS A galvanometer mechanism, fitted with a small first-surface mirror, provides a convenient facility for “random access” viewing of large and/or complex scenes. By effectively panning the camera at the same speed as a moving object, it is also possible to “freeze” its movement, thereby eliminating motion blur without resorting to stroboscopic illumination. For example, in one project, an IR camera was used to inspect red-hot glass-ware on a continuous-motion conveyor. Without the scanning mirror, the images would have been severely blurred. Two types of galvanometer are available. One kind allows the mirror to be *positioned* precisely. (➋ *Fig. 40.100a*) On the other hand, by using a resonant galvanometer, the mirror *motion* can be predicted accurately. (➋ *Fig. 40.100b*) While a modern light-weight camera could be mounted on a scanning mechanism, its connecting lead would be flexed continuously, making operation unreliable. A wire-less power and video link would, of course, circumvent this problem but would increase the load on the galvanometer and reduce its resonant frequency.

REFERENCES Galvanometer scanners, GSI, http://www.gsig.com/. Accessed 6 Sep 2011

LAYOUT DIAGRAMS ➋ *Figure 40.100*

40.105 Method 105: Steering a Laser Beam in Two Directions

OBJECTIVES To establish the 3D structure of a complex scene, using a software-controlled, 2-axis flying-spot scanner.
TYPICAL APPLICATIONS Active stereo for robot vision, or automated vehicle guidance.
EQUIPMENT Laser, plus two galvanometer movements mounted with small mirrors. The latter are driven, via digital-to-analogue converters, from the image processing computer. The laser can also be switched on/off under software control. Standard lighting is also used.
CAMERA One or two array-scan cameras. If high-speed operation is required, two line-scan cameras may be used to locate the laser spot.
MOTION Static.
REMARKS The laser-deflection system is explained in ➜ *Fig. 40.101a* and is intended to be used in a triangulation arrangement, such as that shown in ➜ *Fig. 40.101b*. The laser beam can be steered independently along the x- and y-axes. We could, for example, scan the scene, creating a coarse raster scan, jumping about at random, or following a contour derived from the visual image(s). Since only one range measurement can be obtained during a video field period, this method is far too slow for generating a full-resolution depth map. This remark applies specifically to a system based on array-scan cameras. Specialist analogue, spot-position sensors are available and can greatly increase the speed. However, this does require a more complicated optical set-up. Another fast arrangement for spot location is shown in ➜ *Fig. 40.101c* and uses two line-scan sensors. If a separate range-measurement sensor is used, we must ensure that the camera(s) and spot locator are properly aligned. The rangefinder is typically used to interrogate the visual scene at a relatively small number of points chosen by analysing the image. First, we first switch the laser off and the main lights on. This allows us to obtain a standard visual image (or two, if two cameras are used), which is then processed and a few key points identified. (These might, for example, be points where the

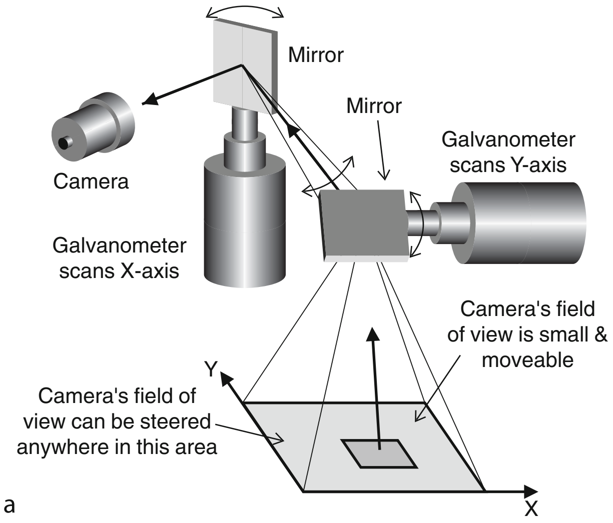
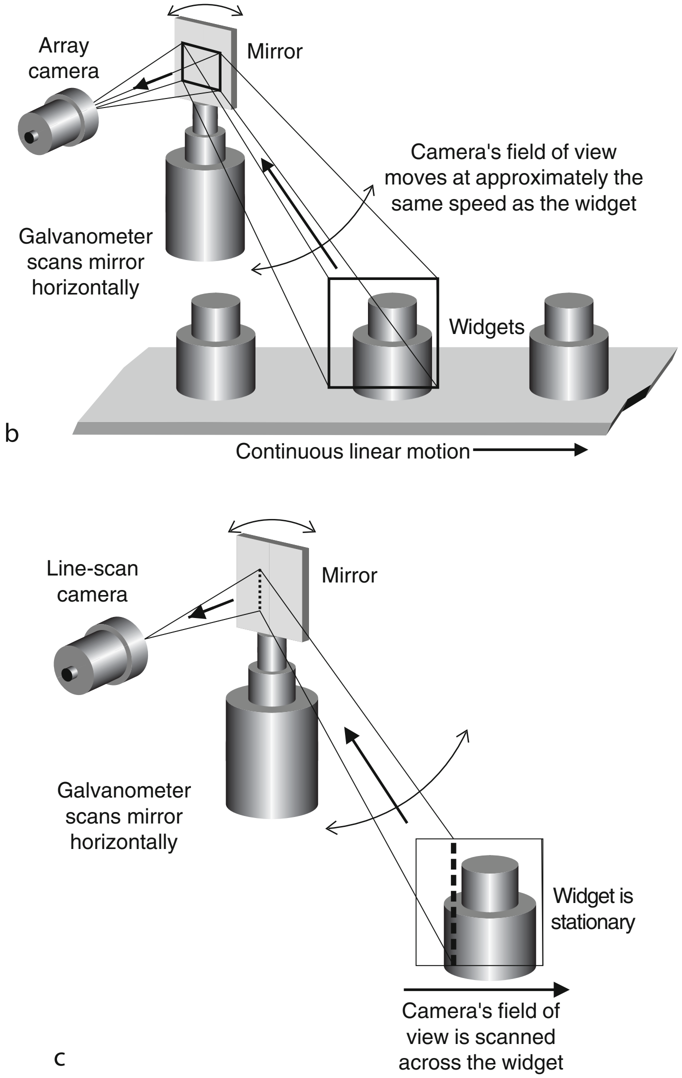

Mirror
Mirror
Galvanometer scans Y-axis
Camera
Galvanometer scans X-axis
Camera's field of view is small & moveable
Y
Camera's field of view can be steered anywhere in this area
a
X

Fig. 40.100
(Continued)


Array camera
Mirror
Camera's field of view moves at approximately the same speed as the widget
Galvanometer scans mirror horizontally
Widgets
b
Continuous linear motion
Line-scan camera
Mirror
Galvanometer scans mirror horizontally
Widget is stationary
Camera's field of view is scanned across the widget
c

◘ Fig. 40.100
Steerable image sensor. (a) Two galvanometer-mirror devices provide an array camera with pan-and-tilt motion. Non-resonant galvanometers are used here. (b) Freezing motion. Resonant galvanometers can be used here. (c) Using a line-scan camera and a resonant galvanometer to view a stationary object

magnitude of the intensity gradient is high.) We then switch the main lights off and the laser on. We then measure the range at these key points, to distinguish between foreground and background surfaces. To do this, we might apply an edge-detection operator to the visual image, to identify a contour of high intensity gradient. We can then use the range finder to establish whether this contour corresponds to a physical edge; we could, for example, distinguish between the leg of a table and its shadow on the wall behind it.

REFERENCES Marshall AD, Martin RR (1992) Computer vision, models and inspection. World Scientific, Singapore. ISBN 9810207727

Galvanometer scanners, GSI, http://www.gsig.com/. Accessed 6 Sep 2011

LAYOUT DIAGRAM ➋ *Figure 40.101*

40.106 Method 106: Stroboscopic Illumination

OBJECTIVE To "freeze" fast continuous motion, using a very short flash of light.

TYPICAL APPLICATIONS Inspecting continuously moving

- Machinery (The motion may be linear or rotary.)
- Discrete parts on a conveyor, slide-way or in free-fall
- Webs (e.g., printing, fabric, sheet materials), extrusions, wire, cable, tubing, ribbon, etc.

EQUIPMENT Strobed light source, synchronised to the camera. A discharge tube or LED array may be used. The strobe does not have to flash for every video frame but it must be triggered at a fixed point in the video cycle. Screening is particularly important to avoid stray light from reaching people nearby. Fibre optics is useful for directing light from a large/remote source onto the surface, whilst eliminating stray light.

CAMERA Array-scan. A progressive scan camera is useful, as it can be triggered by the strobe or an external signal.

MOTION Fast continuous linear, or rotary motion.

REMARKS Stroboscopic lighting is potentially dangerous as it can trigger migraine and epileptic fits. It can also make rotating machinery appear to be stationary. It is therefore imperative that all stray light be eliminated. Stroboscopic lighting, with viewing by eye, has

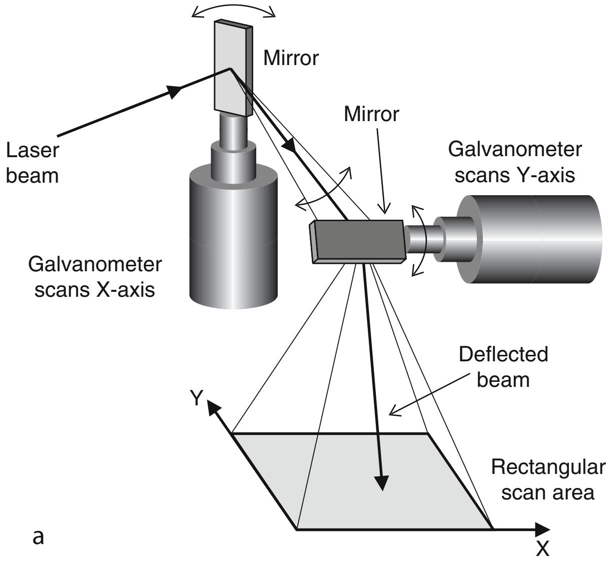
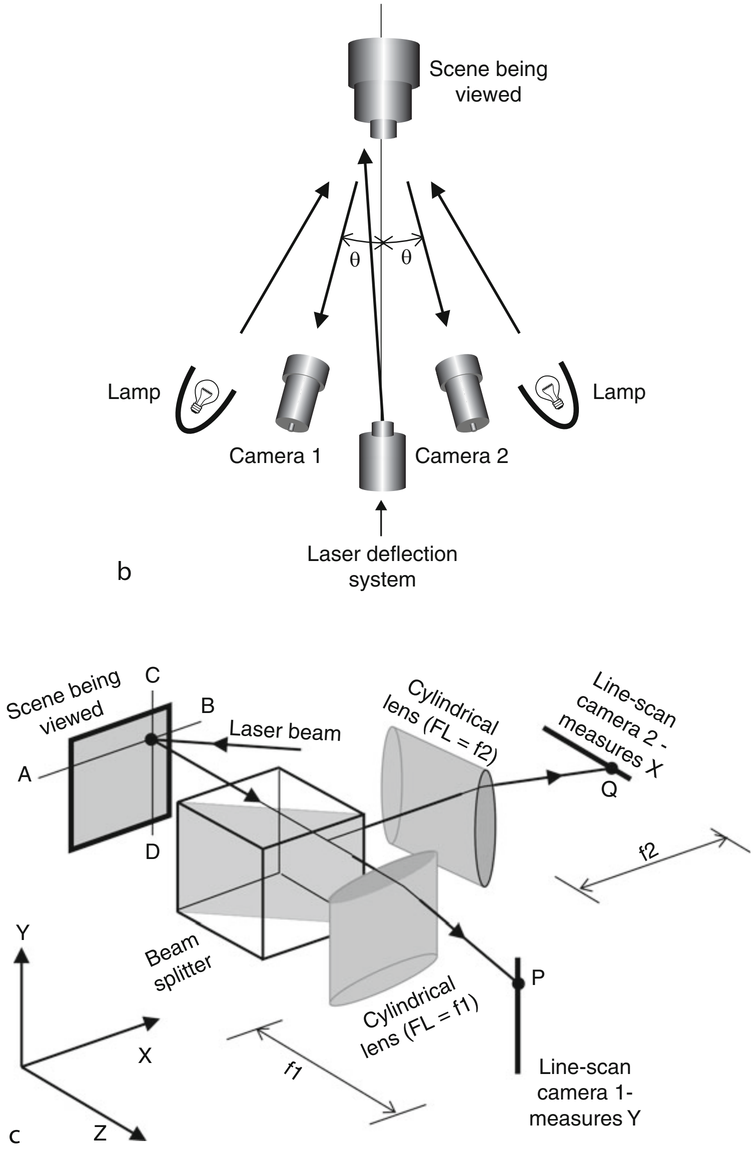


Fig. 40.101
(Continued)



◘ Fig. 40.101

Twin-axis laser flying-spot scanner. (a) Beam-steering sub-system. The galvanometers are from a computer. (b) 2-camera triangulation scheme. Using two cameras reduces the effects of occlusion due to parallax and enables stereo image pairs to be correlated. (c) Using two line-scan cameras to locate a laser spot. All points along the line AB are focussed onto a single point, P, on camera 1. Similarly, all points along the line CD are focussed onto a single point, Q, on camera 2. The main lights are assumed to be off, or at least, very much darker than the laser spot

long been used for inspecting rotating machinery and continuously moving webs that have repeated patterns (e.g., printing). By providing image storage, it is also possible to examine non-periodic, transient events, provided that there is some signal to herald the event of interest. The use of stroboscopic illumination for Machine Vision requires careful consideration of timing; the camera, strobe unit and image processor must all be synchronised to external (factory) machines. The following analysis relates to a non-interlaced array-scan camera synchronised to the strobe.

Web speed:	S metres/second
Video frame rate:	F frames/second
Strobe rate:	F/N pulses/second, where $N \in [1,2,3,\ldots]$
Strobe intensity:	I photons/second *(unusual units to simplify calc.)*
Flash duration:	T seconds (Ensure that $T < < 1/F$)
Surface reflectance:	R $(0 \leq R \leq 1)$
Web movement during scan:	S/F metres
Web movement during flash:	S.T metres
Integration time:	Q/F seconds, where $0 \leq Q < 1$. (Acknowledges that the integration time is less than the clock period. Normally, $Q/F. >> T$).
Light energy at camera:	$R.I.T$ photons/frame. *(Must be large compared to swamp noise but avoid saturating the camera.)*
Minimum pixel size:	S.T metres (along the direction of travel)
Smallest detectable spot:	2.S.T metres
Maximum image resolution:	1/(F.T) pixels (along the direction of travel)

The light source(s) may be placed anywhere, as most effective. For example, we may use paraxial, back, dark-field or grazing illumination. Multiple light sources must be synchronised to each other. A proximity sensor (e.g., LED-phototransistor) may be useful for detecting the arrival of a widget in front of the camera. Webs may be made with special features (e.g., fiducial marks, punched holes) on the edge, to trigger the strobe.

40.107 Method 107: Structured Lighting (Triangulation)

OBJECTIVE To obtain 3-dimensional shape information, generating a *depth map* (also called a *range map*).

TYPICAL APPLICATIONS Measuring machine parts, plastic mouldings, die castings, extrusions, food products (e.g., cakes and loaves), piles of granular material, etc.

EQUIPMENT Laser fitted with cylindrical, or holographic, lens to generate a fan-shaped beam. Linear or rotational transport mechanism.

CAMERA Array.

LENS Analysis given below applies to a narrow-angle of view. Wide-angle lens requires separate calibration procedure.

MOTION Linear or rotary, slow continuous, or indexed.

GEOMETRY Moderate complexity. Occlusion occurs with complex shapes.

MATERIALS Opaque, matt, non-reflective, neither glinting nor very dark.

REMARKS Three methods are described below.

1. **Inferior system** The notation is explained in *Fig. 40.102a*
   (a) The height of the widget surface above the reference plane, at the point where the laser beam impinges upon it, is H
   (b) The camera is placed at height L above the reference plane
   (c) The camera photo-detector spacing is d
   (d) The bright spot projected onto the photo-detector array is off-set by N rows
   (e) The lens has a minification (inverse of magnification) equal to M at a working distance equal to L
   (f) The central axis of the laser beam is inclined at an angle A relative to the vertical. (Usually A is about 45°.)

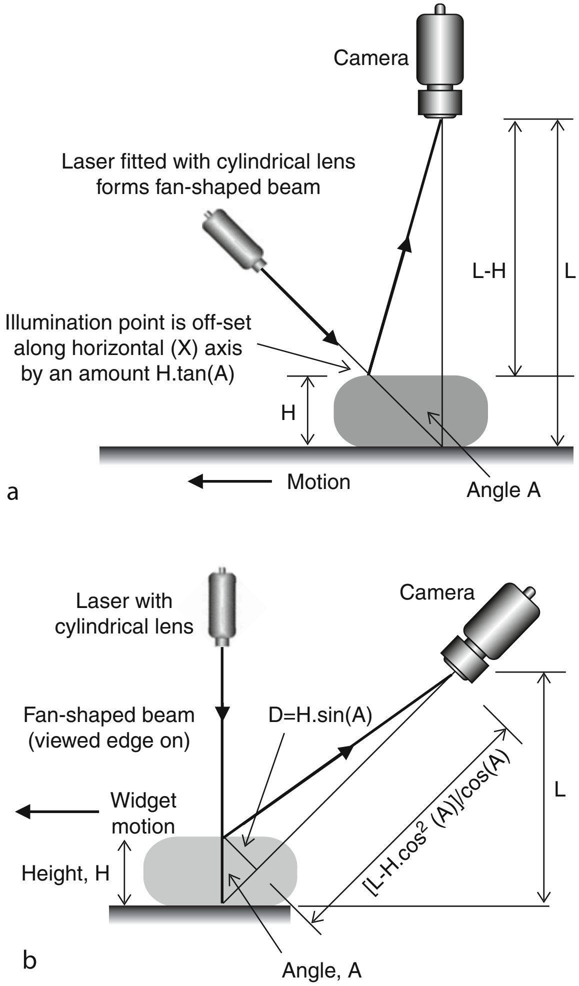
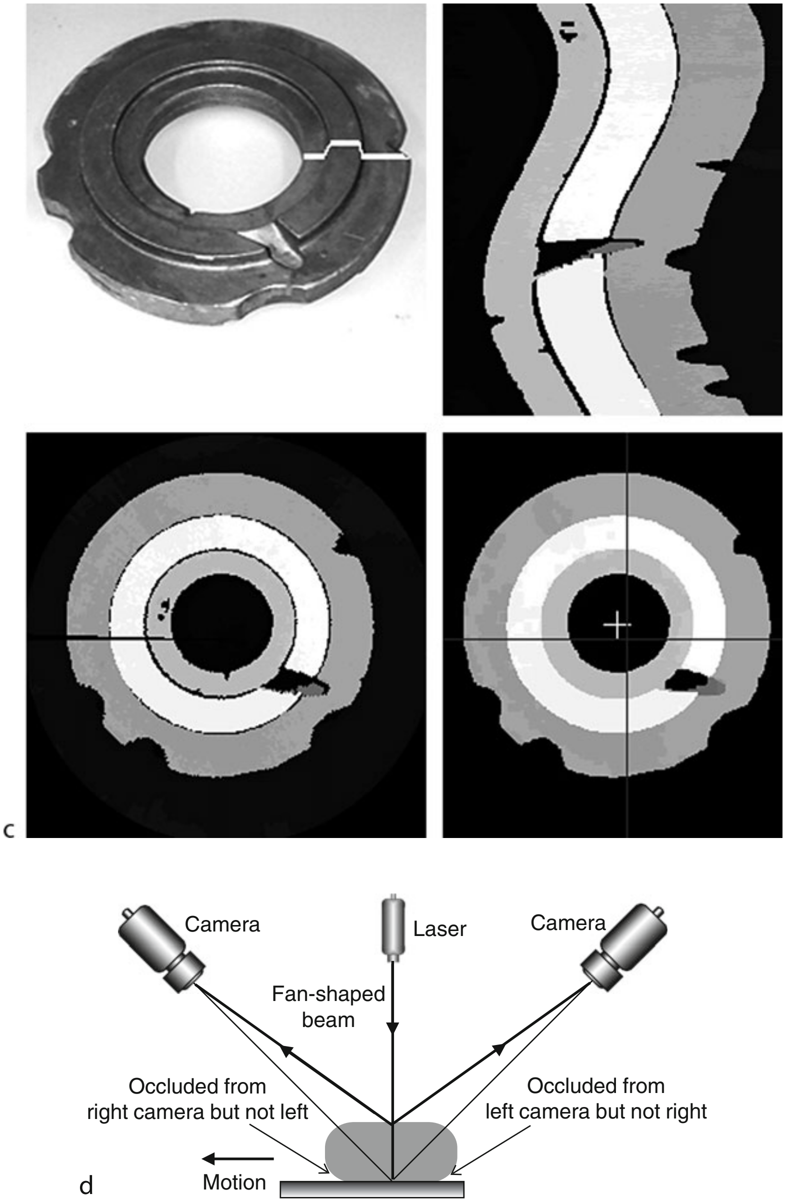


Fig. 40.102
(Continued)


c
Camera
Laser
Camera
Fan-shaped beam
Occluded from right camera but not left
Occluded from left camera but not right
Motion
d

▪ Fig. 40.102 (Continued)

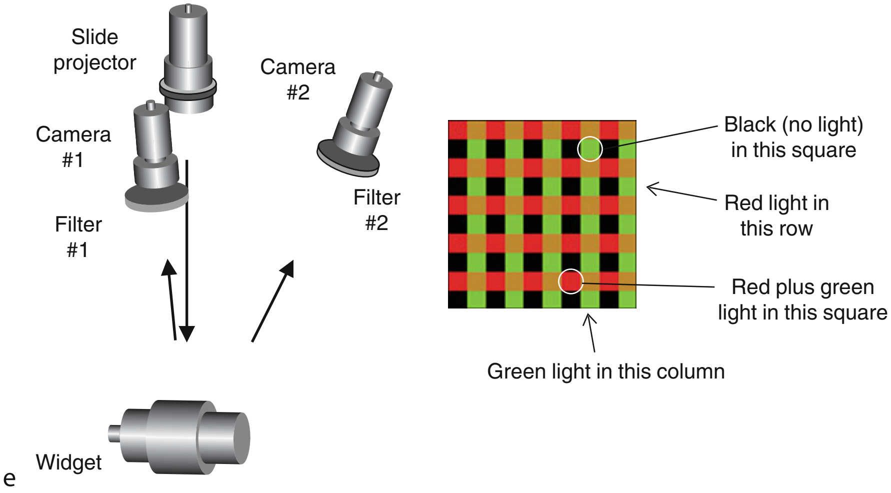

Slide projector
Camera #2
Camera #1
Filter #1
Filter #2
Widget
e
Black (no light) in this square
Red light in this row
Red plus green light in this square
Green light in this column

Fig. 40.102
Generating a range map by triangulation, using structured lighting (a) Inferior method. (b) Superior method. (c) Sample images: Grey toroidal alloy component, scanned on a turn-table. Diameter 200 mm. *Top-left:* Oblique view in natural illumination, with light stripe superimposed *Top-right:* Depth map; intensity is proportional to surface height. Edges have a sinusoidal form because the component was not centred on the turn-table. Black regions indicate areas of occlusion. *Bottom-left:* Cartesian-to-polar co-ordinate transformation. *Bottom-right:* After noise reduction. Centre of the image (black cross, corresponds to the centre of the turntable) and centroid of the hole in the middle (white cross) (d) Using two cameras to reduce the effects of occlusion. (e) Using orthogonal cameras to reduce the effects of occlusion. This set-up and that shown in (d) can be combined to reduce occlusion still further. (f) Illumination pattern for (e). While red-green illumination is shown here for illustrative purposes, other pairs of easily-distinguished wavelengths might be used, if the widget does not scatter both red and green light

The height of the widget surface at the point of illumination (H) is given by

$$H = L.N.d.M/[L.\tan(A) + N.d.M)]$$

If L.tan(A) >> N.d.M, then

$$H \approx N.d.M.\cot(A).$$

Notice however, that the object surface is illuminated at a point located a distance H.tan(A) from the camera axis. Although we have located a point on the object surface, the data is difficult to use. We cannot, for example, easily construct a range map. If we ignore the off-set along the horizontal axis, objects in the depth map appear to lean over. The layout diagram shown in *Fig. 40.102a* is often shown in books and papers on Machine Vision, even though this technique is usually not so useful as that described next.

2. **Superior system** This time, the camera's optical axis is inclined at an angle A relative to the vertical. (Usually about 45°. See ➜ *Fig. 40.102b*.) Then:

$$H = N.L.d.M/[L.\sin(A) + N.d.M.\cos^2(A)]$$

   H is a ratio of two linear functions of N, which is the parameter measured experimentally. All other terms in this equation are constant. If L >> [N.D.M], H reduces to a linear function of N:

$$H \approx N.d.M.\mathrm{cosec}(A).$$

   This assumes that the lens does not have a wide-angle view. Since the laser beam is vertical, the illuminated point on the widget is always in the same horizontal position, whatever the value of H. This is not the case for the inferior method described above, which has been commonly shown in books and papers on Machine Vision.
3. **Using two cameras to avoid occlusion** (➜ *Fig. 40.102d*) This is essentially the same as that just described, except that two cameras are used, instead of one. This allows a greater part of the object surface to be mapped, as it is less susceptible to occlusion. To merge the images from the two cameras, we use the following algorithm. Let L(i,j) denote the (i,j)th pixel in the image from the left-hand camera and R(i,j) be the (i,j)th pixel in the image from the right-hand camera. Then M(i,j), the (i,j)th pixel in the merged output image, is given by

$$M(i,j) = \begin{cases} L(i,j), \text{if } R(i,j) = 0 \text{ (occluded from right camera) and } L(i,j) > 0 \\ R(i,j), \text{if } L(i,j) = 0 \text{ (occluded from left camera) and } R(i,j) > 0 \\ [L(i,j) + R(i,j)]/2, \text{if } L(i,j) > 0 \text{ and } R(i,j) > 0 \\ 0, \text{ otherwise.} \end{cases}$$

REFERENCES Cotter SM (1985) Machine vision. PhD thesis, UWIST, Cardiff, Wales, UK

LAYOUT DIAGRAMS AND SAMPLE IMAGES ➜ *Figure 40.102*

40.108 Method 108: Switched-Colour Light Source

OBJECTIVE To obtain images from a number of different spectral bands in the near-UV, VIS and near-IR.

TYPICAL APPLICATIONS

- Inspecting printing, labels, packaging, etc.
- Inspecting woven textiles and printed fabrics
- Reading resistor colour codes
- Inspecting holograms

EQUIPMENT Switched LED modules (➜ *Fig. 40.103b*). Monochrome camera whose spectral response has previously been calibrated.

REMARKS Obtaining multi-spectral data is made much easier through the use of switched LED sources. Several images are digitised in quick succession, using closely-spaced LEDs, operating at different wave-lengths. 3-colour LED modules are available commercially. These usually contain red, green and blue LEDs, although other combinations are also available. It is possible to span the entire VIS spectrum (0.4–0.7 μm) with LEDs whose wave-lengths vary in increments of 0.01 μm. The emission spectrum from an individual LED is narrow,

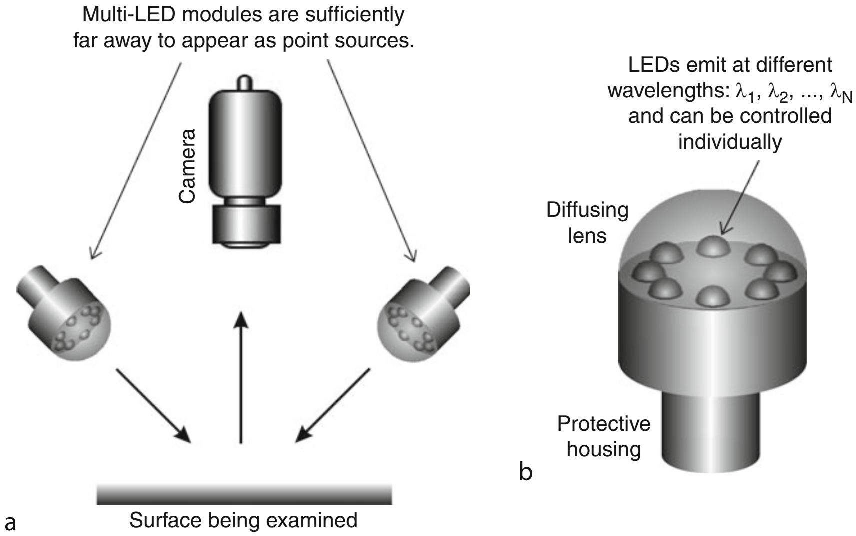

Multi-LED modules are sufficiently far away to appear as point sources.
Camera
Surface being examined
a
LEDs emit at different wavelengths: $\lambda_1, \lambda_2, ..., \lambda_N$ and can be controlled individually
Diffusing lens
Protective housing
b

Fig. 40.103
Multi-LED light sources used as low-intensity lamps. (a) Optical layout. (b) Construction of the multi-LED modules

approximately monochromatic. Hence, we can examine a narrow spectral band in detail (e.g., up to ten samples within a wave-length range of 0.1 µm), or obtain multiple samples (up to about 30), spaced uniformly across the VIS spectrum. (Recall that conventional colour cameras analyse only three channels over the range 0.4–0.7 µm.) This provides a low-cost, versatile and relatively precise method for analysing colours of small objects, including natural products, food-stuffs and food-products, where subtle colour changes occur.
DIAGRAMS ➲ *Figure 40.103*

40.109 Method 109: Telecentric Lens

OBJECTIVE To view objects as if they were at an infinite distance from the camera, thereby eliminating parallax distortion.
TYPICAL APPLICATIONS Metrology. A number of other lighting-viewing methods rely on the use of telecentric lenses.
EQUIPMENT Telecentric imaging system, consisting of a large lens (possibly Fresnel type, if a large objective element is required) and a smaller standard lens. Pinhole aperture.
REMARKS The magnification of a telecentric imaging system is determined by the ratio of the focal lengths of the individual elements: f_2/f_1. The optimal image quality is obtained with a very small pin-hole aperture, although this introduces a high optical loss. In this case, the widget must be brightly illuminated. Notice that the objective element must be at least as large as the widget. Telecentric lenses of diameter up to 475 mm are available commercially. To obtain an even larger field of view (up to 2 m diameter), plastic Fresnel lenses may be used. (It may be

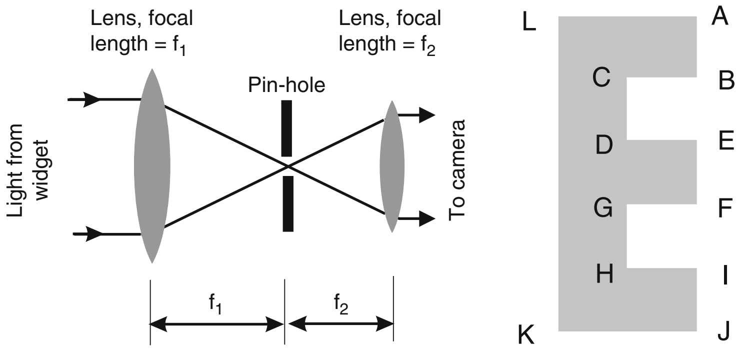


Fig. 40.104
Telecentric imaging. (a) Optical layout. The system has a magnification of f_2/f_1. A large aperture increases the image brightness but allows rays that are not truly parallel to the optical axis to pass through, resulting in a system producing images of degraded quality. (b) If an opaque object with this geometry were viewed from the right using a standard lens. K, L, D and G would be obscured. If the widget were transparent, the features would be aligned in order A, L, B, C, E, D, G, F, H, K, J. However, if a telecentric lens is used, points are aligned in pairs as follows: L with A, C with B, D with E, G with F, H with I, and K with J

necessary to use monochromatic light, to eliminate “rainbows” around high-contrast edges,.) Fresnel lenses (30 cm diameter) intended for overhead projectors can be used. to produce a low-cost telecentric system. Cylindrical-telecentric lenses are also available for line-scan cameras.

REFERENCES

- *Visionmes*®, Telecentric Lens Systems, Carl Zeiss Jena GmbH, Jena, http://www.zeiss.de/en. Accessed 6 Sep 2011
- *Telecentric lens for linear array cameras.* Light Works, Patent 5,975,710. http://www.lw4u.com/. Accessed Sep 2011

LAYOUT DIAGRAM ➲ *Figure 40.104*

40.110 Method 110: Using Mirrors to View Front and Back

OBJECTIVE To view the front and two side-rear faces of a widget.
TYPICAL APPLICATIONS This technique was originally developed for inspecting corks, which were held in place by feeding them through a glass tube.
EQUIPMENT Plane mirrors.
CAMERA Array.
LENS Must be able to focus from the front of the direct view to the rear-most visible point on the virtual images.
MOTION Static.
LIGHTING Matt, non shiny objects are best inspected in this way as there is a limited number of options for lighting.
REMARKS This technique could, in theory, be extended to provide all round inspection of the widget, although it would not then be possible to move it past the camera. The advent of

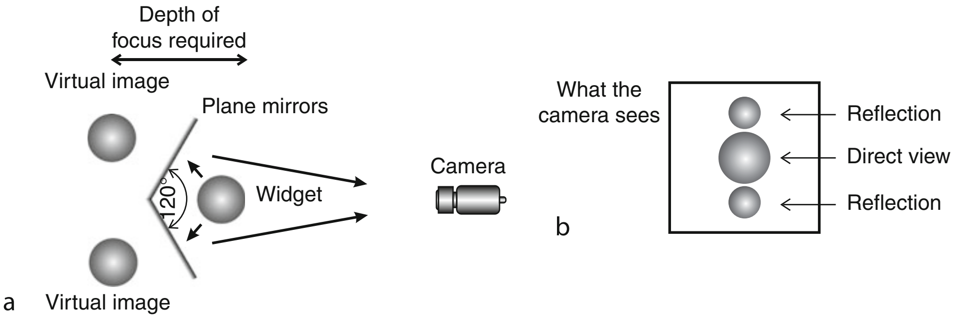


Fig. 40.105
Combining front and two side-view images of a widget using two plane mirrors. (a) Optical layout. (b) What the camera sees. Notice that the two reflected images are smaller than the direct view

low-cost cameras and lenses makes this method less useful than it once was. Notice the change of size of the direct and reflected views and the requirement for a large depth of focus.
LAYOUT DIAGRAM ➋ *Figure 40.105*

40.111 Method 111: Very Large Membrane Mirror

OBJECTIVE To provide a very large mirror, which may be part of an anamorphic or paraxial imaging system.
TYPICAL APPLICATIONS Viewing whole car body, manufacturing plant, robot work space, etc.
EQUIPMENT A large air-tight box is evacuated continuously by a vacuum pump. The front face of this box is perforated and is shaped to the desired form of the mirror. (Very high tolerance machining is not necessary.) A "silvered" (aluminised) Mylar film is stretched over the front face and takes up the shape of the front face. Minor variations in the shape of its reflecting surface are eliminated by the elasticity of the film.
LENS A standard-size wide angle lens is used in conjunction with very large mirror. The mirror optics should be adjusted to correct for geometric distortion introduced by the lens.
REMARKS Concave mirrors can be built most easily in this way; convex mirrors require more precise machining of (parts of) the front face of the vacuum box. Membrane mirrors as large as 4 × 4 m have been used in flight simulators. Membrane mirrors may be anamorphic. and a large paraxial imaging system can be built at moderate cost. The membrane mirror shape should be adjusted to correct for geometric distortion introduced by the wide-angle lens. The vacuum pump should be either a continuous flow type, or be buffered, to prevent oscillations in air pressure from affecting the shape of the mirror.
REFERENCES McKay S et al (1999) Projection displays V. Proc SPIE 3634:144–155
McKay S et al (1999) Stereoscopic displays and virtual reality systems VI. Proc SPIE 3639:122–131
LAYOUT DIAGRAM ➋ *Figure 40.106*

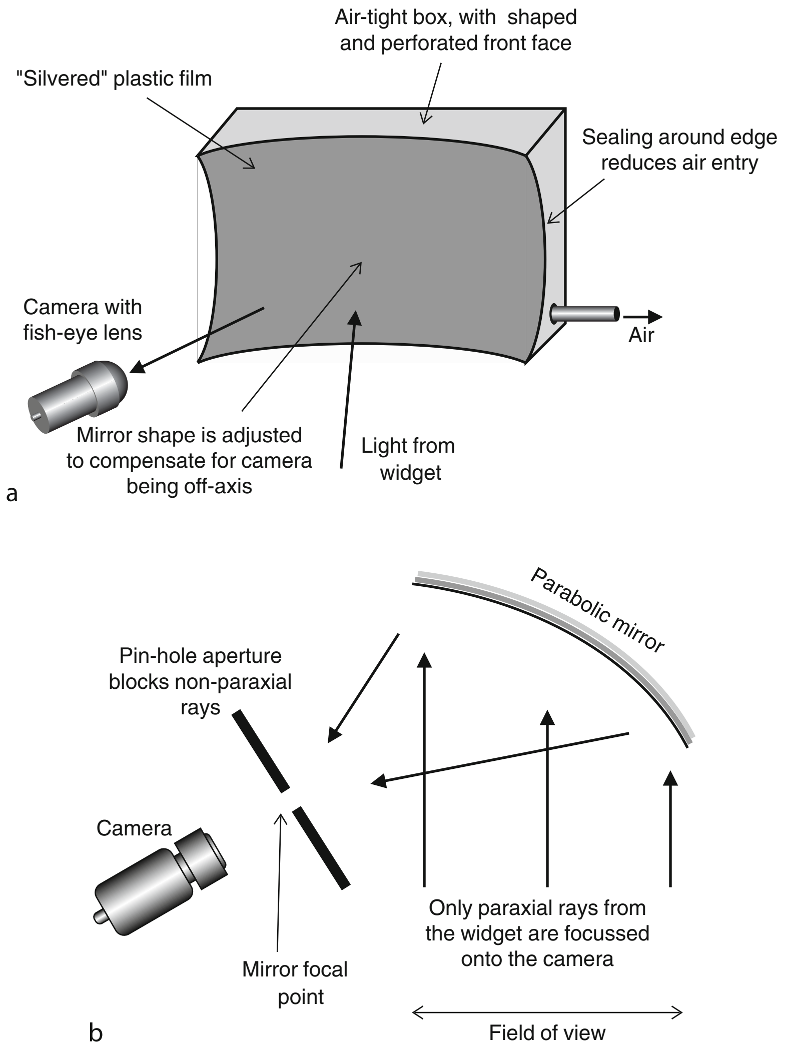

Air-tight box, with shaped and perforated front face
"Silvered" plastic film
Sealing around edge reduces air entry
Camera with fish-eye lens
Air
Mirror shape is adjusted to compensate for camera being off-axis
Light from widget
a
Parabolic mirror
Pin-hole aperture blocks non-paraxial rays
Camera
Mirror focal point
Only paraxial rays from the widget are focussed onto the camera
Field of view
b

Fig. 40.106
Membrane mirror. (a) Mirror construction (b) Using a large mirror of parabolic form to view paraxial rays only

40.112 Method 112: Vibration Analysis: Speckle Pattern Interferometry

OBJECTIVES To analyse vibration patterns in real time using electronic speckle pattern interferometry (ESPI).

TYPICAL APPLICATIONS Inspecting/testing loudspeakers, musical instruments (strings and percussion), panels covering vibrating machines (e.g., car, domestic white goods, pumps, etc.)

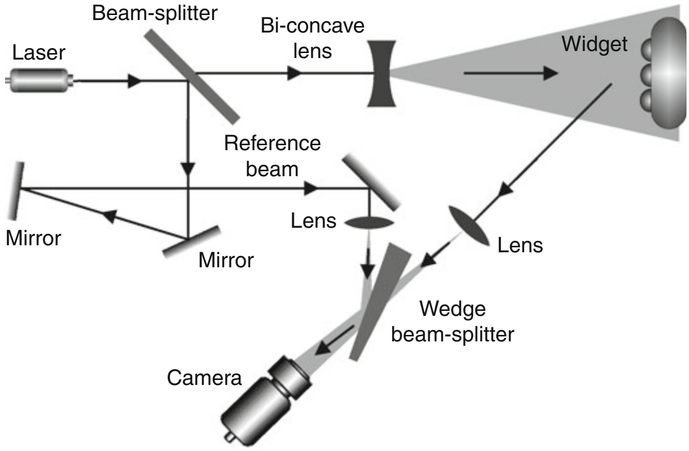


Fig. 40.107
Vibration analysis using speckle pattern interferometry

EQUIPMENT Pulsed laser, various small optical devices.
CAMERA Array.
MOTION Widget is stationary but is vibrating.
GEOMETRY Fairly smooth surfaces.
MATERIALS Low contrast, preferably coated in fine white powder. The spray-on developer used for crack detection is ideal.
REMARKS Notice that the object may be vibrating naturally. It is also possible to excite vibration in panels used in applications such as car bodies, using a loudspeaker mechanism, or piezo-electric device. ESPI is a highly specialised area and requires considerable expertise in optical engineering, for two reasons: (a) to maintain safe operation of high-power laser equipment (b) to keep optical equipment working reliably in a factory environment.
REFERENCES

- Hughes RG (1976) The determination of vibration patterns using a pulsed laser with holographic and electronic speckle pattern interferometry. In: Robinson ER (ed) Proceedings of the conference on the engineering uses of coherent optics, University of Strathclyde, April 1975. Cambridge University Press, Cambridge, pp 199–218. ISBN 0-521-20879-3
- Andrews IM, Leendertz JA (1976) Speckle pattern interferometry of vibration. IBM J Res Develop 20(3):285

LAYOUT DIAGRAM ➲ *Figure 40.107*

40.113 Method 113: View Annulus, Ignoring Central Disc

OBJECTIVE To view an annular region of a scene in high definition, while ignoring the central disc.
TYPICAL APPLICATIONS Inspecting circular objects, such as

- Tops of cans, jars and bottles

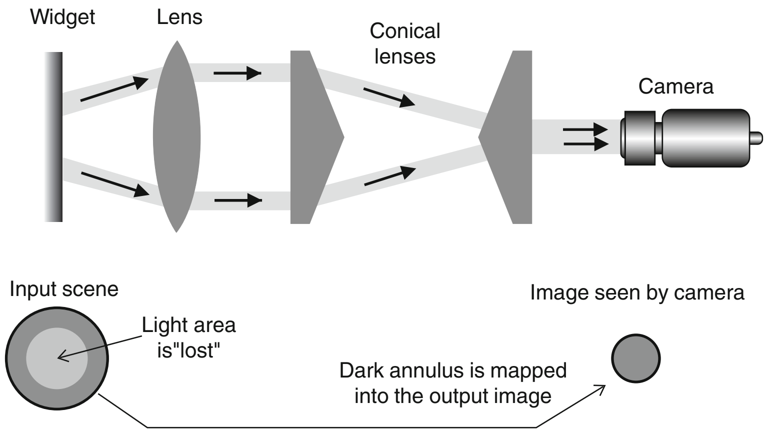


Fig. 40.108
View an annulus, ignoring the central disc

- Rims of holes
- Ends of rods, screws, etc. that are to be inserted into a hole

EQUIPMENT Two axicon lenses. (An axicon is a cone of transparent material.)
CAMERA Array.
MOTION Static.
GEOMETRY Circular, annulus.
REMARKS Using the optical arrangement shown in *Fig. 40.108*, it is possible to avoid storing and processing pixels that are not needed for the inspection process.
REFERENCES Optics for Research, Caldwell. http://www.ofr.com. Accessed 6 Sep 2011
LAYOUT DIAGRAM *Figure 40.108*

40.114 Method 114: View Flat Surface Without Glinting

OBJECTIVE To view a mirror-like surface that is very nearly flat, without observing reflections of the light source.
TYPICAL APPLICATIONS Inspecting glossy and mirror-like surfaces for stains, rust, etc. For example, glossy paper and card, DVDs, flat glass plates, smooth plastic sheet, ceramic tiles, magnetic tape, automobile and refrigerator body panels, etc.
EQUIPMENT Two, four or more flood lamps, or a ring-light (fibre-optic, LED array or circular fluorescent tube).
CAMERA Array or line-scan.
MOTION Static (array camera) or moving (line-scan).
GEOMETRY Nominally flat.
MATERIALS Smooth plastic, glass, ceramics, metal, etc. Lacquered/painted surfaces.
REMARKS This technique is closely related to the popular method used by photographers when recording images on glossy paper or card. Light from each of the lamps is reflected in a direction well away from the camera, which therefore receives only light that has been scattered by diffuse reflection and specular reflections from scratches, embossing, pimples,

pits, etc. A ring-light may be used to provide illumination through 360°, around the widget. Commercially available low-angle ring lights are sometimes described as "dark field lights."
DIAGRAMS AND SAMPLE IMAGE ➋ *Figure 40.109*

40.115 Method 115: View Orthogonally Polarised Images

OBJECTIVE To separate the horizontal and vertical polarisation components of an optical beam, so that they can be viewed simultaneously, using different cameras.
TYPICAL APPLICATIONS Inspecting.

- Surfaces coated with paint, lacquer, varnish, etc.
- Plastics, ceramics, glass, perspex (plexiglass), rubber and other non-metallic materials that have a smooth shiny surface
- Shiny, wet/oily objects
- Analysing stress in glassware and clear plastic materials

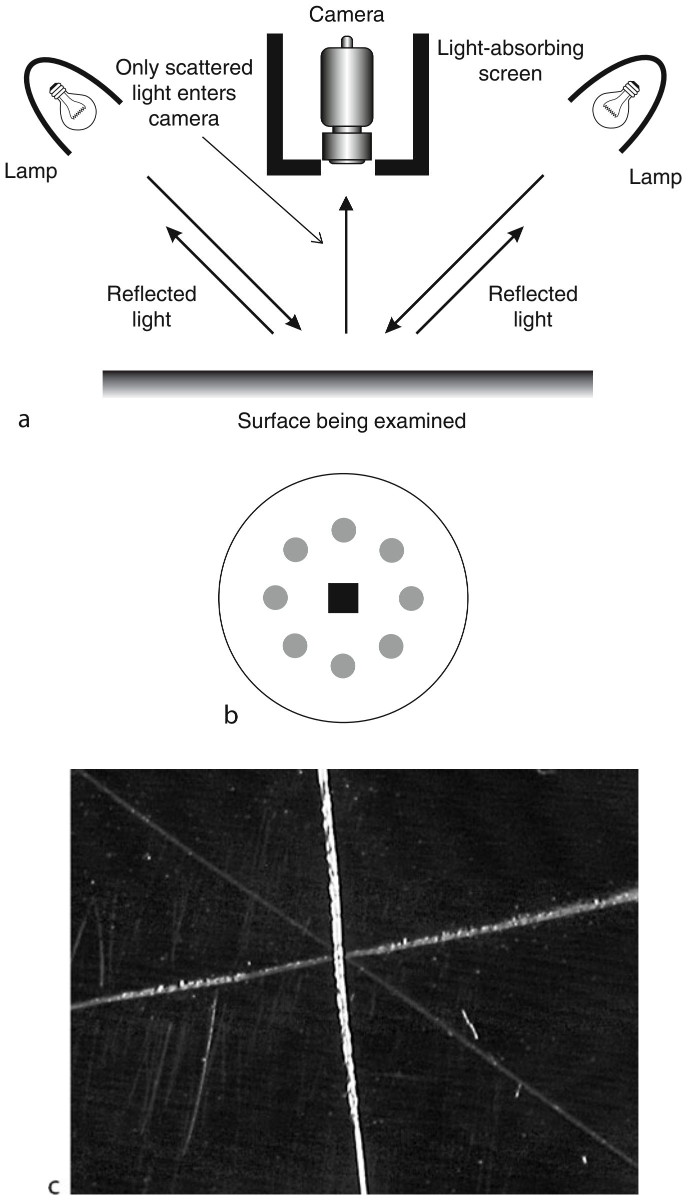
EQUIPMENT One of the following:

- Polarising prism. (This resembles a cube beam-splitter but has a multi-layer dielectric film coating on the interface surface between them.)
- Wollaston prism, made of calcite, or some other bi-refringent material. Other possibilities exist: Rochon, Ahrens, Sénarmont, Foster, Glan Thompson, Feussner, Bertrand
- A conventional beam-splitter and two linearly polarising sheets could be used instead

MATERIALS

- Non-metals, such as plastics, ceramics, glass, perspex (plexiplass), rubber, coated paper/card, etc.
- Painted/lacquered/varnished surfaces
- Surfaces with oil/moisture or other contamination

REMARKS Dielectric surfaces often produce reflections that are more strongly polarised in one direction than the other. (➋ *Fig. 40.110b*) Viewing a wet road, or the paint-work on a car body produces polarisation patterns, which are readily visible through polarising sun-glasses. If a ray of light impinges on a horizontal non-conducting (i.e., non-metal) surface at angle θ, then the ray reflected (angle $-\theta$) has a strong horizontally polarised component. (➋ *Fig. 40.110b*) A vertically polarising filter placed in the optical path strongly attenuates the horizontally polarised component, but leaves the vertically polarised component unchanged. (➋ *Fig. 40.110c*) Using the optical arrangements shown in ➋ *Fig. 40.110a* and *d*, it is possible to visualise defects in surface condition (e.g., breaks in coating, oil, water or other contamination), since these alter the relative strengths of the polarisation. components. A Wollaston prism consists of two prisms made of calcite, or some other bi-refringent material, cemented together. (➋ *Fig. 40.110d*) Both beams travel together (i.e., co-linearly) through the first prism although they experience different refractive indices. At the interface, one beam experiences an increase in refractive index, while the other observes a decrease. At this point, the beam paths diverge. The angular separation between their paths is increased as the beams emerge from the second prism. The Wollaston prism produces nearly symmetrical deviations of the light path, while others (e.g., the Rochon prism) produces asymmetric deviations. Several other types of bi-refringent polariser exist: Ahrens, Senarmont, Foster, Glan Thompson, Feussner, Bertrand. All of these are able to generate two images, so that the polarisation components can be viewed separately and simultaneously. It may, of course, be possible to enhance the differences between



◘ Fig. 40.109

Viewing a mirror-like surface without seeing the light source. (a) Optical layout. (b) Illumination diagram for an array of eight lamps. (c) Sample image: Scratches on a mirror. Light was projected from just one source (*top left*). This highlights some scratches but not others, emphasising the importance of multi-directional lighting

these images by using polarised lighting; we project the illuminating beam through a polarising sheet. Prisms made of calcite have a wide spectral range, extending from UV into the IR waveband: roughly 0.19–2.50 µm. To ensure good separation of the polarisation components, the incoming beam should contain only parallel rays. For this reason, a telecentric lens system, should be used. (Method 107) Stress can be visualised in glass and clear plastic by separating the horizontal and vertical polarisation components. (Method 122).

REFERENCES Bennett JM, Bennett HE (1978) Polarization. In: Driscoll WG, Vaughan W (eds) Handbook of optics. McGraw-Hill, New York, pp 10.1–10.164. ISBN 0-07-047710-8

LAYOUT DIAGRAMS AND SAMPLE IMAGE ➋ *Figure 40.110*

40.116 Method 116: Viewing Side Walls of a Bore

OBJECTIVES To view all round the side wall of a blind hole.

TYPICAL APPLICATIONS

- Inspecting internal (female) screw threads
- Examining bores for debris, scratches, cracks, pitting, pitting, corrosion, etc.

EQUIPMENT One or two convex conical mirrors. A double cone may be needed to ensure that the illumination strikes the bore surface at the correct angle. A plane mirror with a hole in its centre and collimator are also needed.

CAMERA Array.

LENS Standard.

MOTION Static.

GEOMETRY Blind (i.e., single ended) hole, large enough to insert probe.

MATERIALS Bright shiny materials may give some problems due to glinting.

REMARKS A collimated annulus of light is projected onto the illumination cone. From there, it is reflected onto the wall of the bore. The illumination cone angle must be chosen carefully, to ensure good illumination. In some situations, it is possible to use the imaging cone for illumination. In this case, the illumination is normal to the bore surface. The conical mirror maps (fixed-diameter) cylindrical coordinates (X,θ) into polar coordinates. The image formed by the conical mirror is sampled by the camera, using a Cartesian coordinate system. Let (U, V) be the position of a point in the digital image derived from the camera and (U_0, V_0) be the apparent position of the apex of the cone in that image. Hence, assuming that the imaging cone mirror has a half-angle of 45°

$$\theta = \tan^{-1}((V - V_0)/(U - U_0))$$
$$X = X_0 + \sqrt{[(U - U_0)^2 + (V - V_0)^2]}$$

where X_0 is the X-coordinate of the apex of the cone mirror along the bore axis. The conical mirror provides an example of anamorphic imaging: paraxial lines on the bore wall generate radial lines; circles around the circumference are mapped to concentric circles; a helical screw thread forms an Archimedes spiral in the output image. (Its radius varies linearly with angle, θ.) To avoid blurring, the camera lens should have a high *f*-number.

REFERENCES Batchelor BG (1978) Proposals for the automatic visual inspection of female screw threads. In: Proceedings of the 3rd international conference on automated inspection and product control, Nottingham. IFS, Bedford, pp 157–176
Conical mirror. Edmund Optics, http://www.edmundoptics.com. Accessed 6 Sep 2011

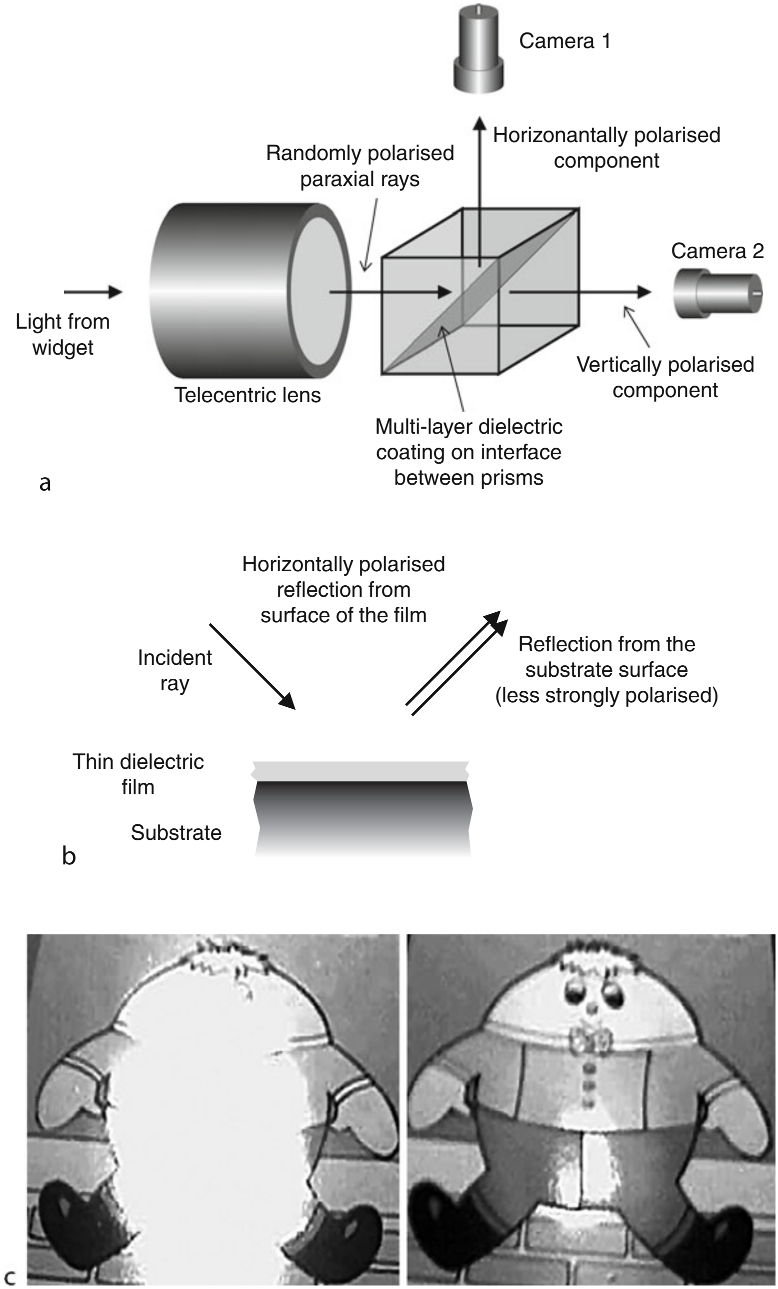
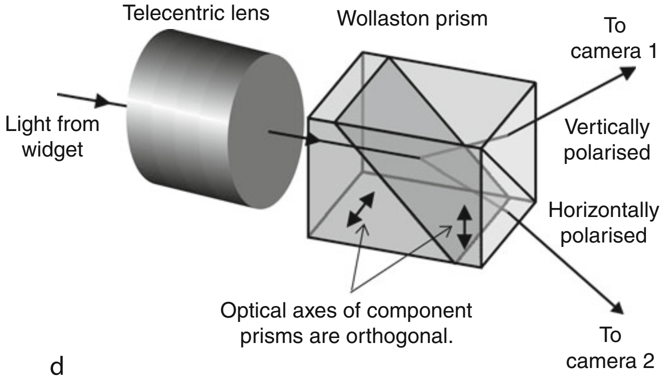
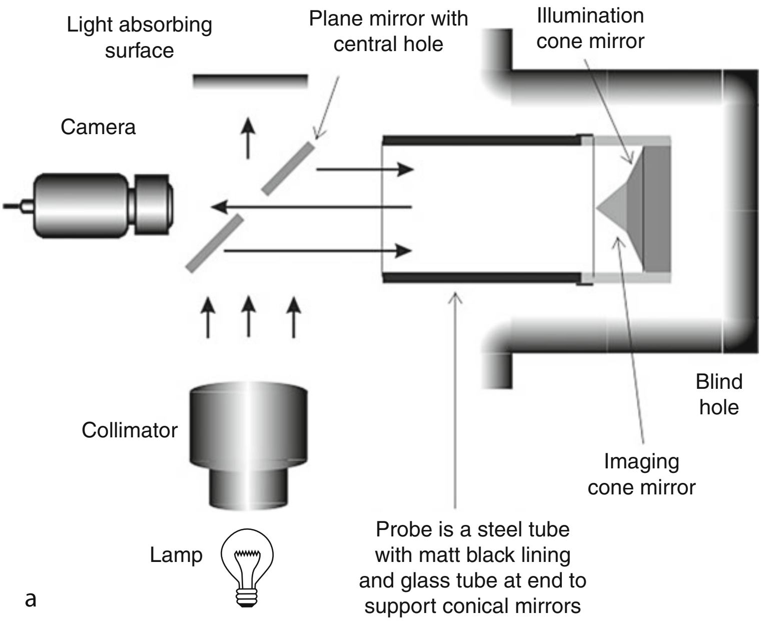

Camera 1
Horizonantally polarised component
Randomly polarised paraxial rays
Camera 2
Light from widget
Telecentric lens
Vertically polarised component
Multi-layer dielectric coating on interface between prisms
a
Horizontally polarised reflection from surface of the film
Incident ray
Reflection from the substrate surface (less strongly polarised)
Thin dielectric film
Substrate
b
c

Fig. 40.110
(Continued)



◘ Fig. 40.110
View orthogonally polarised images. (a) Optical set-up, using a cube polariser. (b) Reflections from a dielectric surface are polarised. (c) Sample image: reflections from a glossy printed card. *Left:* No polariser present. *Right:* With (plastic-sheet) polariser (An IR-blocking filter was inserted in the optical path, as this polariser does not operate properly in the NIR waveband.) (d) Using a Wollaston prism


Light absorbing surface
Plane mirror with central hole
Illumination cone mirror
Camera
Blind hole
Collimator
Imaging cone mirror
Probe is a steel tube with matt black lining and glass tube at end to support conical mirrors
Lamp
a

◘ Fig. 40.111
(Continued)

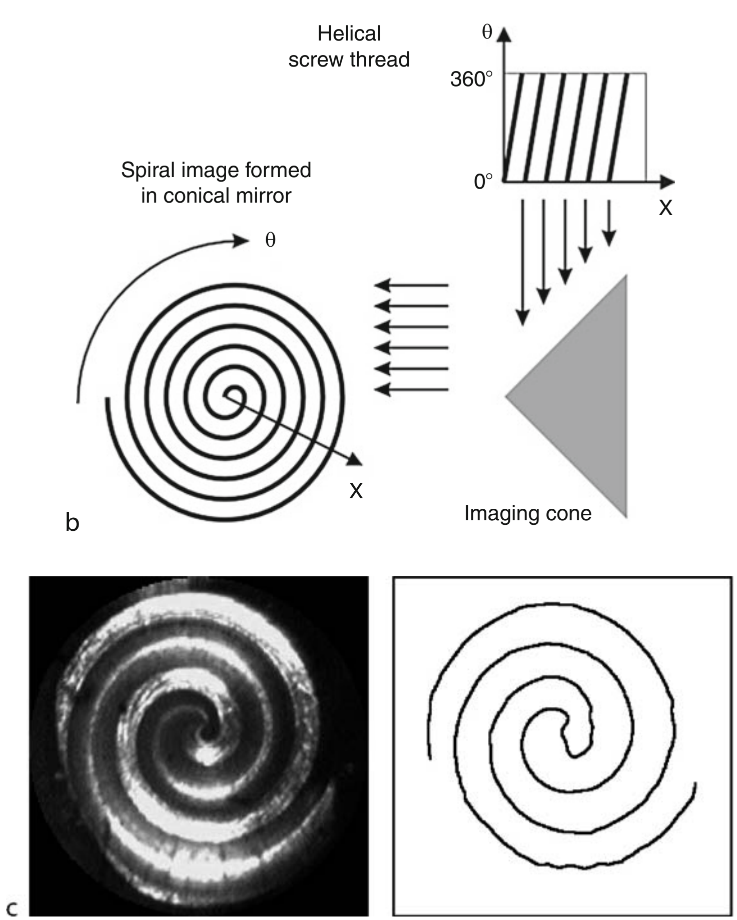


Fig. 40.111

Examining the side-wall of a hole using a conical mirror. (a) Optical layout. (b) The conical mirror transforms a helix (female screw thread) into an Archimedes spiral (c) Sample image: female screw thread. *Left:* Image obtained using a small conical mirror (10 mm diameter). Illumination was supplied by a fibre optic ring light. *Right:* Same image processed

40.117 Method 117: Viewing Aerosol Spray

OBJECTIVES

- To analyse the overall shape of an aerosol spray cone
- To examine the internal structure splitting of the cone and identify jets and flying drops of liquid

TYPICAL APPLICATIONS Inspection of

- Domestic, commercial and industrial aerosol sprays
- Fuel sprays for internal combustion engines, oil-fired heaters, etc.
- Coating and painting, sprays

EQUIPMENT Environmental protection enclosure is used to confine the spray and avoid contamination of optical equipment. This enclosure is fitted with two separate windows for illumination and viewing the spray cone. The internal surfaces of these windows must be kept clean, possibly using air purges. At the end of each inspection cycle, the atmosphere within the enclosure should be purged by a current of air.

LIGHTING The lighting sub-system may use fibre optics, an array of LEDs or fluorescent tubes, to ensure uniform, cold (IR-free) illumination of the spray cone. Care must be taken to eliminate UV, to avoid fluorescence. The lighting unit(s) must not be allowed to raise the temperature within the environmental protection enclosure, as this may alter the nature of the aerosol spray.

CAMERA Array.

LENS Standard.

MOTION Static.

REMARKS The importance of keeping the windows clean within the environmental protection enclosure cannot be over-emphasised. Similar lighting techniques can be used to obtain images of smoke plumes. A fine jet of smoke is often deliberately introduced into wind-tunnels, so that the air-flow pattern can be visualised. The jet may be illuminated in the same way, using Dark Field Illumination (Method 27), or a combination of the two.

REFERENCES

- Batchelor BG, Cotter SM (1983) Parameter measurement on aerosol sprays by image processing. Sensor Rev 3(1):12–16
- Hamamatsu Photonics K.K., System Division, Hamamatsu City. Cat Method No. SSCS 1008E02, June 1991 CR (9010)

LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.112*

40.118 Method 118: Viewing Inside an Irregular Transparent Object

OBJECTIVE To observe internal detail within a piece of material of known refractive index, and irregular shape.

TYPICAL APPLICATIONS

- Detecting foreign material, air bubbles, fractures, etc. in crystals, such as gems
- Examining cut glass objects for air bubbles and inclusions

EQUIPMENT Bath of water, oil, or other liquid, having approximately the same refractive index as the material being inspected.

GEOMETRY Complex and/or irregular. May not be known.

MATERIALS Transparent, plastics and other transparent materials. The refractive index must be fixed and known. Diamond and other materials with very high refractive index cannot be viewed in this way, because it is not possible to obtain a transparent liquid with a matching high refractive index.

REMARKS If the difference in the refractive indices of the solid material (μ_s) and liquid (μ_l) are exactly equal, the outer edge of the crystal is totally invisible. If μ_s and μ_l are nearly equal, the outer edge can be seen but with some difficulty. (Recall that ice cubes floating in water are nearly invisible. Only trapped air bubbles are clearly seen.) This assumes the surface of the solid material is clean and does not have air bubbles adhering to it. It is possible, therefore, to use

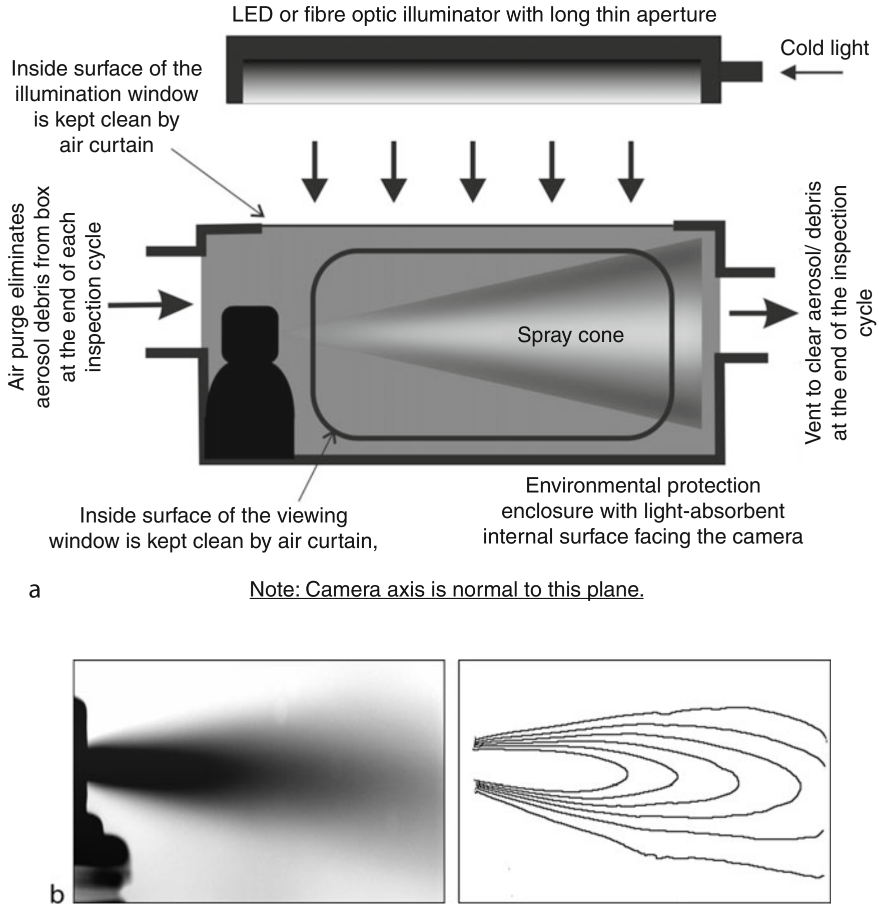

LED or fibre optic illuminator with long thin aperture
Cold light
Inside surface of the illumination window is kept clean by air curtain
Air purge eliminates aerosol debris from box at the end of each inspection cycle
Spray cone
Vent to clear aerosol/ debris at the end of the inspection cycle
Inside surface of the viewing window is kept clean by air curtain,
Environmental protection enclosure with light-absorbent internal surface facing the camera
a
Note: Camera axis is normal to this plane.
b

Fig. 40.112
Inspecting aerosol sprays. (a) Optical layout. The inside surface of the environmental protection enclosure should be cleaned regularly and the atmosphere purged after each inspection cycle. (b) Sample image: domestic furniture polish. *Left:* The negative of the original image is shown here to increase clarity. *Right*: Isophotes (intensity contours) indicate the density of the aerosol

water (μ = 1.3) to provide an approximate match for glass (μ = 1.5). Some liquids of organic origin have a high refractive index can be used to view inside gems such as sapphire and ruby. However, such liquids are often expensive and may be toxic. It is better to view the sample through an optical window set in the side-wall of the container (camera 2 in *Fig. 40.113*), rather using an overhead camera (camera 1). This avoid image distortion due to ripples on the surface of the liquid and, hence, makes the optical system more tolerant of vibration. It also eliminates highlights, due to reflections on the liquid surface.

LAYOUT DIAGRAMS *Figure 40.113*

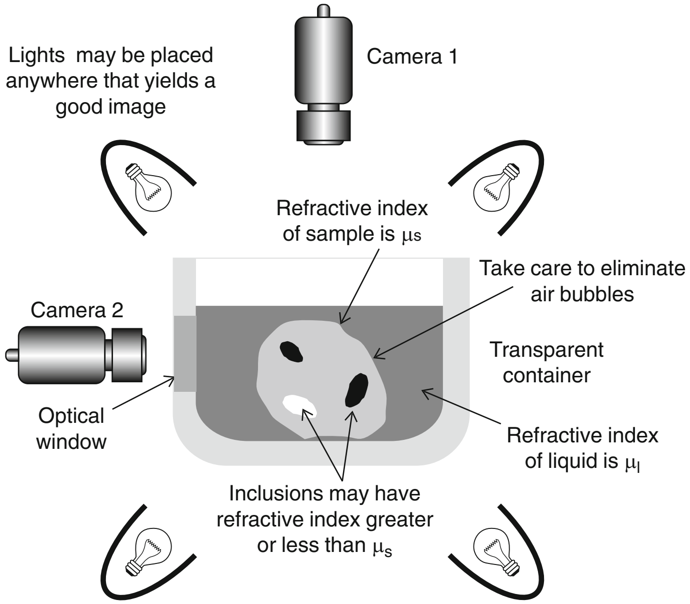

Lights may be placed anywhere that yields a good image
Camera 1
Refractive index of sample is μ_s
Take care to eliminate air bubbles
Camera 2
Transparent container
Optical window
Refractive index of liquid is μ_l
Inclusions may have refractive index greater or less than μ_s

Fig. 40.113
Viewing inside an irregular transparent object by immersing it in a bath of liquid with a similar refractive index

40.119 Method 119: Viewing Small Objects (Short Focus Lens Attachment)

OBJECTIVE To view small objects using a standard zoom lens.
TYPICAL APPLICATIONS Experimental work for prototype development, when viewing objects of approximate size 5–50 mm.
EQUIPMENT Short focus lens attachments, screwed onto a standard zoom lens.
REMARKS The combination of a standard zoom lens and a set of short-focus lenses provides a convenient macro zoom lens that is ideal for prototype development. Short-focus lens attachments produce slightly inferior picture quality, compared to extension tubes. However, they are rather more convenient to use, since they are easier and quicker to change. Short-focus lenses are available in a range of powers. Chromatic aberration can be problematical with lens attachments. For this reason, monochromatic or narrow-band lighting is preferred. A useful set of lenses for experimentation consists of a standard zoom lens, used in conjunction with +1,+2 and +4 dioptre short-focus attachments. (The power of a lens, measured in dioptres, is the reciprocal of its focal length in metres.) The effect of adding two or more short-focus lenses is additive. The corners of the picture may be distorted and blurred when large-power short-focus lenses are fitted. In addition, some vignetting may occur. The short-focus/zoom lens combination is best used for experimental application analysis. Once a good image has been obtained in the laboratory, it may be possible to select an equivalent a macro lens for use in the target system.

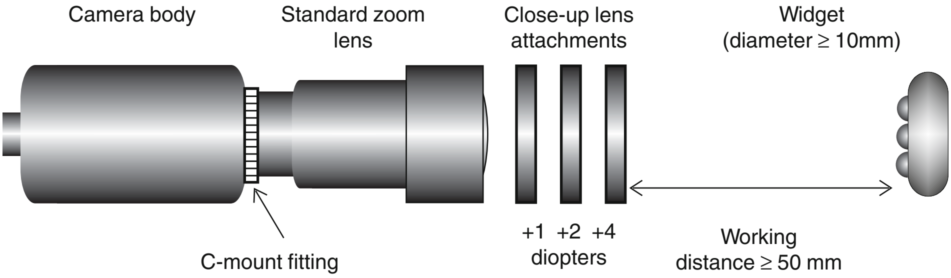


Fig. 40.114
Short-focus attachment lenses screw onto the main lens. They can be cascaded, if necessary, to reduce the focal length further

Table 40.2
Focal lengths for a standard lens fitted with a short-focus attachment. All distances are in millimetres

Short-focus attachment	Nominal setting, main lens			
	∞	3,000	1,000	500
1 dioptres	100	75	50	33
2 dioptres	50	43	33	25
3 dioptres	33	30	25	20
5 dioptres	20	19	16	14
10 dioptres	10	10	9	8

LAYOUT DIAGRAM ❯ *Figure 40.114*
FOCAL LENGTH CALCULATION See ❯ *Table 40.2*

40.120 Method 120: Viewing Small Objects/Features

OBJECTIVE To view objects/features of microscopic size.
TYPICAL APPLICATIONS Inspecting the following objects and surfaces:

- Bearing surfaces (wear damage)
- Metallurgical samples
- Micro-electronic devices
- CDs/DVDs
- Printing (fine structure of colour printing)
- Electrical switch contacts
- Surgical/razor blades (cutting edge)
- Metal-working tools (cutting edge)
- Abrasives
- Gems

- Internal structure of materials (viewing cross sections of industrial foams, baked dough and other food products)
- Textile fibres

EQUIPMENT Either of the following:

- Microscope, fitted with a suitable adaptor for a video camera (e.g., C-, CS-mount, etc.). Many modern microscopes allow the user to view the optical image simultaneously with image capture/recording by a video or film camera
- Video microprobe. This is a shortened variant of the endoscope that has been designed to provide high optical magnification. Light can be projected onto the widget from an in-built a ring light, surrounding the objective lens element. A microprobe has no mounting, or translation stage, for the sample being examined

CAMERA Array, monochrome or colour.

LENS The microscope/microprobe is fitted in lieu of a standard lens.

MOTION Static – avoid vibration.

GEOMETRY Features larger than about 20 μm can usually be examined optically. Objects smaller than this usually require the use of an electron microscope.

MATERIALS Provided suitable lighting is used, any solid material, which may be opaque, translucent or transparent.

REMARKS There are many possible lighting methods available for use with for microscopes including: front, inclined, grazing, side, back, dark-field, phase contrast, fluorescence, polarised, coloured, ultra-violet, infra-red, etc. Etching/staining can be used to enhance image contrast.

REFERENCES Hill DA (1976) Fibre optics. Business Books, London

LAYOUT DIAGRAM ➋ *Figure 40.115*

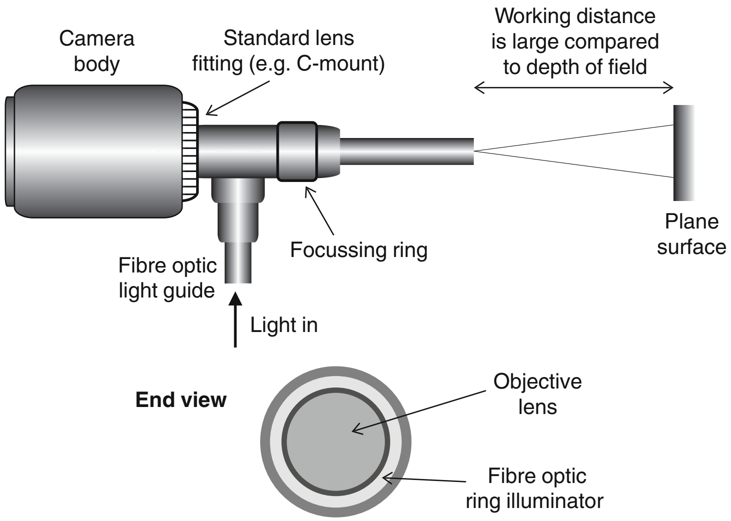

Camera body
Standard lens fitting (e.g. C-mount)
Working distance is large compared to depth of field
Plane surface
Focussing ring
Fibre optic light guide
Light in
End view
Objective lens
Fibre optic ring illuminator

Fig. 40.115
Video microprobe

40.121 Method 121: Viewing Small Objects (Extension Tubes)

OBJECTIVE To view small objects using a standard lens.
TYPICAL APPLICATIONS Experimental work for prototype development, when viewing objects of approximate size 5–50 mm.
EQUIPMENT Extension tubes fitted to standard lens.
REMARKS The combination of a standard lens and an extension tube provides a convenient alternative to a macro lens that is suitable for prototype development in the laboratory. Extension tubes produce slightly superior picture quality, compared to short focus lens attachments but are more difficult to use. Extension tubes lenses are available in a range of powers. For the standard C-mount lens fitting, a useful set of tubes for experimental purposes consists of 20 mm, 10 mm, 5 mm and a set of washers of various thicknesses. The effect of adding two or more extension tubes lenses is additive. When using extension tubes, care are should be taken to avoid vignetting. Extension tubes are well suited for experimental problem analysis; once a good image has been obtained in the laboratory, it may be possible to substitute an equivalent macro lens for the factory-floor system.
LAYOUT DIAGRAM ➋ *Figure 40.116*

40.122 Method 122: Viewing Stress in a Transparent Sheet

OBJECTIVE To display stress patterns in glassware and transparent sheet materials.
TYPICAL APPLICATION

- Checking annealing of glass-plastic hollow-ware (bottles, jars and other containers)
- Visualising stress in experimental components test structures and components. (The structure to be tested is modelled in glass or perspex.)

EQUIPMENT Two linear polarisers. An infra-red (IR) blocking filter may also be needed. The light source should not emit significant amounts of infra–red. (For this reason, filament lamps should be avoided.)
CAMERA Array.
LENS Standard.
MOTION Static.
MATERIALS Clear glass, perspex (plexiglass), plastic, polycarbonate, celluloid, acetate film, etc.

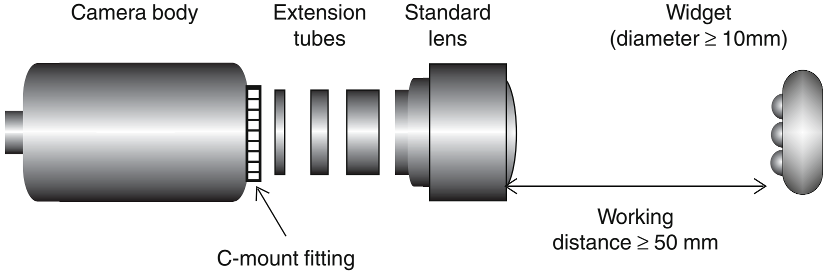
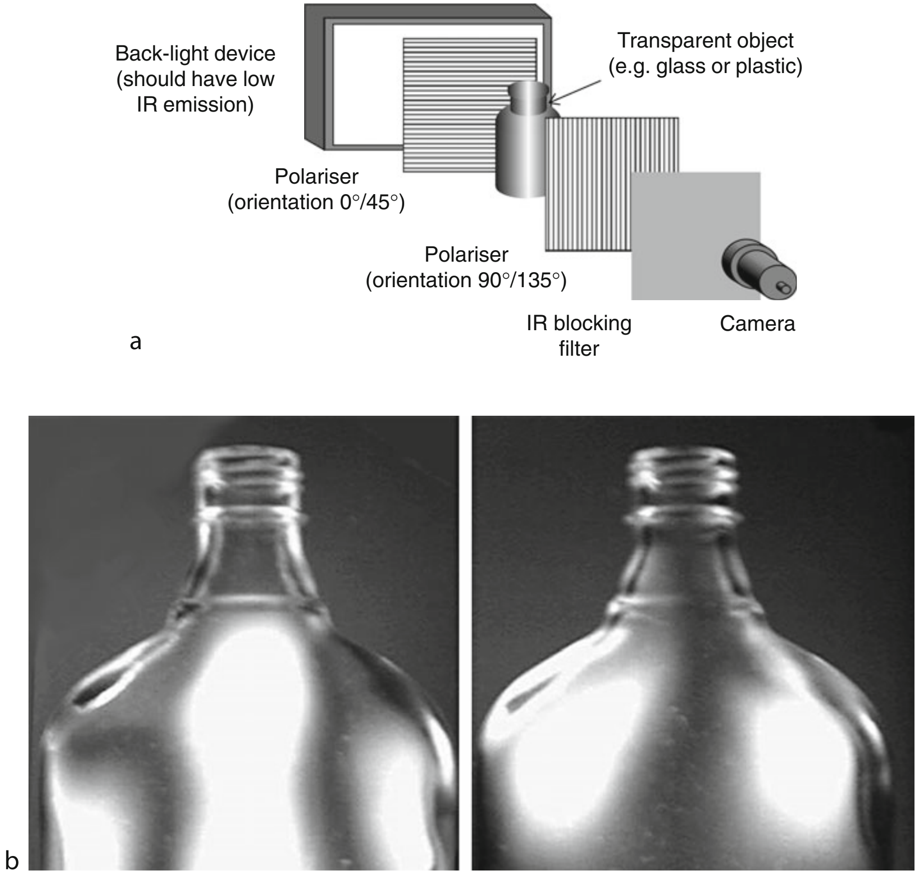


Fig. 40.116
Extension tubes fit between the camera and its main lens

REMARKS Stress is a vector quantity, which varies in magnitude and direction, from point to point. In a sheet of glass, only those components of the stress vector that are normal to the optical path (i.e., parallel to its plane sides) are visible. Some polarisers do not function in the near infra-red region of the electro-magnetic spectrum, while many solid-state cameras have a significant response to IR. As a result, images obtained without the IR-blocking filter shown in ➋ *Fig. 40.117a* may have a low contrast. In some cases, the stress patterns are not visible at all to a camera, even though they can clearly be seen by eye. An optical filter which absorbs infra-red will greatly improve the contrast. The light source should not emit strongly in the IR wave-band; filament lamps are not recommended. The optical set-up in ➋ *Fig. 40.117a* is the classical arrangement shown in school-level text books. However, it is clear from ➋ *Fig. 40.117b* that the



■ Fig. 40.117
Viewing stress patterns using two crossed linear polarisers. (a) Optical arrangement. (b) Sample images: stress patterns in a glass "hip flask" (a flattened bottle). *Left:* polarisers at 0° and 90°. *Right:* polarisers at 45° and 135°. The effects of the front and back walls of the bottle are superimposed

image seen with polarisers at one orientation (e.g., 0°/90°) does not highlight the stress patterns completely, as stresses in some directions are not made visible. It is necessary to rotate the polarisers by 45° (to 45°/135°) to obtain a second image. These two images are then added to obtain a measure of the total stress in the plane of the sheet. In order to avoid stress direction dependence, a pair of circular polarizers may be used instead of two linear polarisers. Notice that circular polarisers should be used with monochromatic light whose wavelength matches that of the polariser design.

REFERENCE Large sheets of polarising laminated film is available from a number of suppliers, for example, Edmund Optics, http://www.edmundoptics.com/. Accessed 6 Sep 2011

OPTICAL LAYOUT AND SAMPLE IMAGES ➋ *Figure 40.117*

40.123 Method 123: Viewing Through a Small Aperture

OBJECTIVE To view inside a tube or vessel that has only a small viewing aperture, without entering it.

TYPICAL APPLICATIONS Inspecting pipes, tubes, internal threads, engine cylinders, hydraulics cylinders, tanks.

EQUIPMENT Hypercentric lens with small spacer between it and the camera, ring light.

CAMERA Array camera.

MOTION Static.

GEOMETRY Typically hollow cylinder.

MATERIALS Not critical.

REMARKS Other options for lighting are available, including coaxial illumination and viewing.

REFERENCES Hypercentric lens. Light Works, LLC. http://www.lw4u.com. Accessed 2 Feb 2011

LAYOUT DIAGRAM AND SAMPLE IMAGE ➋ *Figure 40.118*

40.124 Method 124: Visual Sensor On Robot

OBJECTIVE To provide the following facilities:

- Camera, with accompanying lights, mounted on a robot arm
- Range finder, mounted on a robot arm
- Image sensor and range finder, mounted on an autonomously guided vehicle

TYPICAL APPLICATIONS Inspecting an array of components held on a large tray, large panels, (e.g., refrigerator, cooker, automobile/aircraft body panels, etc.), complex assemblies of components, checking machining, assembly, stacking, painting, spraying, etc. Agricultural/horticultural robotics: harvesting, milking, pruning, potting plants, etc. Robotic trimming of natural products: trimming vegetables, fruit, timber, butchery, etc.

EQUIPMENT Rugged camera head containing super-bright LEDs and diode laser light-stripe generator.

CAMERA Miniature array camera.

LENS Short focus, wide angle lens.

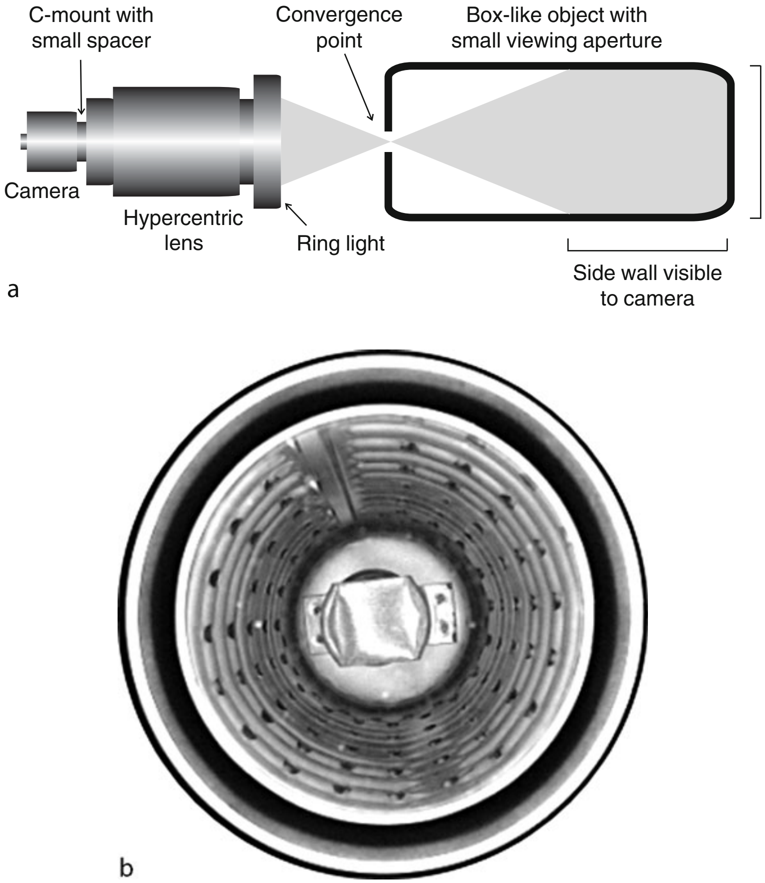

C-mount with small spacer
Convergence point
Box-like object with small viewing aperture
Camera
Hypercentric lens
Ring light
Side wall visible to camera
a
b

Fig. 40.118
Hypercentric lens used as a borescope (a) Layout (b) Interior of a car oil filter, viewed through an 18 mm hole [Image kindly supplied by Spencer Luster, Light Works. http://www.lw4u.com]

REMARKS An earlier design used fibre optic bundles for both illumination and imaging. However, this is not robust, as the continued flexing of a fibre optic bundle is likely to cause a small proportion of the optical fibres within it to break. This results in uneven illumination of the widget/assembly being examined, and black spots in the image. Moreover, the robot has to cope with the additional load placed on it by the stiffness of two ruggedised bundles with tough cladding. The design in *Fig. 40.119a* is generally superior. The advent of very small cameras and very bright, efficient LEDs makes it possible to build a unit like this within a small volume 50×50×50 mm^3. The video signal can be conveyed to the image processing system via a fine, flexible (and cheap) cable, or a short-range radio transmitter. Of course, radio/IR control of the laser and LEDs is also possible using *Bluetooth* technology. The camera-lighting head can be fixed to the robot *in lieu* of the gripper. Alternatively, it may be fitted to the wrist of the robot, alongside the gripper, within the gripper unit. It can even be held in a cradle beside the base of the robot until needed, then picked up and held in its gripper, during the inspection process. An air purge can be built into the camera-lighting head, to keep the optics clean in smoky, dusty and aerosol-laden atmospheres. (See Method 93) A large depth of field can be achieved by using a wide-angle lens, with a small aperture. The range finder (❯ *Fig. 40.119a*) helps in keeping the working distance nearly optimal and compensating for changes of magnification as working distance varies. It should be appreciated that there are significant algorithmic/computational problems involved in safely manoeuvring a camera round any object of complex form These are exacerbated if the scene being examined is of unknown form and size.

LAYOUT DIAGRAMS ❯ *Figure 40.119*

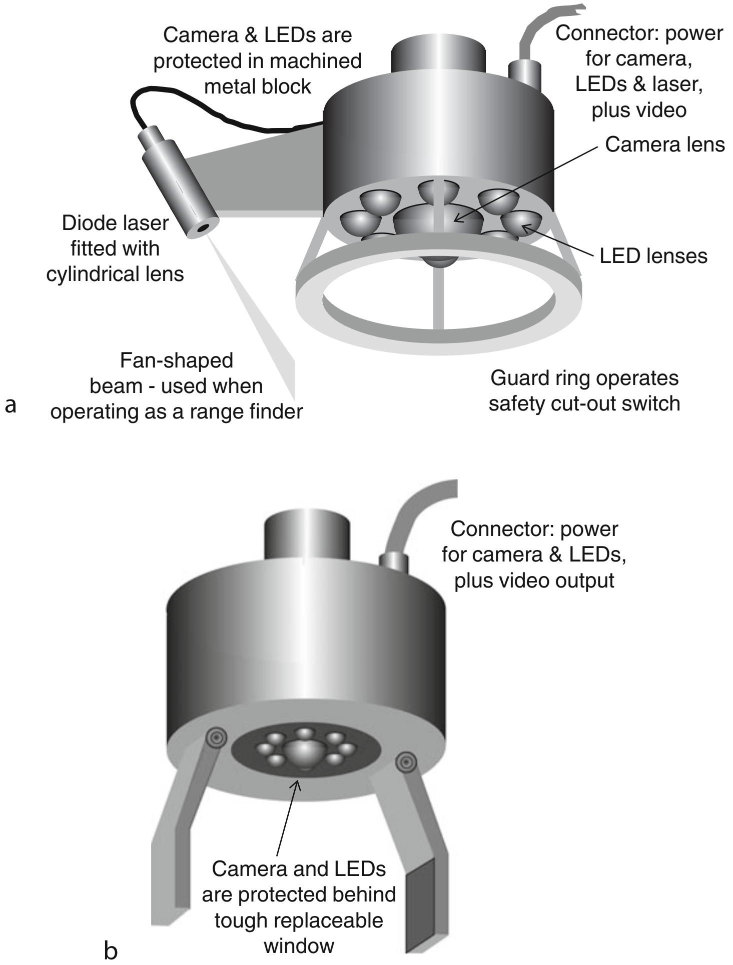

Camera & LEDs are protected in machined metal block
Connector: power for camera, LEDs & laser, plus video
Camera lens
Diode laser fitted with cylindrical lens
LED lenses
Fan-shaped beam - used when operating as a range finder
a
Guard ring operates safety cut-out switch
Connector: power for camera & LEDs, plus video output
Camera and LEDs are protected behind tough replaceable window
b

Fig. 40.119

Two camera-lighting units for attachment to a multi-axis robot. A wire-less video link (not shown here) avoids trailing cables. (a) Stand-alone unit. (b) Incorporated into the gripper unit

40.125 Method 125: Visualise Changes in Refractive Index

OBJECTIVE To highlight changes in refractive index in a transparent medium, such as air, or water.

TYPICAL APPLICATIONS

- Visualising gas, air/fluid flow
- Visualising convection patterns in air/liquid
- Detecting shards of clear glass/plastic in water
- Visualising fluid in glass/plastic container
- Analysing emulsions of clear liquids
- Studying aerodynamic flow and convection currents

EQUIPMENT

- *Schlieren Imaging*: Intense source of nearly monochromatic light such as a laser, filter, two high-precision parabolic mirrors. A specially constructed test cell is needed for studying fluid flow. This must have precision optical windows
- *Phase Contrast Imaging*: Three lenses. Two masks. Sample is in the form of a thin transmissive film, labelled "Transparency" in ➲ *Fig. 40.120c*

CAMERA Array.

MOTION Static, or slow enough for motion to be "frozen" by strobing camera or lights.

REMARKS Two methods are described: Schlieren Imaging (➲ *Fig. 40.120a*) and Phase Contrast Imaging (➲ *Fig. 40.120b*). Both of these employ optical interference to highlight small-scale changes in refractive index in a transmissive medium. These may be caused by local pressure, or temperature, gradients. Schlieren Imaging was developed by A. Toepler in 1864. Its name is derived from the German word meaning "streak." Phase contrast imaging is familiar in microscopy, where it is used extensively to obtain images of transparent micro-organisms and to increase contrast in images obtained from tissue samples. Both methods are highly sensitive to changes in refractive index and optical path length. The "Transparency" in ➲ *Fig. 40.120c* might consist of a thin test cell, with parallel walls, made of optical-quality glass and containing a clear fluid. A telecentric lens with a high f-number and a collimated light source can produce good Schlieren images.

REFERENCES Hecht E, Zajac A (1974) Optics. Addison Wesley, Reading, pp 478–481

LAYOUT DIAGRAM AND SAMPLE IMAGE ➲ *Figure 40.120*

40.126 Method 126: Visualising Heat Distribution

OBJECTIVE To visualise heat distribution, without using an expensive thermal imaging camera.

TYPICAL APPLICATIONS

- Prototype development of low-volume printed circuit boards, where the high cost of dedicated thermal imaging equipment cannot be justified
- Remote monitoring for over-heating in bearings, motors, engines, pumps, pipe-work, etc.
- Crack detection
- Checking structural integrity of adhesive joints

EQUIPMENT Thermo-chromic (liquid crystal) paint (TC-paint), or thin plastic film coated with such paint. TC-paint can be made to detect changes (i.e., change colour) over a narrow, well-defined range of operating temperatures. (These materials are used in disposable clinical

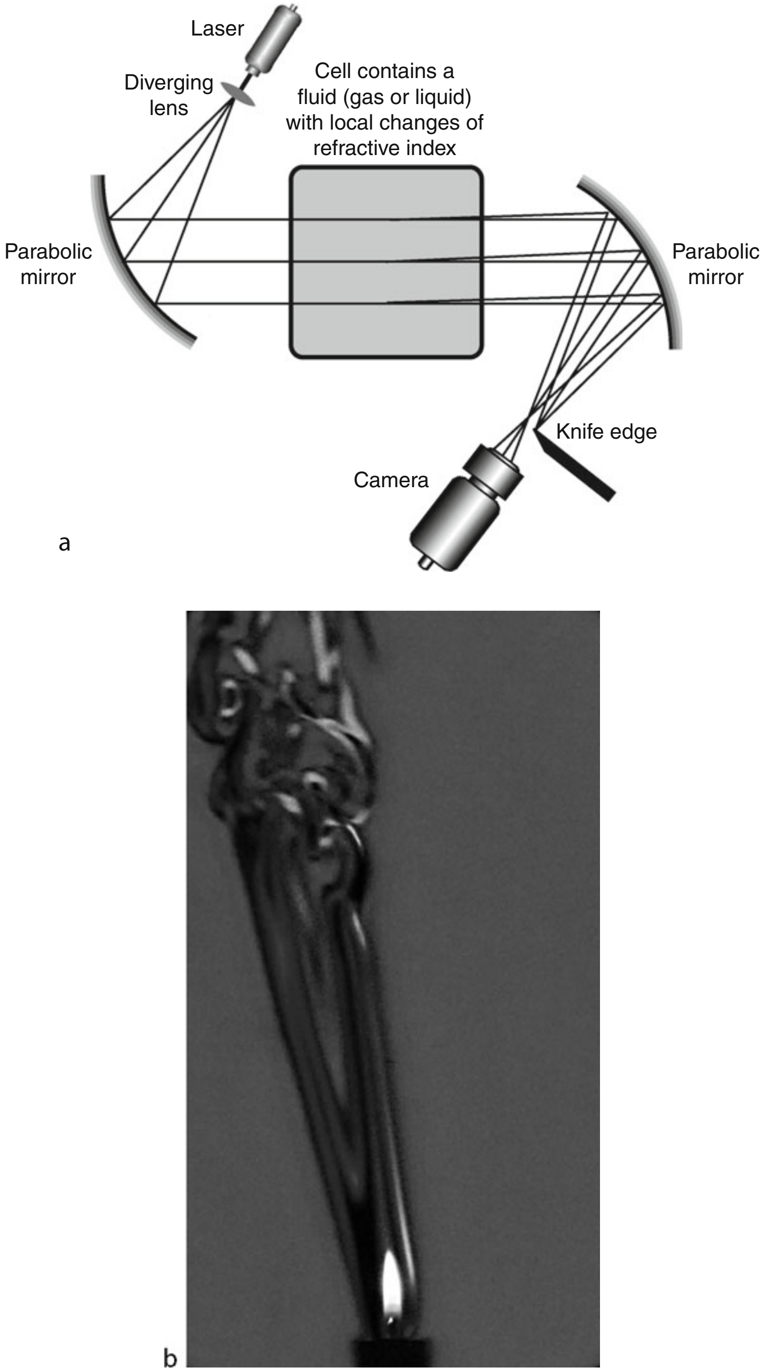

Laser
Diverging lens
Cell contains a fluid (gas or liquid) with local changes of refractive index
Parabolic mirror
Parabolic mirror
Knife edge
Camera
a
b

Fig. 40.120 (Continued)

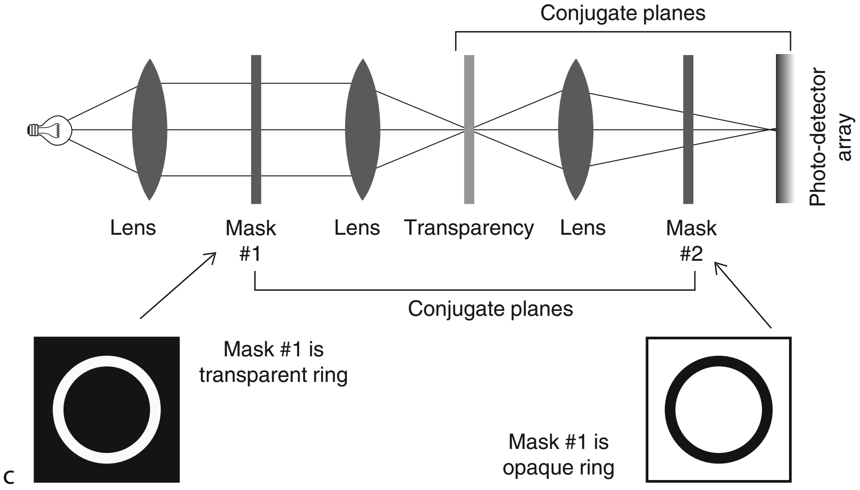


Fig. 40.120

Visualising changes of refractive index. (a) Schlieren imaging, optical layout. (b) Sample schlieren image of the thermal plume from a burning candle disturbed by a breeze from the right. Adapted from a photograph by Gary S. Settles, Penn State University. URL http://en.wikipedia.org/wiki/Schlieren (c) Phase contrast imaging, optical layout

thermometers.) The illumination must be "cold" (i.e., containing no significant amount of infra-red.) For this reason, an incandescent filament lamp is *not* suitable; a VIS LED array is ideal.

CAMERA Standard array-scan colour camera. A monochrome camera may be used in conjunction with an optical filter, to obtain good colour (i.e., thermal) differentiation.

MATERIALS Metals, plastics, ceramics, and other materials. The surface must be clean, non-absorbent, relatively smooth and able to accept the TC-paint without damage. Alternatively, thermo-chromic film may be attached by adhesive, applied uniformly to ensure good and consistent thermal contact.

REMARKS Thermo-chromic materials reversibly change colour as the temperature changes. They can be made as semi-conductor compounds, from liquid crystals, or using metal compounds. Some older forms of TC-paint are based on mercury compounds and are poisonous. The change in colour occurs at a well-defined temperature, which can be varied by doping the material. They can be supplied in liquid form, applied to a thin film, or mixed into moulding, or casting materials. This method can be applied to check crazing, cracking, adhesive joints, plating and other coatings, and to check for integrity in honeycomb and lattice structures. Heat may be deliberately applied at one end of a structure and the temperature changes observed over time along its length. In this way, it is possible to detect cracks, which impede thermal conduction and therefore alter the temperature-time profile. Using TC-paint provides the basis for a low-cost, low-precision inspection method. It is capable of providing only qualitative results, albeit at a well defined temperature threshold. It is well suited to detecting "hot spots" on printed circuit boards, or monitoring bearings for overheating. This

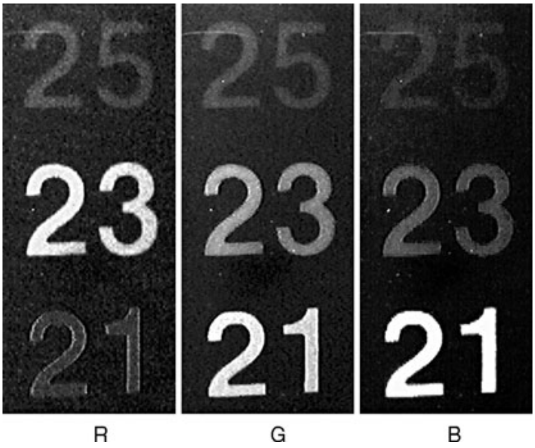


Fig. 40.121
Image derived from a thermo-chromic room thermometer. The picture shows the RGB colour channels for a thermometer. Numbers denote degrees Celsius. "Cold" near-white light was provided by an LED array

technique was developed for medical applications: using thermo-chromic film in good thermal contact with the skin, it is sometimes possible to detect hot-spots overlying breast tumours. Thermo-chromic materials have also been applied to small batteries providing an integral charge tester and to decorate T-shirts, coffee mugs, etc.
SAMPLE IMAGE ➔ *Figure 40.121*

40.127 Method 127: Wireless Image Acquisition

OBJECTIVE To provide a wireless connection between the camera and lights on the one hand and the control/image processing computer on the other. There are several factors to be considered:

(a) Power for the lighting module(s), camera and possibly the lens
(b) Control signals for the lighting module(s)
(c) Control signals for the cameras
(d) Control signals for the lens

TYPICAL APPLICATIONS

- Working in awkward locations where it is difficult to install cables
- Installing a camera on moving machinery (e.g., a lathe, hoist, robot, autonomic guided vehicle)

EQUIPMENT Various methods are described below, using

- Magnetically coupled coils, located close to each other on opposite sides of the wall of a container, forming a simple transformer
- Wireless (radio or IR) video link
- Cell phone video link
- Bluetooth

REMARKS There are numerous and varied circumstances where it is useful to operate a video camera and/or lights without providing direct electrical contacts:

- Autonomously guided vehicle (AGV). The wireless link avoids the use of a long, unwieldy and unreliable “umbilical” cable
- It may simply be more convenient (i.e., cheaper) to use a wireless data link, rather than installing cables, particularly if the inspection equipment is expected to be in situ for only a short time. This might typically apply to prototype systems, or short-run inspection applications

LAYOUT DIAGRAMS ➲ *Figure 40.122*

40.128 Method 128: X-Ray Imaging

OBJECTIVE To examine and measure internal features that are not be visible on the surface of an object or assembly.

TYPICAL APPLICATIONS

- Detecting foreign bodies in food, pharmaceuticals and toiletries. (The most important categories of mis-placed items are manufacturing waste, and natural contaminants. This includes articles such as false teeth, hair clips, contact lenses, screws, shards of glass, metal swarf, plastic splinters, bones, nut shells, rodents, insects, stones, twigs, etc.)
- To identify missing parts and mis-alignment in complex, multi-component products and assemblies
- Examining internal detail, correctness of assembly and internal alignment in safety-critical assemblies, such as munitions and electrical equipment

EQUIPMENT X-ray source and sensor, enclosed within a secure cabinet fitted with a range of safety devices. Widgets must be transported automatically through the x-ray beam.

CAMERA Array or line-scan detector. A high sensitivity camera is normally needed as the fluorescence generated by the x-ray beam is often weak.

REMARKS Care should be taken to choose a long-life x-ray tube, or one which can be fitted easily and quickly with a new filament. Safety is obviously of crucial importance when operating x-rays. However, with careful system design of the cabinet, there is no fundamental reason why x-ray inspection systems cannot be used perfectly safely in a factory. The maintenance and regular servicing of x-ray systems are crucial for safe and effective operation. Key points to note are:

(1) Fit micro-switches which turn off the power to the x-ray source when a door on the cabinet is opened
(2) Design the equipment so that no part of the body can be irradiated by the x-ray beam. For example, long entrance and exit tunnels can prevent operators from placing their hands within in the beam.
(3) Fit opto-electronic switches which turn the tube beam current *ON*, only when an object of the correct size is in place for inspection. Turn the current *OFF* again immediately afterwards
(4) Make the x-ray beam as narrow as possible. For an array sensor, use a pencil-shaped beam, if possible. When using a line-scan camera, mask unwanted rays, to create a thin fan-shaped beam

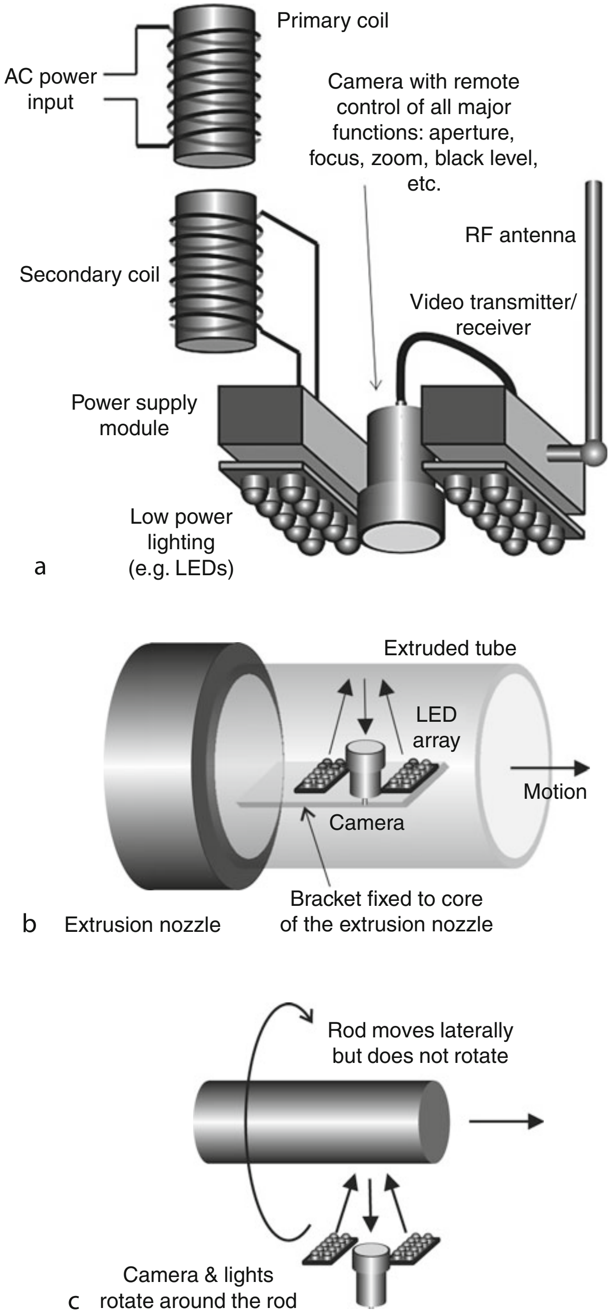

Primary coil
AC power input
Camera with remote control of all major functions: aperture, focus, zoom, black level, etc.
RF antenna
Secondary coil
Video transmitter/ receiver
Power supply module
Low power lighting (e.g. LEDs)
a
Extruded tube
LED array
Motion
Camera
Bracket fixed to core of the extrusion nozzle
b
Extrusion nozzle
Rod moves laterally but does not rotate
Camera & lights rotate around the rod
c

Fig. 40.122 (Continued)

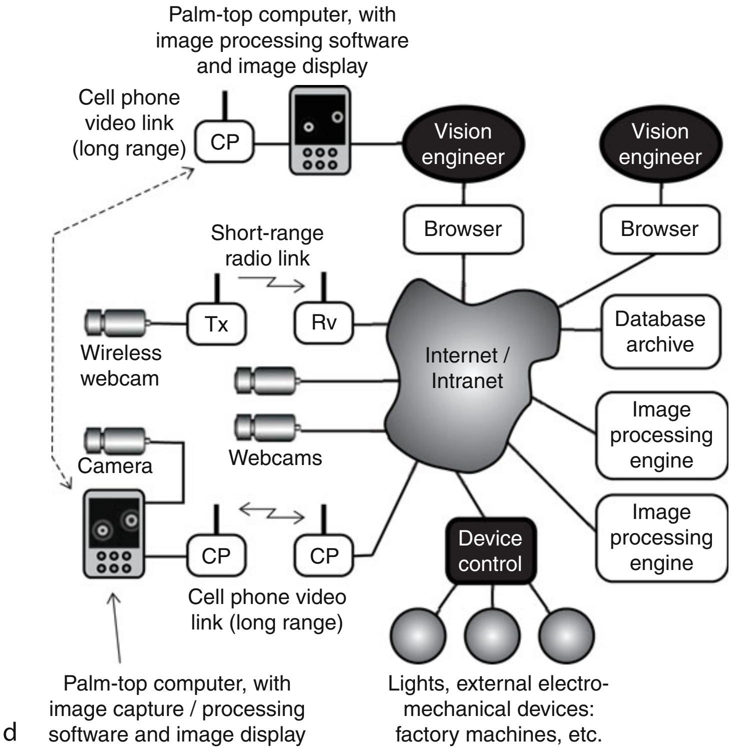

Palm-top computer, with image processing software and image display
Cell phone video link (long range)
CP
Vision engineer
Vision engineer
Browser
Browser
Short-range radio link
Tx
Rv
Wireless webcam
Internet / Intranet
Database archive
Image processing engine
Webcams
Camera
Image processing engine
CP
CP
Device control
Cell phone video link (long range)
Palm-top computer, with image capture / processing software and image display
Lights, external electro-mechanical devices: factory machines, etc.
d

Fig. 40.122

Wireless image acquisition. (a) Transmitting power across the thin walls of a closed non-magnetic container. Two juxtaposed coils form a simple transformer, able to transmit power for multi-LED lighting modules and a camera. A radio link transmits the video signal from the camera to the image processor (not shown). (b) Inspect the internal surface of a large-diameter extruded tube. (c) Inspecting all round a solid rod using a wireless video link. Power might be supplied by slip-rings. (d) Network configuration for wireless video transmission via cellular 'phone and/or short-range radio link

(5) Use appropriate shielding to attenuate all unwanted x-rays. Pay attention to scattered x-rays
(6) Service the machine regularly
(7) Unauthorised and untrained personnel should not be able to gain access to the interior of the cabinet to make modifications to the machine. The cabinet should be lockable and secure fixings used to prevent side panels from being removed with standard tools, such as a screw-driver
(8) All personnel who work near the x-ray system should wear radiation detector (film) badges. These must be checked regularly

Contrary to some people's misplaced fears, x-rays do not damage food, or make it radioactive and there is no residual radioactivity. The x-ray dose is minimal. The energy levels are minute,

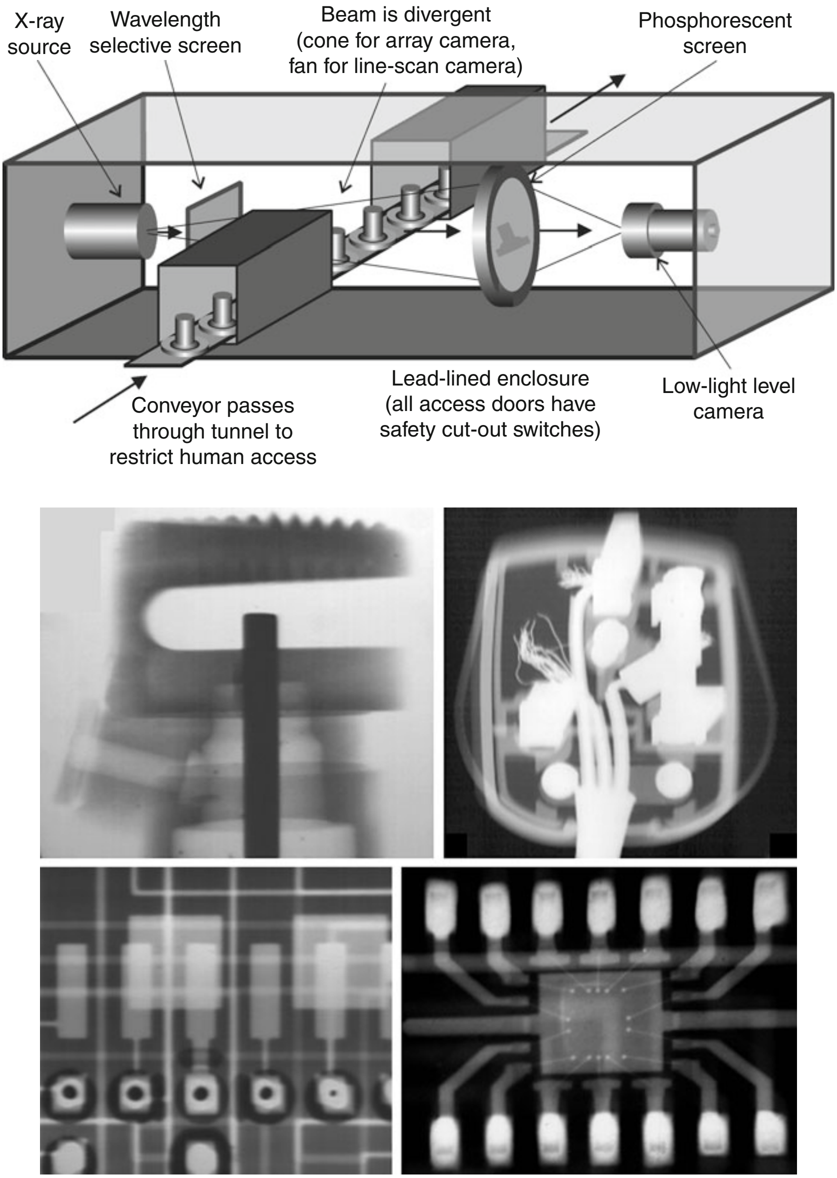

X-ray source
Wavelength selective screen
Beam is divergent (cone for array camera, fan for line-scan camera)
Phosphorescent screen
Conveyor passes through tunnel to restrict human access
Lead-lined enclosure (all access doors have safety cut-out switches)
Low-light level camera

Fig. 40.123

X-ray inspection of industrial artefacts. (a) Equipment layout. (b) *Top left:* Aerosol spray assembly. *Top right:* Power plug (UK standard, 240 V/13A, square-pin plug). Notice the dangerous loose strands of wire. *Bottom left:* Multi-layer printed circuit board. *Bottom right:* Dual in-line integrated circuit. When printed at higher resolution, fine connecting wires attached to the silicon chip are clearly visible

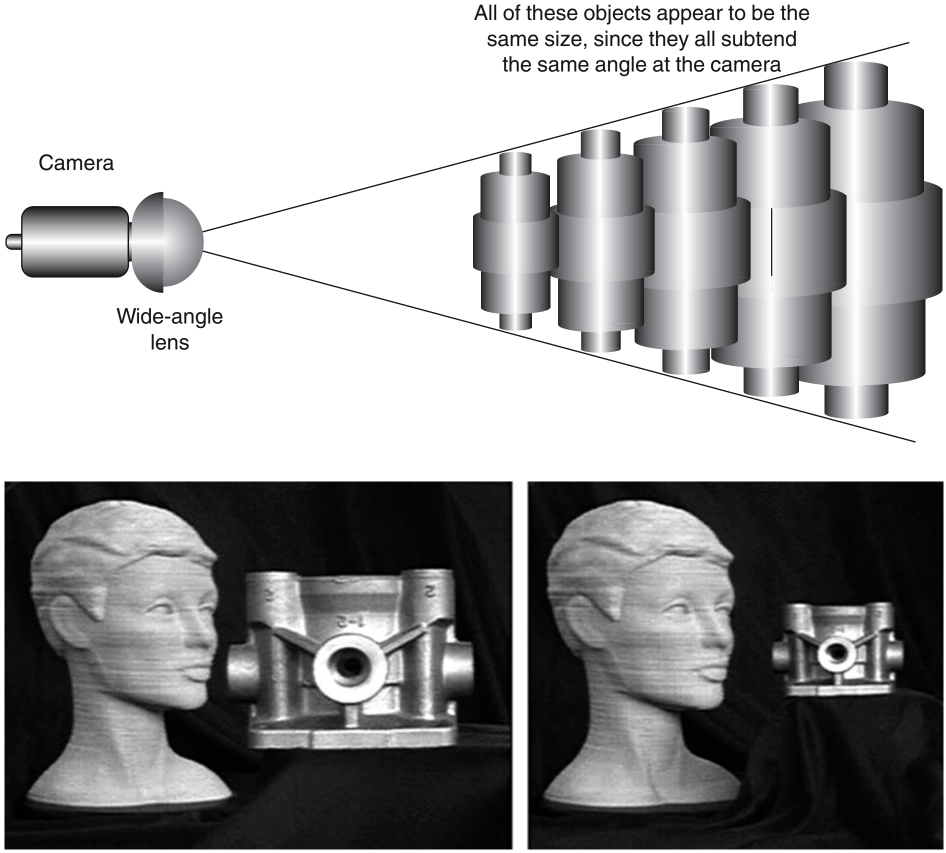

All of these objects appear to be the same size, since they all subtend the same angle at the camera
Camera
Wide-angle lens
2
1-3

Fig. 40.124

Varying magnification by changing the widget-to-camera distance. (a) Optical layout. In practice, the camera might be mounted on a robot arm, or a motorised slide-way. (b) Sample images: hydraulics manifold moving relative to the camera. The life-size sculptured head is at a fixed distance (same as the manifold in the image on the right). Notice that the focus is maintained despite the large proportional change in position (about 2:1). The number of pixels across the hydraulics manifold is reduced in the image on the right (Lens: focal: length 8 mm, *f*-number: 1.3.)

compared to those used, for example, during nuclear radiation sterilisation. Damage to large organic molecules (e.g., proteins in food products) does occur with very high x-ray doses but the effect is minute with the beam energy and exposure time normally used for inspection. Similar x-ray inspection methods are used for inspecting food materials, e.g., detecting bone in chicken, parasitic worms in fish, insect infestation in plant material, etc.

REFERENCES Graves M (1999) "X-ray machine vision for on-line quality control in food processing", PhD thesis, University of Wales Cardiff, Cardiff, Wales, UK

Graves M (2003) X-ray Bone Detection in Further Processed Poultry Production, in Graves M & Batchelor B G, Machine Vision for the Inspection of Natural Products, Springer. ISBN 1-85233-525-4

LAYOUT DIAGRAM AND SAMPLE IMAGES ➜ *Figure 40.123*

40.129 Method 129: Zooming By Movement

OBJECTIVE To vary the magnification of an optical system by varying the widget-to-camera distance.

TYPICAL APPLICATION Robot vision: using a camera with a fixed-focus lens, mounted on a robot arm, it is possible to perform a zoom function.

CAMERA Array.

LENS It is important that the lens is able to provide a large depth of field. A pin-hole lens, a wide-angle lens, or a standard lens with large *f*-number, may be used.

MOTION The position of the camera relative to the widget is controlled by a robotic device. Either the camera or the widget can be moved by the robot.

REMARKS The lens must be able to maintain sharp focus as the widget-to-camera distance varies over a wide range. Since the magnification is linearly related to the widget-to-camera distance, the distance between the camera and widget must be known accurately. It is important to realise that the spatial resolution varies as the magnification changes.

LAYOUT DIAGRAM AND SAMPLE IMAGES ➔ *Figure 40.124*

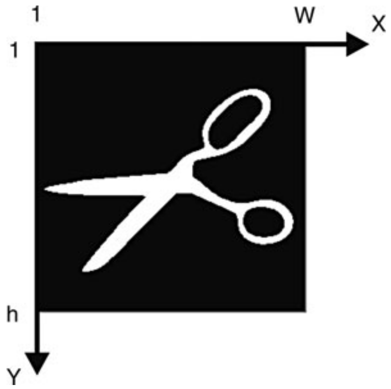
41 QT Image Processing Functions

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_41,

41.1 Notation

Coordinate axes



Input image 1 (associated with QT's Current Image)	*A = {A(x,y)}, [x,y] ∈ [1,w] × [1,h]*
Input image 2 (associated with QT's Alternate Image)	*B = {B(x,y)}, [x,y] ∈ [1,w] × [1,h*]
Result of image processing operation (replaces QT's Current Image)	*C = {C(x,y}, [x,y] ∈ [1,w] × [1,h]*

Intensity scale

$0 \le A(x,y) \le W, [x,y] \in [1,w] \times [1,h]$

$0 \le B(x,y) \le W, [x,y] \in [1,w] \times [1,h]$

Normalisation of the intensity scale is applied to ensure that

$0 \le C(x,y) \le W, [x,y] \in [1,w] \times [1,h]$

The operator ⇐ denotes a repeated assignment, performed on all pixels, $[x,y] \in [1,w] \times [1,h]$

The following assignments are equivalent

$\{C(x,y) \leftarrow \{\mathrm{f}(A(x,y), B(x,y))\}, [x,y] \in [1,w] \times [1,h]$

and

$C \Leftarrow f(A, B)$

Example (QT operator *adi*)

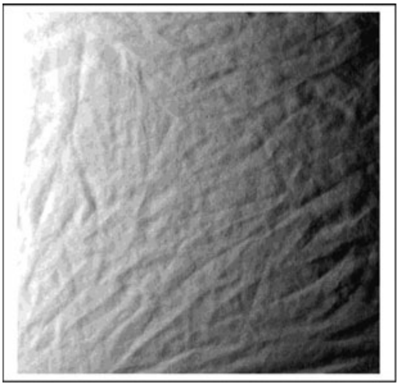
$C = (A+B)/2$ % Adds corresponding pixels in image arrays *A* and *B* and then divides by 2

"*Switch images*" indicates that the Current Image has been an altered by the function (*f*), following which the Alternate Image displays the previous Current Image

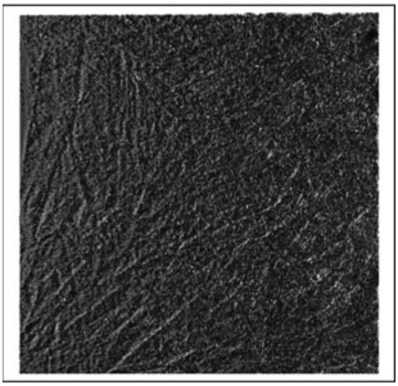
41.2 abd

Format	*abd*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Dyadic
Function	Absolute value of difference in intensities between current and alternate images
Definition	$C \Leftarrow \|(A - B)\|$
Switch images	Yes
Typical uses	Emphasising local differences between images, irrespective of which is brighter/darker
Remarks	*abd* is most commonly used as a component within more complex algorithms, such as filters which high-light texture, or detect bright/dark intensity anomalies
Illustration	
Input images	Creased fabric and blurred version of same (*caf*)
File number	28
Command	*caf*, *abd*, *enc*

Left: original (Other image is output of *caf*)



Right: after processing



41.3 abv

Format *abv*

Arguments None

Defaults Not applicable

Output values None

Image type Grey

Type Monadic

Function Fold the intensity scale about the mid-grey value. (White and black are both mapped to the same value: white.) This function calculates the 'absolute value' of the image intensity, referred to mid-grey. If {a(i, j)} and {b(i, j)} are two images of the same size, then the output is {c(i, j)}, where

$$c(i, j) \Leftarrow |\,(W + a(i, j) - b(i, j))|/2$$

This is repeated for all *points*, $[i, j]$, in the input images

Switch images Yes

Typical uses In conjunction with various filters (e.g., high-pass), so that dark and bright intensity anomalies are treated in a similar way

Remarks The function *abv* 'folds' the intensity scale about the mid-grey value, so that white and black are both mapped to the same value (i.e. white)

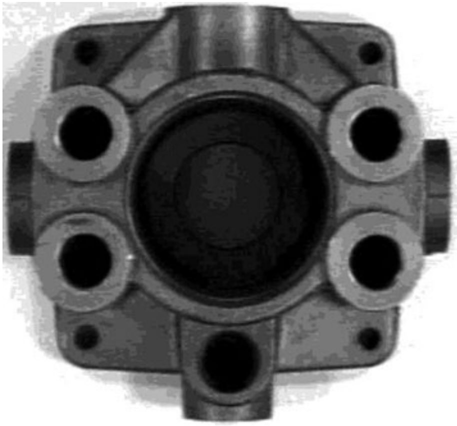
Illustration

Input image Hydraulics component

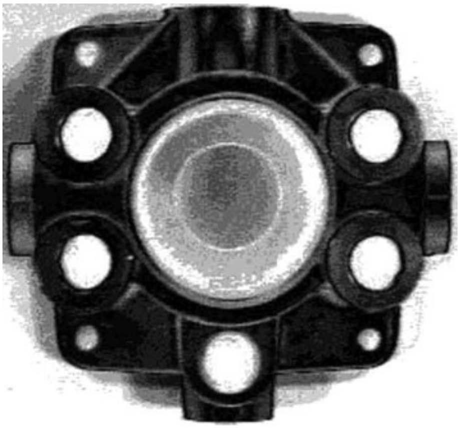
File number 37

Command *abv*

Left: original



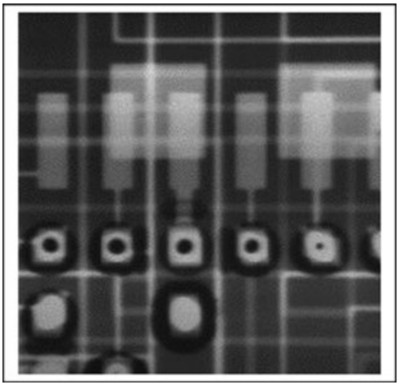
Right: after processing



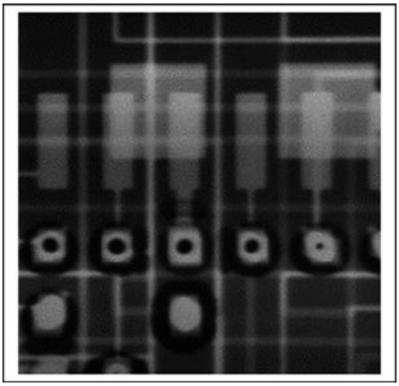
41.4 acn

Format	*acn*(*a*)
Arguments	a Increment to applied intensity values ($-255 \le a \le 255$)
Defaults	None
Output values	None
Image type	Grey
Type	Monadic
Function	Add a constant to all intensities in the current image
Definition	$C \Leftarrow A + a$ with hard limits placed at 0 and 255
Switch images	Yes
Typical uses	Used to improve contrast. Also used within functions, such as *blo* and *enc*
Remarks	Also see *blo*, *din*, *enc*, *hin* and *heq*
Illustration	
Input image	PCB
File number	135
Command	*acn*(*−34*) (The minimum intensity in the PCB image is 34)

Left: original



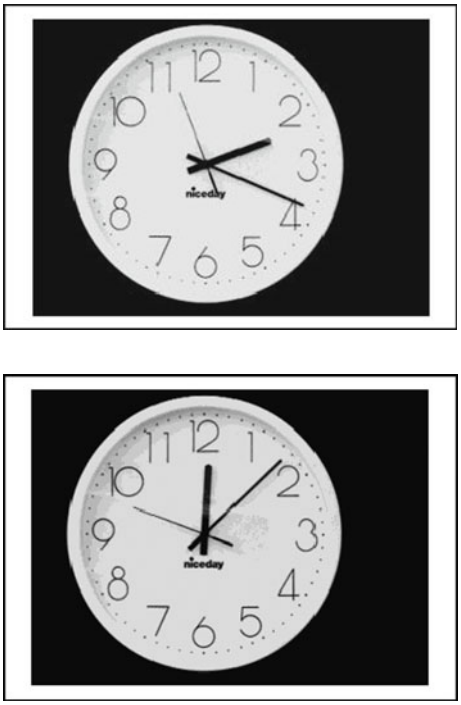
Right: after processing



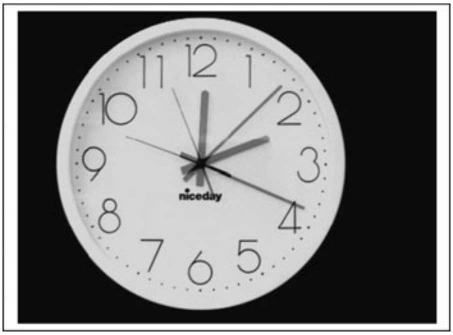
41.5 adi

Format	*adi*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Dyadic
Function	Add the intensities in the current and alternate images
Definition	C ⇐ (A + B)/2
Switch images	Yes
Typical uses	Image averaging Superimposing two images, for display purposes
Remarks	*adi* gives a similar effect to that seen in a photographic double exposure
Illustration	
Input image	Clock 1 and Clock 2
File number	142 and 143
Command	*adi*

Left: two originals



Right: after processing



41.6 aia

Format	*aia*(*a*)	
Arguments	*a*	Name of the folder where new images will be deposited
Default value	*a*	QT_images
Output values	None	
Image type	Binary, grey or colour	
Type	QT maintenance/extension	
Function	Add the current image to the image archive, stored in folder *a*, or by default in QT_images The new image file is in TIFF format	
Switch images	No	
Typical uses	*aia*	Saving pictures as they are captured from a camera
Remarks	*aia* is easier to use than *wri*, as the user does not have to worry about typing the file name. The image can be read back using *rni*. Also see *sia*	

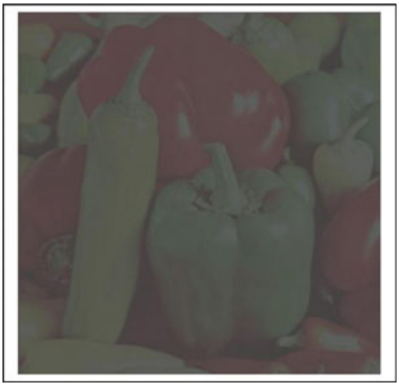
41.7 aic

Format	*aic*
Arguments	None
Output values	None
Image type	Colour
Type	Monadic
Function	Apply a linear stretching of the intensity scale of the RGB channel between the limits [*a*, *b*], where *a* is the minimum of the intensities in the RGB channels and *b* is the maximum. [*a*, *b*] is mapped to [0, 255]
Switch images	Yes
Typical uses	Increasing contrast of colour images
Remarks	The following have similar functions: *pci*('*enc*') *pci*('*heq*') *Also see air*, *pci*

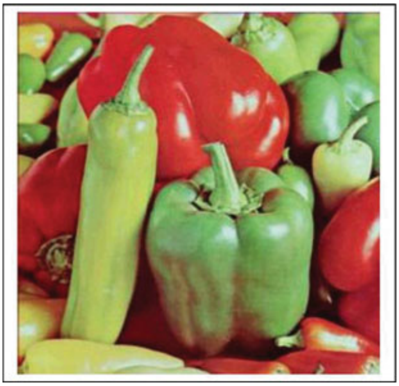
Illustration

Input image	Peppers
File number	73
Preprocessing	*rci*(*73*), *pci*('*hin*'), *pci*('*hin*'), *pci*('*hin*'), *acn*(*67*)
Command	*aic* *pci*('*enc*') *pci*('*heq*')

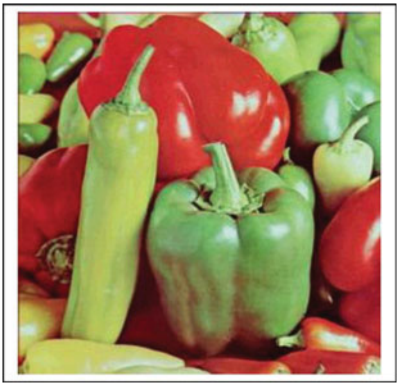
Top-left: original



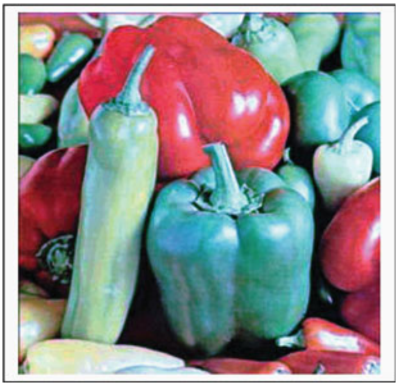
Top-right: *aic*



Bottom-left: after *pci*('*enc*')

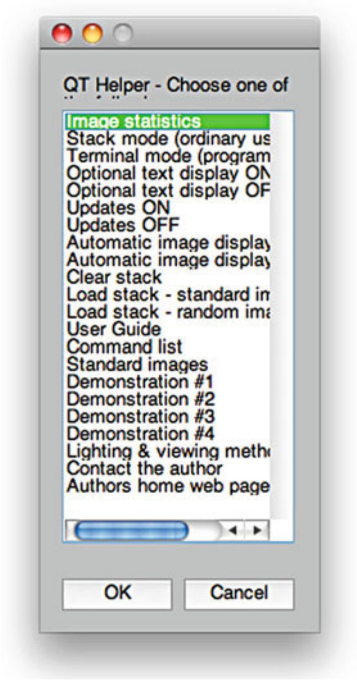


Bottom-right: after *pci*('*heq*')



41.8 aid

Format	*aid*
Arguments	None
Output values	None
Image type	Any
Function	Assistance for the user
Switch images	No
Remarks	Also see *hlp*, *doc*, *img*, *author*



QT Helper - Choose one of

Image statistics
Stack mode (ordinary us
Terminal mode (program
Optional text display ON
Optional text display OF
Updates ON
Updates OFF
Automatic image display
Automatic image display
Clear stack
Load stack - standard im
Load stack - random ima
User Guide
Command list
Standard images
Demonstration #1
Demonstration #2
Demonstration #3
Demonstration #4
Lighting & viewing meth
Contact the author
Authors home web page

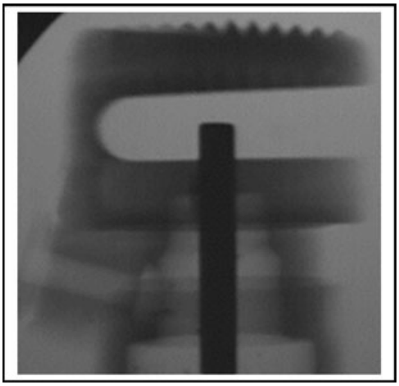
41.9 air

Format	*air*(*a*, *b*)
Arguments	a Lower intensity limit b Upper intensity limit

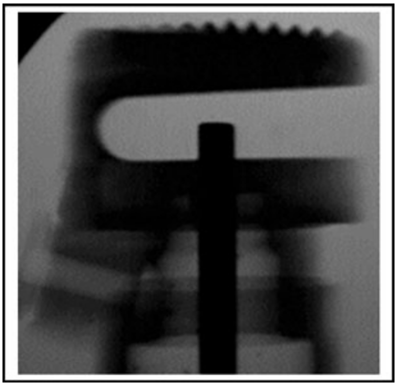
Defaults	No arguments
	a = 13; b = 242 (approximately 5% and 95% of white level)
	Single argument
	b = 255 − a

Output values	None
Image type	Grey
Type	Monadic
Function	Apply a linear stretching of the intensity scale between the limits [a, b]; pixels with intensities outside this range are set to 0 or 255
Definition	$C \Leftarrow 255*(A - a)/(b - a)$ with hard limits placed at 0 and 255
Switch images	Yes
Typical uses	Increasing contrast
Remarks	Also see *blo*, *din*, *enc*, *hin* and *heq*
Illustration	
Input image	Aerosol spray assembly
File number	111
Command	*air*(*50*, *200*)

Left: original



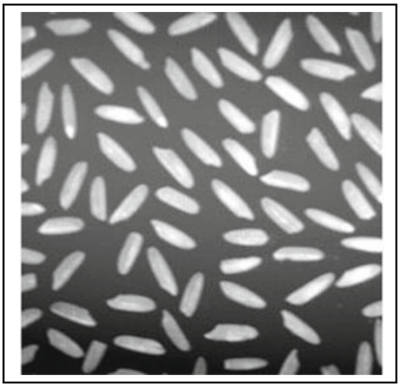
Right: after processing



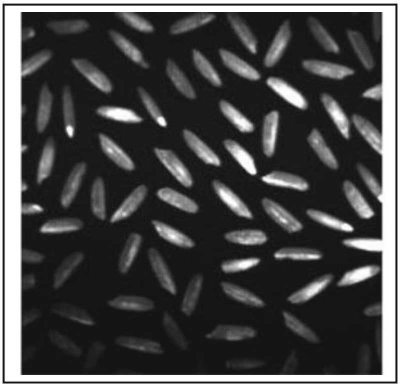
41.10 alg

Format	*alg*
Arguments	None
Defaults	None
Output values	None
Image type	Grey
Type	Monadic
Function	Antilogarithm (exponential) of intensities in the current image
Definition	$C \Leftarrow 255 * \text{antilog}(A)/\text{antilog}(255)$
Switch images	Yes
Typical uses	Used to improve contrast within bright regions of the image
Remarks	Also see *nlg*, *sqr* and *sqt*
Illustration	
Input image	Rice grains after contrast enhancement (*enc*)
File number	3
Command	*alg*

Left: original



Right: after processing

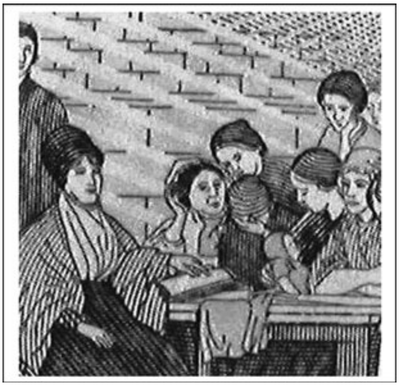


41.11 ani

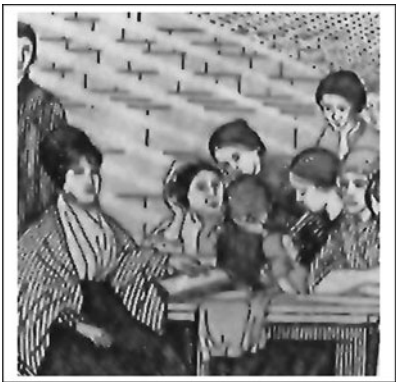
Format	*ani*(*a*, *b*, *c*)	
Arguments	*a*	*Number of iterations* (*niter in anisodiff*)
	b	Conduction coefficient (*kappa* in anisodiff)
	c	Select the Perona Malik diffusion equation to be used
Defaults	*a*	10
	b	50
	c	1 (Perona-Malik diffusion equation No. 1)
Output values	None	
Image type	Grey	
Type	Image filter	
Function	Apply the Anisotropic diffusion filter defined by P. Perona and J. Malik	
Switch images	Yes	
Reference	Perona P, Malik J (July 1990) Scale-space and edge detection using ansotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(7):629–639	
Remarks	With minor modifications, the M-file *anisodiff* was written by Peter Kovesi, School of Computer Science and Software Engineering, The University of Western Australia, Email: *pk@csse.uwa.edu.au*. URL *http://www.csse.uwa.edu.au*	

Illustration	
Input image	Currency note (detail)
File number	211
Command	*ani*

Left: original



Right: after processing



41.12 asi/mxs

Format *asi*(*a*, *b*, *c*)
mxs(*a*, *b*, *c*, *d*)

Arguments *a* Number of images acquired
b Camera
c Time delay
d Maximum/minimum selector (*mxs* only)

Defaults *a* None
b 6
c 5 (seconds)
d 1 (*mxs* computes the maximum over *a* images)

Output values None

Image type Grey

Type Image acquisition

Function asi Average *a* images from camera *b*. Time interval between samples: *c* seconds
mxs ($d = 0$) Minimum of *a* images from camera *b*. Time interval: *c* seconds
mxs ($d = 1$) Maximum of *a* images from camera *b*. Time interval: *c* seconds

Switch images Yes

Typical uses Eliminating camera noise (static scenes) Obtaining a back-ground image, clear of vehicles/ people when analysing traffic

Remarks Imagine a static camera viewing a clock with a white face, black numerals and black hands. For suitable values of *a* and *c*, *mxs*(*a*, *b*, *c*, *0*) derives an image in which the hands have been eliminated; the clock appears as if there were no hands. This image can then be subtracted from a ‘snap-shot’ image of the clock, to remove back-ground details of the face, and provide a clear view of the hands

41.13 author/bgb

Format	*author*
Arguments	None
Output values	None
Image type	Not relevant
Type	Information
Function	Learn about the author of QT
Switch images	No

41.14 avg

Format	*a* = *avg*
Arguments	None
Defaults	Not applicable
Output values	*a* Average intensity
Image type	Grey-scale
Type	Grey-scale image measurement
Function	Find the average intensity over the whole image
Switch images	No
Typical uses	Choosing a threshold parameter
Remarks	Also see *dev*

41.15 bay

See *chu*.

41.16 bbx

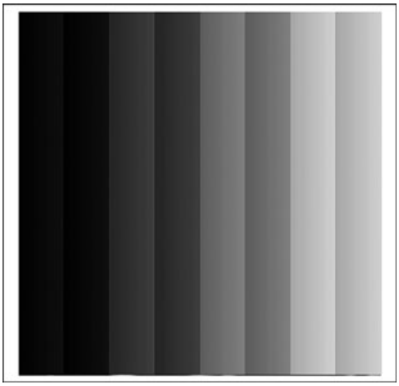
Format	$[a, b, c, d] = bbx$	
Arguments	None	
Defaults	Not applicable	
Output values	a	Minimum X-coordinate of the first blob found
	b	Minimum Y-coordinate of the first blob found
	c	Maximum X-coordinate of the first blob found
	d	Maximum Y-coordinate of the first blob found
Image type	Binary	
Type	Binary image measurement	
Function	Bounding box	
Definition	Perform a raster scan to find one object. Scan direction: top to bottom; left to right. Find the minimum and maximum X and Y coordinates	
Switch images	No	
Typical uses	Analysing/locating blob like objects	
Remarks	Also see *mar*	

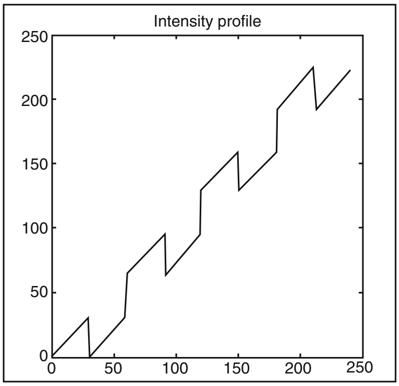
41.17 bcl/bcp/bst

Format	*bcl*(*a*)/*bcp*(*a*)/*bst*(*a*)
Arguments	a Index number of bit to be cleared/complemented/set ($1 \leq a \leq 8$)
Defaults	None
Output values	None
Image type	Grey
Type	Monadic
Function	*bcl*: For each pixel, set bit number *a* to 0 in the 8-bit binary representation of its intensity *bcp*: For each pixel, complement bit number *a* in the 8-bit binary representation of its intensity *bst*: For each pixel, set bit number *a* to 1 in the 8-bit binary representation of its intensity
Switch images	Yes
Remarks	Bit-set and bit-complement operations can be implemented by changing line 5 of bcl as follows: Bit set (bst): *current = bitset(current, a, 1)*; Bit complement (bcp): *current = bitcmp(current, a)*; These functions have only limited use. Also see *sca*
Illustration	
Input image	Intensity wedge (*wgx*)
Command	*bcl*(6), *plt*

Left: after processing

Right: mapping function





41.18 bcp

See *bcl.*

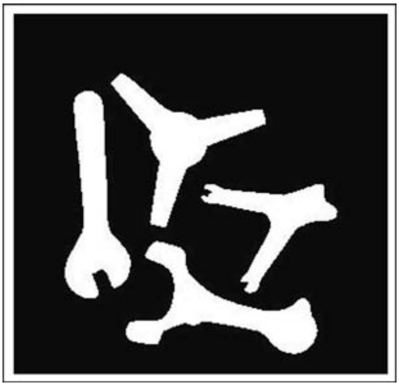
41.19 bed

Format	*bed*(*a*)
Arguments	a Connectivity (a = 4 or 8)
Defaults	a 8 (8-connectivity)
Output values	None
Image type	Binary
Type	Blob transform
Function	Binary edge detector
Definition	White pixels that have at least one black *a*-neighbour (*a* = 4 or 8) are set to white. All other pixels are set to black
Switch images	Yes
Typical uses	Inspecting object silhouettes
Remarks	The output is a binary image containing a white arc that traces the outer edge of the object. The edges of lakes (holes) are similarly defined. All blobs are processed in this way

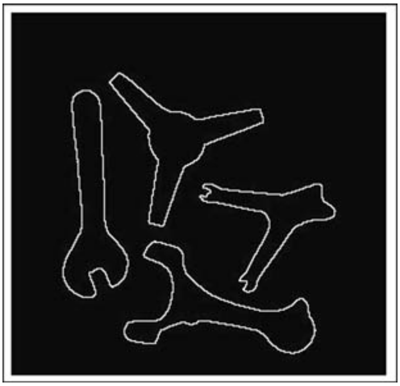
Illustration

Input image	Metal components
File number	46
Command	*bed*

Left: original image



Right: *bed*



41.20 bgb

See *author.*

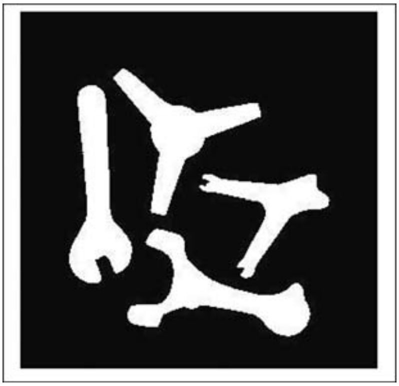
41.21 big

Format	*big*(*a*)	
Arguments	*a*	Rank size of blob to be isolated
Defaults	a	1
Output values	None	
Image type	Binary	
Type	Blob transform	
Function	Isolate the *a*th biggest blob in a binary image	
Switch images	Yes	
Typical uses	Analysing scenes with several large blobs and possibly many smaller ones	
Remarks	*big* will only work properly if there are fewer than 255 white blobs in the image. Otherwise *ndo* will saturate, combining many blobs into a single label. Also see *ndo* and *kgr*	

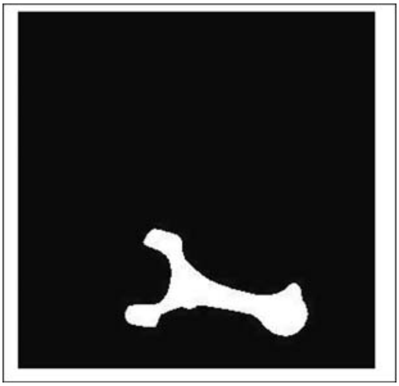
Illustration

Input image	Metal objects
File number	46
Command	*thr*, *big*(*1*)

Left: original



Right: after processing



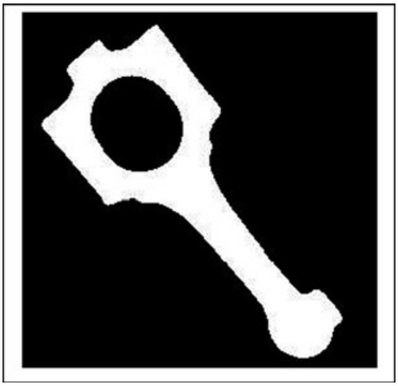
41.22 blf/lak

Format	*blf*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob transform
Function	Fill all holes (lakes) in the current image
Switch images	Yes
Typical uses	Blob shape analysis
Remarks	A lake is an 8-connected black region that is totally enclosed within an 8-connected white region. The function *lak* isolates the lakes. Also see *chu*

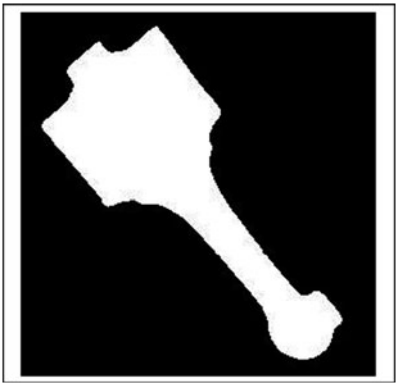
Illustration

Input image	Con-rod
File number	47

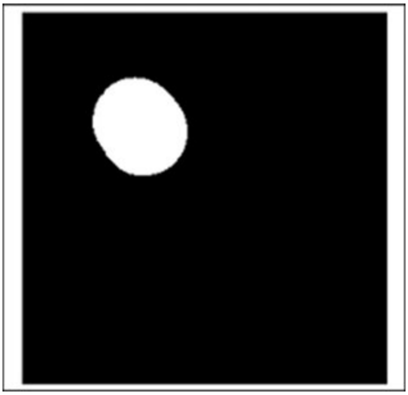
Top-left: original



Top-right: after *blf*



Bottom: after *lak*



41.23 blo

Format	*blo*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Monadic
Function	Expand the central part of the intensity range; level 64 is mapped to 0 and level 192 to 255. Hard limiting is applied at 0 and 255
Definition	C ⇐ 2∗(A − 64) with hard limits placed at 0 and 255
Switch images	Yes
Typical uses	Increasing contrast after high-pass filtering, where intensities tend to be clustered around mid-grey (level 128)
Remarks	Functionally equivalent to *air*(*64*, *192*)

41.24 blp/brp/lbp/ltp/rbp/rtp/tlp/trp

Format	[*x*, *y*] = *blp* [*x*, *y*] = *brp* Etc.
Arguments	None
Output values	*x* X-coordinate of the point found *y* X-coordinate of the point found
Image type	Binary
Type	Image analysis
Function	Find the following points: *blp* – Bottom-left point *brp* – Bottom-right point *lbp* – Left-bottom point *ltp* – Left-top point *rbp* – Right-bottom point *rtp* – Right-top point *tlp* – Top-left point *trp* – Top-right point
Switch images	No
Remarks	Also see *mar*

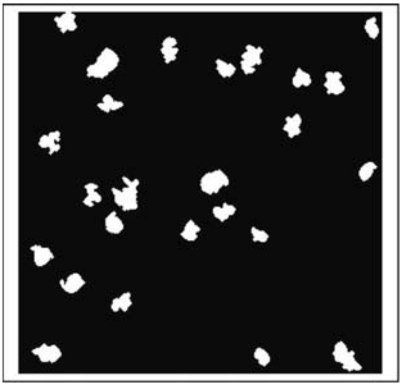
41.25 bnb

Format	*function* [*u*, *v*, *r*] = *bnb*	
Arguments	None	
Output values	u	X-coordinates of a white point in the bridge
	v	Y-coordinates of a white point in the bridge
	r	A measure of the size of the gap that has been bridged
Image type	Binary	
Function	Find the shortest bridge between nearby blobs. The position of the centre of the bridge ([u, v]) is returned, together with the approximate size of the gap between the closest blobs. The centre of the bridge is identified in the output image by a cross	
Switch images	Yes	
Typical uses	Analysing binary images	
Remarks	A C-shaped blob will be detected first, if the gap is smaller than the distance between separate blobs	

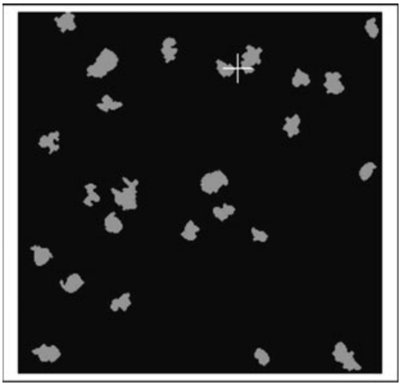
Illustration

Input image	*spn*, *caf*, *enc*, *thr*, *kgr*(*50*), *dil*, *blf* (Collection of 'random' blobs)
Commands	*bnb*, *adi*, *mxi*

Left: original image



Right: result of processing



41.26 bpa/cab/cxa/eab/ecy/eqd/ext/far/oab/mab/sly

Format	$b = bpa(a)$
Arguments	*a* Measurement required (for options, see the table below)
Defaults	Not applicable
Output values	*b* Vector of measurements
Image type	Binary
Type	Binary image measurement
Function	Calculate shape/size parameters of all blobs in a binary image
Definition	The measurement specified by the input parameter a is derived for each blob in the input image. For the measurement options available, see the table below
Switch images	No
Typical uses	Characterisation of multi-blob images, such as those derived from granular materials Analysis of the surface properties of spot-like features on a web or sheet material
Remarks	When the command $q = bpa$(‘*Area*’) is executed, a vector of length *N* is generated that measures the area of each of the (N) blobs in the image Similarly, in every other case, except when the input parameter *a* is equal to ‘*Centroid*’, the output vector *b* contains one measurement value for each blob in the image. For the special case *a* = ‘*Centroid*’, the output vector is of the form $\{X_1, Y_1, X_2, Y_2, X_3, Y_3, \ldots\}$, where $[X_i, Y_i]$ is the centroid of the *i*th blob. Notice that *bpa* is similar to the auxiliary function *blob_parameters*, which is used by the blob-analysis functions *lmi* and *cgr*. However, the latter measure only one blob. Additional functions that are easier to use can be defined in terms of *bpa*:

Name	**Function**	**Implementation**
cab	Centroids	*bpa*(‘*Centroid*’)
cxa	Convex areas	*bpa*(‘*ConvexArea*’)
eab	Euler numbers	*bpa*(‘*EulerNumber*’)
ecy	Eccentricities	*bpa*(‘*Eccentricity*’)
eqd	Equivalent diameters	*bpa*(‘*EquivDiameter*’)
ext	Extent scores	*bpa*(‘*Extent*’)
far	Areas after lakes have been filled	*bpa*(‘*FilledArea*’)
oab	Orientations	*bpa*(‘*Orientation*’)
sly	Solidities	*bpa*(‘*Solidity*’)

The function *mab* measures blob parameters for all blobs. The result is an array with 13 columns and *N* rows, where *N* is the number of blobs in the image

Table: Permitted values for the input parameter, *a*, for command *bpa*

Argument	Measurement (These are calculated for each blob separately)
Area	Number of white pixels
Centroid	Coordinates of the centroid. (Odd-numbered elements of the output indicate the horizontal coordinates; even-numbered elements indicate the vertical coordinates)
ConvexArea	Number of pixels in the convex hull
Eccentricity	Eccentricity of the ellipse that has the same second-moments as the blob. (Eccentricity is the ratio of the distance between the foci of the ellipse and the length of its major axis. It lies within the range [0, 1].)
EulerNumber	One minus the number of lakes ('holes')
Extent	Proportion of the pixels in the minimum bounding box that are also in the blob. (That is, the area of the blob divided by the area of its bounding box)
FilledArea	Number of on pixels after lakes ('holes') have been filled
MajorAxisLength	The length of the major axis of the ellipse that has the same values for the second moments as the blob
MinorAxisLength	The length of the minor axis of this ellipse
Orientation	Orientation of the principal axis (That is the angle between the horizontal axis and the major axis of this ellipse. Measured in degrees)
Solidity	Proportion of the pixels in the convex hull that are also in the blob (This is equivalent to Area/ConvexArea)

Typical output when *bpa* is applied to the image of four metal objects (File number 46. See *ndo.*)

***u* = *bpa*('*Area*')**

```
        u = 4332    4215     4476     2926
```

***v* = *bpa*('*ConvexArea*')**

```
        v = 6337    8455     9535     6006
```

***w* = *bpa*('*Centroid*')**

```
       w = 64.9162 154.9234 133.0643 101.8769 167.1032 238.0487 210.2584 165.8226
```

The measurements are listed in the following order:
Spanner (*wrench*) *3-pointed star* *Con-rod* *Y-shaped part*

Function *mab* (same image)

Area	Convex area	Ecc.	Eul no.	Equiv. diam.	Ext.	Fill. area	Maj. axis	Min axis	Solid.	X centr.	Y centr.	Orient.
4332	6286	1.0	1	74.3	0.5	4332	192.5	041.0	0.7	64.9	154.9	−87.6
4215	8348	0.2	1	73.3	0.3	4215	107.6	106.4	0.5	133.1	101.9	−59.4
4476	9471	0.9	1	75.5	0.3	4476	180.9	071.3	0.5	167.1	238.0	−14.4
2926	5931	0.5	1	61.0	0.3	2926	099.5	083.3	0.5	210.3	165.8	20.0

41.27 bst

See *bcl.*

41.28 cab

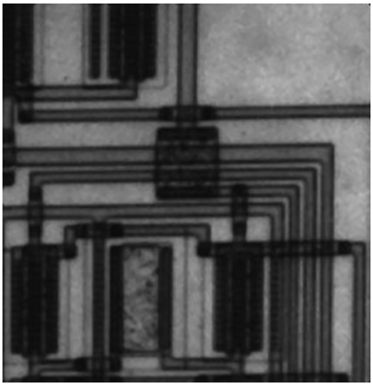
See *bpa.*

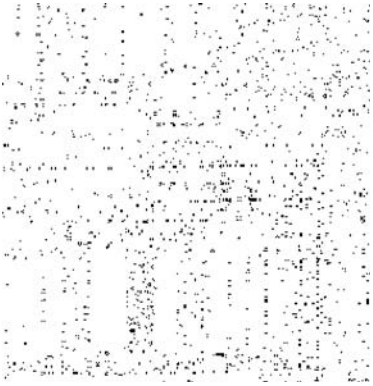
41.29 cae

Format	$b = cae(a)$
Arguments	a Allowed value for the square of intensity difference
Defaults	a 0
Output values	b Number of pixels with similar intensity values
Image type	Grey
Type	Dyadic
Function	Identify and count the points where the square of the difference of intensities in the current and alternate images is less than or equal to a given parameter, a
Definition	$C(i, j) = 255$, if (i, j)th $[c(i, j) - a(i, j)]$^2 $\leq a$ $C(i, j) = 0$, otherwise b is the number of points for which the condition $[c(i, j) - a(i, j)]$^2 $\leq a$ applies
Switch images	Yes
Typical uses	Comparing similar images, identifying their differences
Remarks	If the input parameter (a) is zero, *cae* finds where the images are identical
Illustration	
Input images	Circuit and the same image after applying *mdf* (median filter)
File number	*136*
Command	*mdf*, *cae*(5)

Top-left: original

Top-right: after processing (Black pixels indicate where *mdf* has made a significant change)





41.30 caf

Format *caf*(*a*)

Arguments *a* Filter kernel size

Defaults *a* 7

Output values None

Image type Grey

Type Linear filter

Function Averaging filter within a circular window of diameter *(2a + 1)* pixels

Description A square array of size (2*a* + 1).(2*a* + 1) is computed first. For *a* = 5, this array has the following form

0	0	0	*	*	*	*	0	0	0
0	0	*	*	*	*	*	*	0	0
0	*	*	*	*	*	*	*	*	0
*	*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	*	*	*
0	*	*	*	*	*	*	*	*	0
0	0	*	*	*	*	*	*	0	0
0	0	0	*	*	*	*	0	0	0

where '$*$' represents 255. In general, the array generated by *caf*(*a*) contains a set of non-zero elements arranged to form an approximation to circular disc of radius *a*. This array defines the weight matrix used in a linear convolution filter. Thus, all pixels lying within a disc of radius *a* and centred on pixel (*i*, *j*) are averaged to form the new value for the intensity at (*i*, *j*). The MATLAB code refers to the operator *dcd*(*a*), which draws a solid circular disc of radius *a*, located at the centre of an image of size $a*a$

Switch images Yes

Typical uses Reducing the effects of noise

As part of a high-pass filter:

caf(*m*), *sub*

(*caf* performs low-pass filtering, so subtracting its output from the original image accentuates high spatial frequencies)

caf is also used within a band-pass filter, implemented thus:

caf(*m*), *wri*, *swi*, *caf*(*n*), *rei*, *sub*

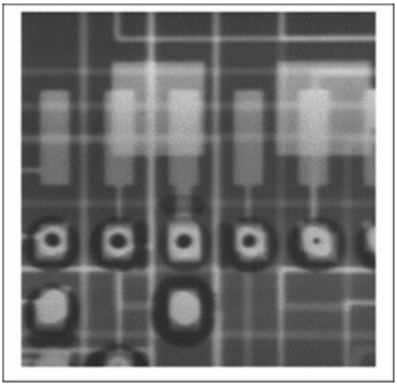
where *m* and *n* are positive integers,

Remarks *caf* performs isotropic low-pass filtering; horizontal and vertical edges are blurred by equal amounts. The effect becomes more pronounced as *a* is increased. Also see *raf*, *lpf*

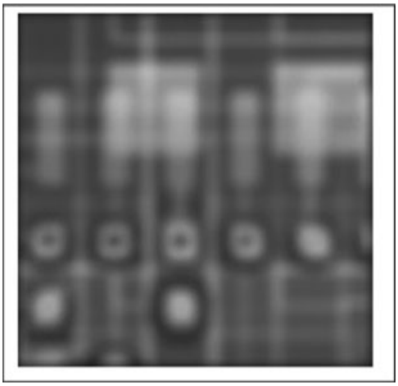
Illustration

Input image	Printed circuit board
File number	135
Command	*caf*(*11*)

Left: original



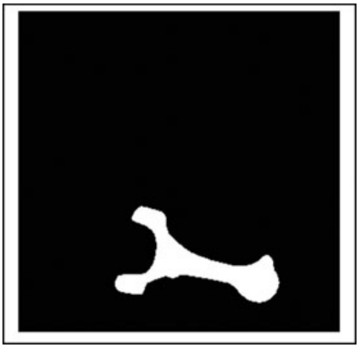
Right: after processing



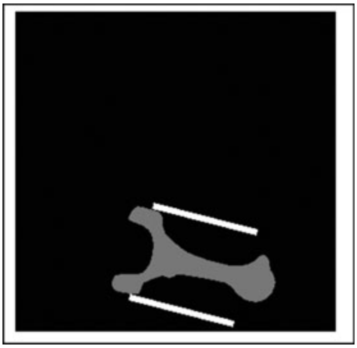
41.31 caliper

Format	p = *caliper*(*w*, *n*)	
Arguments	w	Width of the caliper jaws
	n	maximum opening of the caliper jaws
Defaults	w	50
	n	100
Output values	p	Object diameter
Image type	Binary	
Type	Binary image measurement	
Function	Virtual caliper gauge	
Switch images	Yes	
Typical uses	Measuring object diameter	
Remarks		
Illustration		
Input image	*Conrod* (*one of the objects in image 46*)	
File number	46	
Command	*rni*(*46*), *thr*, *big*, *caliper*(*25*, *100*), *adi*, *mxi*	

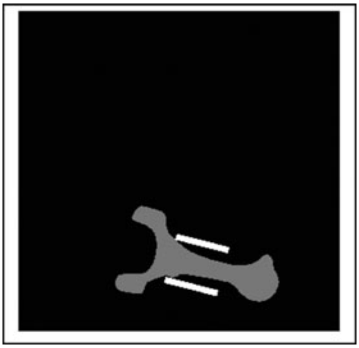
Top-left: original image



Top-right: after processing



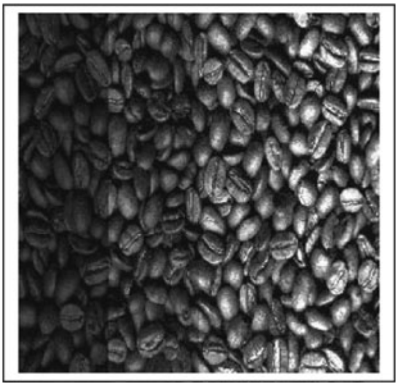
Bottom-left: shorter caliper jaws



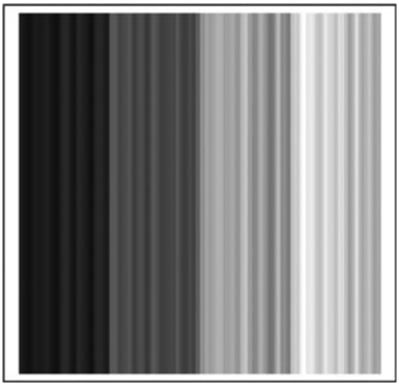
41.32 cav

Format	*cav*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Image measurement
Function	Average for each column in the image
Definition	The *i*th pixel in each row is the average (mean) of the intensities in column *i*
Switch images	Yes
Typical uses	Texture analysis
Remarks	Use *yxt*, *cav*, *yxt* to calculate the average for each row
Illustration	
Input image	Coffee beans (image 4) modified by multiplying by an intensity 'stair-case'.
Image generation	*wgx*, *sca*(*2*), *acn*(*64*), *rni*(*4*), *mul*, *enc*
Command	*cav*

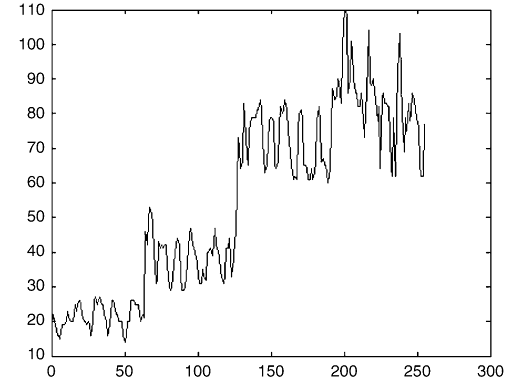
Top-left: original



Top-right: *cav*



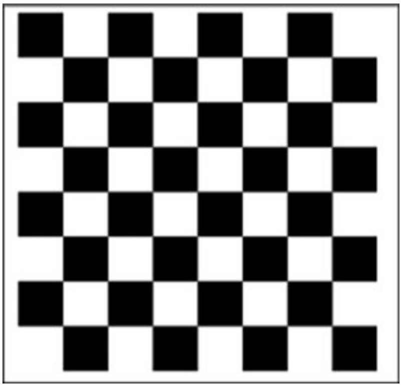
Bottom-left: *cav*, *plt*



41.33 cbd

Format	*cbd*
Arguments	None
Defaults	Not applicable
Output values	None
Type	Drawing and image generation
Switch	Yes
Function	Generate a simple (8∗8) ‘chessboard’ pattern
Switch images	Yes
Typical uses	Testing various image processing functions, such as filters
Remarks	Also see *grd*
Illustration	
Commands	*cbd*

Below: chessboard



41.34 cbl

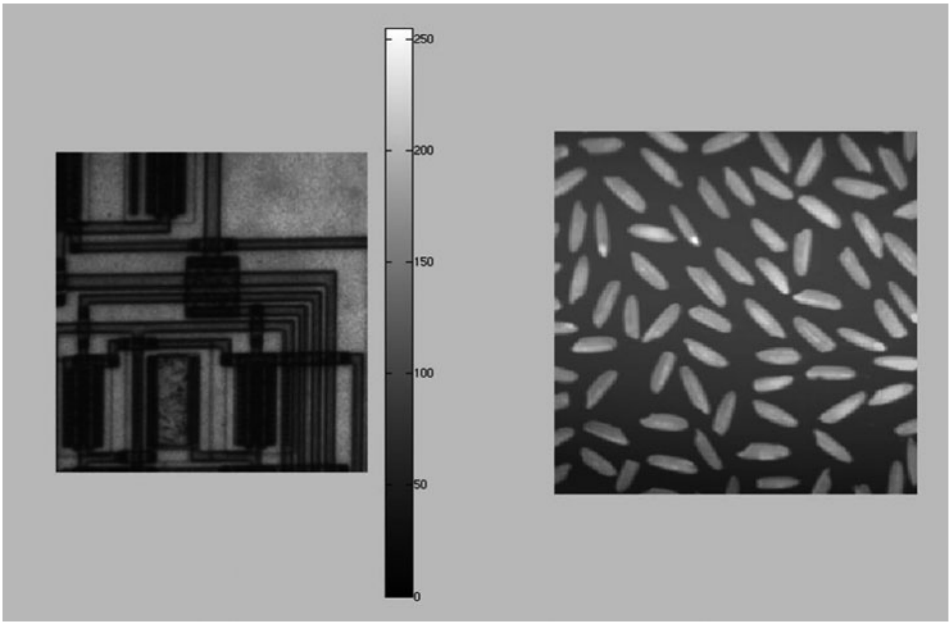
Format	$b = cbl(a)$	
Arguments	a	Connectivity (4- or 8-connected Hence, $a \in \{4, 8\}$)
Defaults	a	8
Output values	b	Number of blobs
Image type	Binary	
Type	Binary image measurement	
Function	Count the number of a-connected ($a \in \{4, 8\}$) blobs in a binary image	
Switch images	No	
Typical uses	Counting blobs. Computing average blob area (using *cwp*, as well)	
Remarks	Notice that *cbl* computes the number of blobs directly, while *eul* computes the Euler number (i.e., the number of blobs minus the number of holes)	

41.35 cbr

Format	*cbr*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Any
Type	Miscellaneous
Function	Draw a colour bar beside the current image
Switch images	No
Typical uses	Monitor set-up. Also, aids understanding of image structure
Remarks	Also see *pxv*

Illustration

Screen shot of *Figure No. 1*. The current image is always displayed on the left and the alternate image on the right



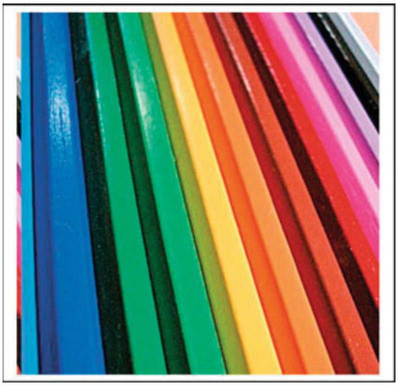
41.36 ccv

Format	[*r*, *g*, *b*] = *ccv*(*x*, *y*)	
Arguments	*x*	X-address of the pixel to be accessed
	y	Y-address of the pixel to be accessed
Defaults	None	
Output values	*r*	R-channel vale value at point [*x*, *y*]
	g	G-channel vale value at point [*x*, *y*]
	b	B-channel vale value at point [*x*, *y*]
Image type	Colour	
Type	Accessing individual pixels	
Function	Find the RGB values [*r*, *g*, *b*] at a given point [*x*, *y*] in the current image	
Switch images	No	
Typical uses	Analysing colour images. Designing colour filters	
Remarks	Use *ccv*, rather than accessing individual pixels of the current image directly, since it follows the QT convention that the x-address precedes the y-address. Also see *civ* and *scv*	

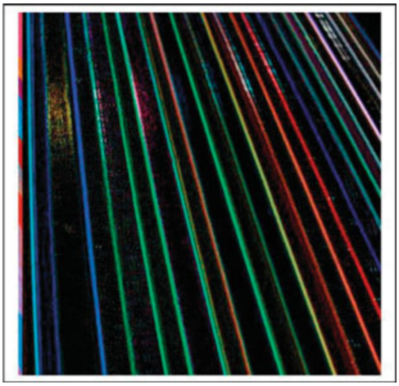
41.37 cdf

Format	*cdf*
Arguments	None
Output values	None
Image type	Colour
Type	Colour image processing
Function	Colour difference between two images. The RGB channels are processed separately. First, the two images are subtracted. Then, the ‘absolute value’ of the difference is calculated
Switch images	Yes
Typical uses	Filtering and edge detection in colour images
Remarks	See *abv* for an explanation of the term ‘absolute value’.
Illustration	
Input image	Pencils and the same image shifted horizontally by 5 pixels
File number	270
Command	*rci(270), pci(‘psw(5, 0)’), cdf, pci(‘enc’)*

Left: original



Right: after processing



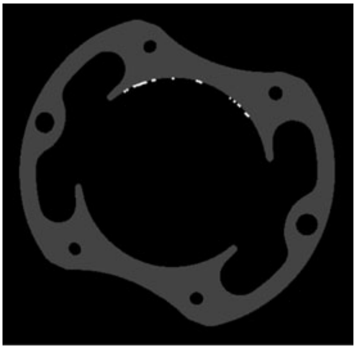
41.38 ceb

Format	*ceb*(*a*)
Arguments	*a* Size limit (number of pixels)
Default	*a* 5000
Output values	None
Image type	Binary edge image
Type	Binary image analysis
Function	Fit a circle to the set of edge points enclosing a blob, using the least-squares algorithm
Switch images	Yes
Typical uses	Analysing a binary image with a single continuous edge contour
Remarks	*fcl* cannot accommodate a closed edge contour, whereas *ceb* can. Also see *dec*

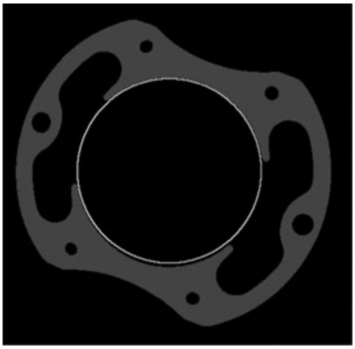
Illustration

Input Image	*Binary image derived from image 174* (*stator for electric motor*), *followed by manual selection of part of the edge*
Commands	*ceb*

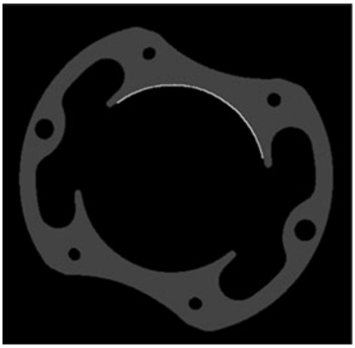
Top-left: white points, starting data for ceb



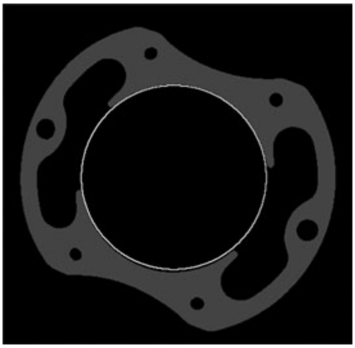
Top-right: output from *ceb* compared to original



Bottom-left: as above, but more white points



Bottom-*right*: circle is slightly better fit



41.39 ced/led/ped/zed

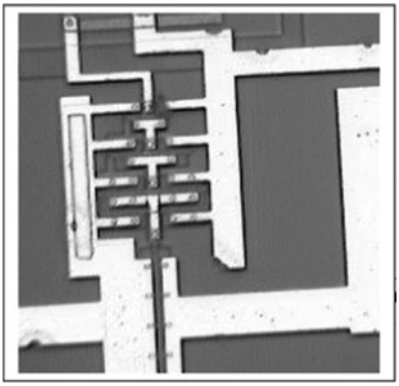
Format	b = *ced*(*a*)
	b = *led*(*a*)
	b = *ped*(*a*)
	b = *zed*(*a*)	

Arguments	*a*	Intensity threshold
Defaults	*a*	Calculated automatically
Output values	Calculated intensity threshold	
Image type	Grey	
Type	Edge detector	
Function	Canny/Laplacian of Gaussian/Prewitt/and Zero-cross edge detector functions, with optional automatic thresholding	
Switch images	Yes	
Typical uses	Highlighting edges (i.e., regions of large intensity gradient)	
Remarks	The following code is representative for all four functions, simply substitute ‘*led*’, ‘*ped*’ or ‘*zed*’ in lines 6 and ‘9’. These are more sophisticated functions than those listed under *edd* and *sed*, since they automatically perform thresholding, thereby generating a binary image. This can be a disadvantage, since the user effectively loses control of the processing and the speed of operation is reduced. All four functions produce similar results. For the ‘log’ and ‘zerocross’ methods, if the input parameter, *a*, is zero, the output image contains closed contours	

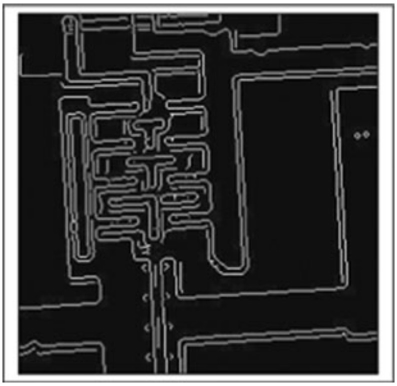
Illustration

Input image	Thick film circuit
File number	137
Command	*ced*

Left: original



Right: after processing



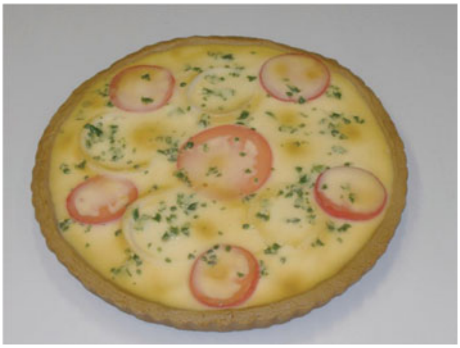
41.40 cen/chq/cgm

Format	*cen*	
	chq	
	cgm(*a*, *b*, *c*)	
Arguments	*a*	R-channel gamma adjustment (*cgm* only)
	b	G-channel gamma adjustment (*cgm* only)
	c	B-channel gamma adjustment (*cgm* only)
Defaults	None	
Output values	None	
Image type	Colour	
Type	Colour image processing	
Function	*Equivalent to cen, chq, cgm*	
Switch images	Yes	
Typical uses	Contrast adjustment for easier viewing/processing	
	Compensating for lighting variations	
Remarks	Also see *pci*	

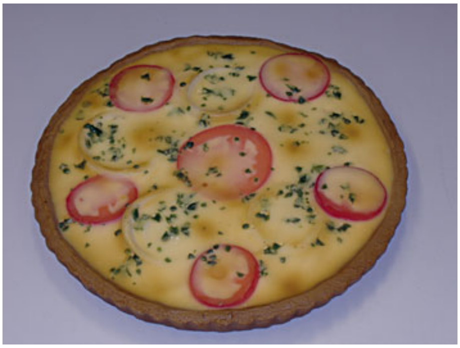
Illustration

Input image	Plastic replica of a pizza
File number	219
Commands	*cen, chq, cgm(0.7,1.2,1.2)*

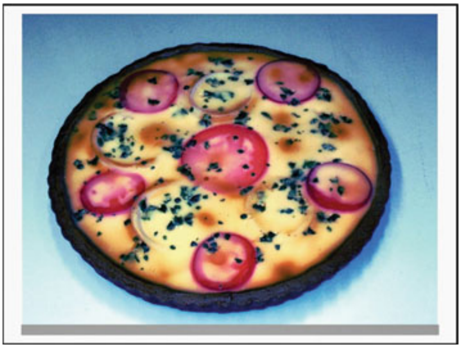
Top-left: original



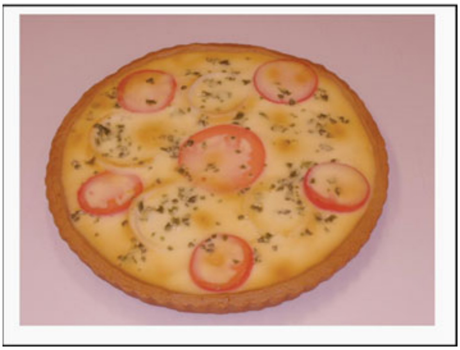
Top-right: after processing, *cen*



Bottom-left: after processing, *chq*



Bottom-right: after processing, *cgm*(*0.7*, *1.2*, *1.2*)



41.41 ceo

Format	*ceo*(*a*)
Arguments	*a* Select the edge detection operator to be applied
Defaults	*a* 0
Output values	None
Image type	Colour
Type	Colour image processing
Function	Apply the edge detector selected by the user to each of the RGB channels separately
Switch images	Yes
Remarks	The input parameter, *a*, selects the function applied:

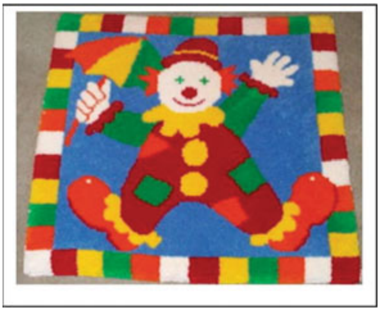
Input parameter, *a*	Edge detector
0 (default)	Dilate, subtract & enhance contrast
1	Sobel
2	Roberts
3	Prewit
4	Canny
Other	Dilate & subtract. Do not enhance contract

Also see *pci*

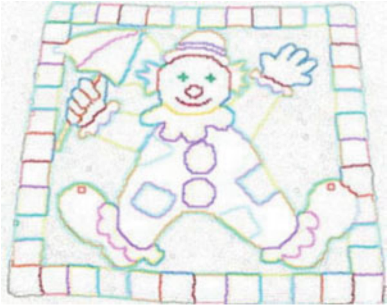
Illustration

Input image	rug
File number	354
Command	*ceo*(*0*)

Left: original



Right: after processing

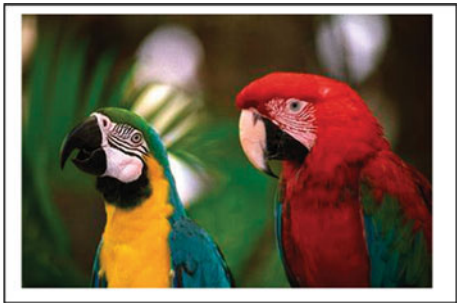


41.42 cga

Format	*cga*(*a*, *b*, *c*)	
Arguments	*a*	Gamma adjustment made to the *R* channel
	b	Gamma adjustment made to the *G* channel
	c	Gamma adjustment made to the *B* channel
Defaults	*b*	*a*
	c	*a*
Output values	None	
Image type	RGB colour image	
Type	Image enhancement	
Function	*3 arguments supplied*: Apply a (different) gamma adjustment to each of the RGB channels separately *1 argument supplied*: Apply the same gamma adjustment to each of the RGB channels	
Typical uses	Image enhancement for visual examination of a scene with low visibility features	
Remarks	Also see *gma*	
Illustration		
Input image	Birds	
File number	144	
Command	*cga*(*1.5*, *1.6*, *1.7*)	

Left: original

Right: after processing



41.43 cgp

Format	$b = cgp(a)$
Arguments	a Grey level, $a \in [0, 255]$.
Defaults	None
Output values	b Number of pixels with intensity a
Image type	Grey-scale
Type	Grey-scale image measurement
Function	Count the number of pixels with intensity a
Switch images	No
Typical uses	Used in conjunction with *ndo*, *cgp* is able to identify the biggest blob, second biggest blob, etc. in a binary image
Remarks	Notice that *cwp* is equivalent to *cgp*(*255*). Also see *hpi*

41.44 cgr

Format	$[a, b] = cgr$
Arguments	None
Defaults	Not applicable
Output values	a X-coordinate of the centroid
	b Y-coordinate of the centroid
Image type	Binary
Type	Binary image measurement
Function	Calculate the coordinates of the geometric centroid of a blob-like figure
Switch images	No
Typical uses	Finding the position of a blob
Remarks	If there is more than one blob, *cgr* computes the centroid of just one: the first blob that is encountered during a top-to-bottom, left-to-right raster scan

41.45 cgr_all/lmi_all/npo_all

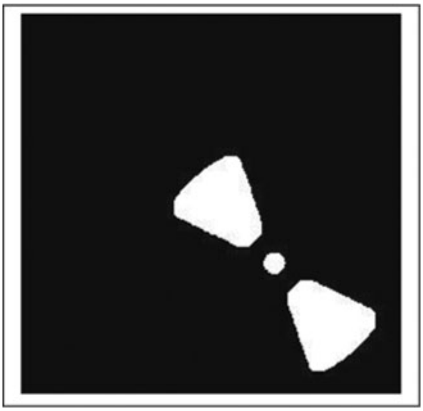
Format	[*x*, *y*] = *cgr_all* [*x*, *y*, *z*] = *lmi_all* *npo_all*	
Arguments	None	
Output values	*x*	X-coordinate, all blobs considered together
	y	Y-coordinate, all blobs considered together
	z	Axis of minimum second moment, all blobs considered together
Image type	Binary	
Type	Binary image analysis/manipulation	
Function	The following calculations/operations are performed taking all non-black pixels into account:	
	cgr_all	Centroid
	lmi_all	Centroid and orientation of axis of minimum second moment
	nxy_all	Normalise, so centroid is at the centre of the image
	npo_all	As *nxy_all* and normalise axis of minimum second moment
Switch images	*cgr_all*	No
	lmi_all	No
	nxy_all	Yes
	npo_all	Yes
Remarks	In the illustration below (at bottom-right), three aligned image features (i.e., small circular hole and two wedge-shaped holes) are used to calculate the parameters needed to normalise the position and orientation of a gear These functions all rely on the calculation of moments, which is performed by *moment*	

Illustration

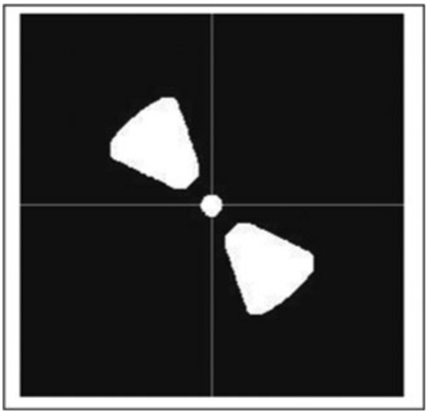
Input image Gear (image 210) after thresholding

Command *thr*(*115*)

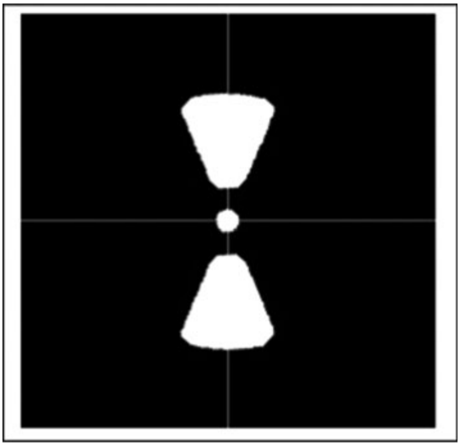
Top-left: 3 features, aligned along a straight line, were selected (Program details are an unnecessary distraction for this illustration)



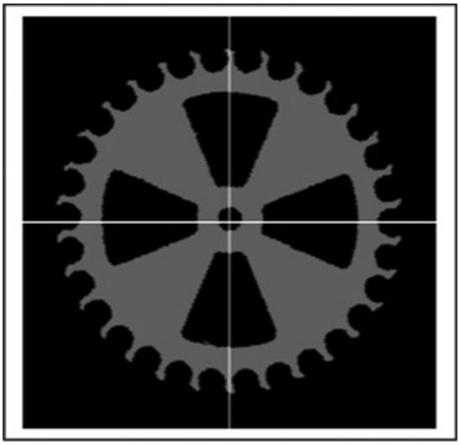
Top-right: *nxy_all* fixes the position of the centroid of the three blobs, but not the orientation



Bottom-left: *npo_all* fixes the position and orientation of the three blobs taken together



Bottom-right: normalised position and orientation of the gear, based on values calculated by *lmi_all* operating on the image at *top-left*



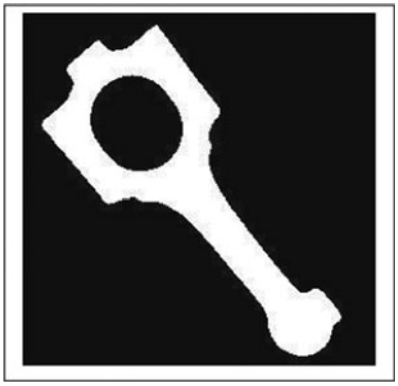
41.46 chu/cvd/bay

Format	*chu*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob transform
Function	Convex hull of a single blob
Switch images	Yes
Typical uses	The ratio of the area of a blob to that of its convex hull is a useful shape measure and also forms the basis for a simple test of convexity Convex hull forms the basis of an hierarchical shape description known as the Concavity Tree representation
Remarks	The convex hull of a blob is the area that would be enclosed within a rubber band stretched around it. The convex hull encloses two types of concavity: indentations ('bays') and holes ('lakes') Notice that the function implemented by *chu* calculates the convex hull of only one connected binary object at a time (i.e., the one that is found first by a raster scan running from top to bottom and left to right.) The function *cvd* calculates the convex deficiency. This is the difference between the convex hull and the original blob. The function *bay* isolates the bays (indentations)

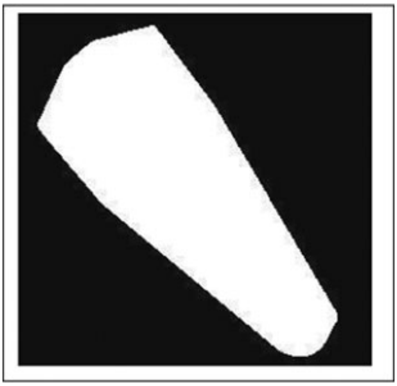
Illustration

Input image	Con-rod
File number	47
Command	*thr*, *chu* (N.B. *thr* avoids the artifacts created by storing a binary image in JPEG format. Although image file 47 is a TIF file now, it was originally generated and stored in JPEG format)

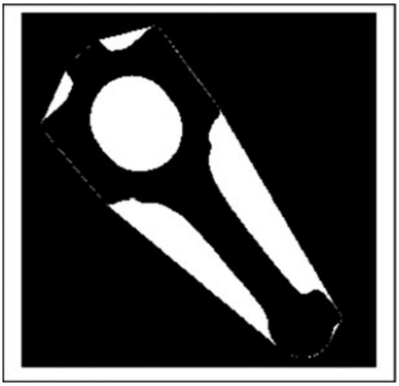
Top-left: original



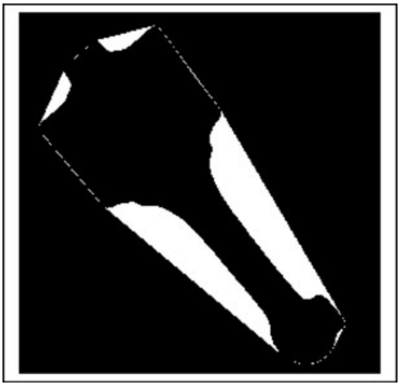
Top-right: after chu



Bottom-left: after *cvd*



Bottom-right: after *bay*



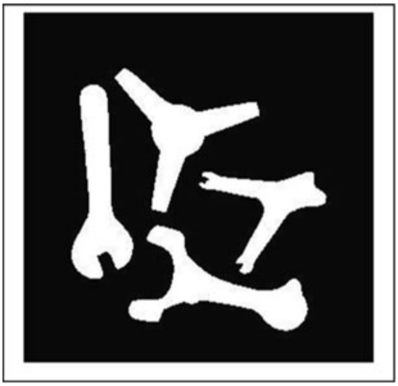
41.47 chu_all

Format	*chu*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob transform
Function	Convex hull of a set of separate blobs
Switch images	Yes
Typical uses	Analysing complex objects incorporating several blob-like features
Remarks	*chu* funds the convex hull of a single blob, whereas *chu_all* finds the smallest convex polygon that encloses a set of disconnected blobs

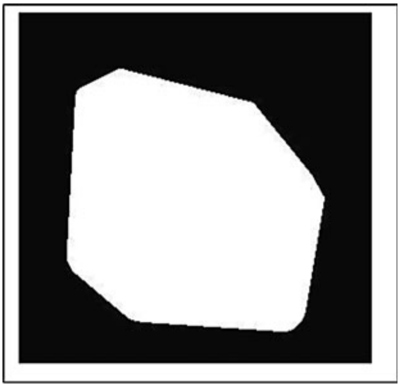
Illustration

Input image	Metal parts
File number	46
Command	*chu_all*

Left: original



Right: after processing



41.48 cig

Format	$b = cig(a)$	
Arguments	*a*	Intensity level
Defaults	*a*	128
Output values	*b*	Number of pixels
Image type	Grey	
Type	Image measurement	
Description	Count points with intensity greater than or equal to a given value	
Switch images	No	
Typical uses	Image analysis	
Remarks	Also see *cgp*	

41.49 cim

Format	$[b,c] = cim(a)$	
Arguments	*a*	Control (switch drawing on/off)
Defaults	*a*	0 (drawing off)
Output values	*b*	X-coordinate of the centre of the image
	c	Y-coordinate of the centre of the image
Image type	Grey	
Type	Image measurement	
Function	Find the coordinates of the centre of the image (if $a = 0$); draw the centre of the image ($a > 0$)	
Switch images	Yes	
Typical uses	Image analysis	
Remarks	Also see *dgw*	

41.50 cir

Format *cir*(*a*, *b*, *c*, *d*)

Arguments *a* X-coordinate of circle centre

b X-coordinate of circle centre

c Radius of circle

d Intensity of circle

Defaults *d* 255

Output values None

Image type Grey or binary

Type Drawing

Function Draw an 'open' circle

Definition A circle centred on [*a*, *b*] and of radius *c* and intensity *d* is superimposed on the current image

Switch images Yes

Typical uses Annotating images for publication

Remarks The program listed below generates a 'circle' consisting of 20000 points. This number can be increased (line 15) to make sure that any gaps are filled, but execution will take longer

41.51 civ

Format *a* = *civ*(*b*, *c*)

Arguments *b* X-address of the pixel to be accessed

c Y-address of the pixel to be accessed

Defaults None

Output values *a* Intensity value at *[b, c]*

Image type Any

Type Accessing individual pixels

Function Find the intensity (*a*) at point [*b*, *c*] in the current image

Switch images No

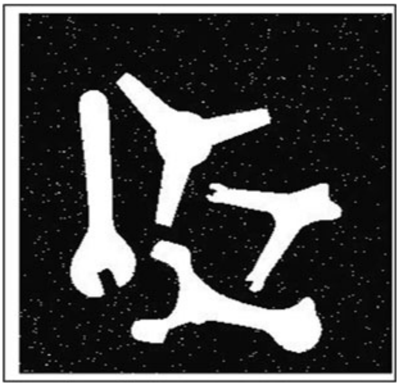
Typical uses Analysing the output of *ftm*, *rdn*, *npo* and other operators that produce an image that is normalised spatially

Remarks Use *civ*, rather than accessing individual pixels of the current image directly, since it follows the QT convention that the x-address precedes the y-address. It also avoids the need to define the current image as *global* via the Command Window

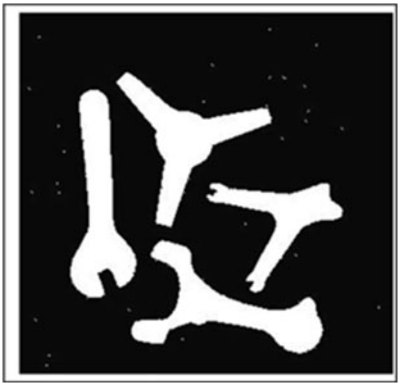
41.52 cln

Format	*cln*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob transform
Function	Remove isolated white pixels
Definition	White pixels that have no white 8-neighbours are set to white. All other pixels remain unchanged
Switch images	Yes
Typical uses	Noise reduction
Remarks	Also see *kgr*
Illustration	
Input image	Metal components with superimposed salt-and-pepper noise
Image generation	rni(46), thr, spn, blf
Command	*cln*

Left: original image



Right: *cln*



41.53 cls

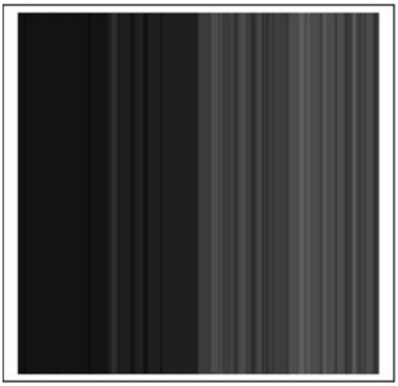
See *dil.*

41.54 cmd

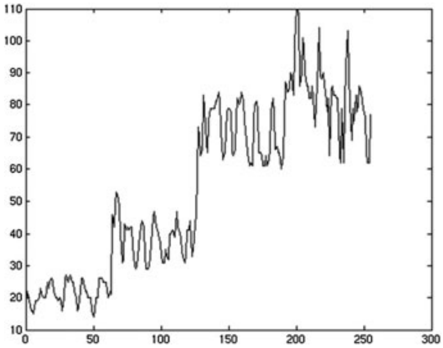
Format	*cmd*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Image measurement
Function	Median intensity for each column in the image
Definition	The *i*th pixel in each row is the median of the intensities in column *i*
Switch images	Yes
Typical uses	Texture analysis
Remarks	Use *yxt*, *cmd*, *yxt* to calculate the median for each row
Illustration	
Input image	Coffee beans (image 4) modified by multiplying by an intensity 'stair-case'.
Image generation	*wgx*, *sca*(*2*), *acn*(*64*), *rni*(*4*), *mul*, *enc*
Command	*cmd*

Top-left: original

Top-right: *cmd*



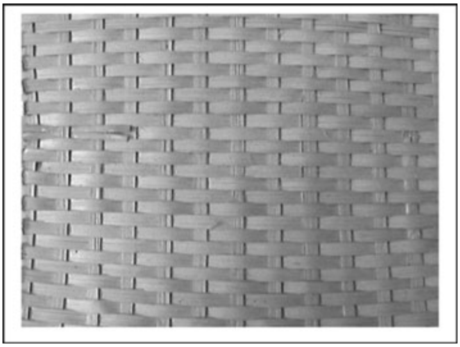
Bottom: *cmd*, *plt*



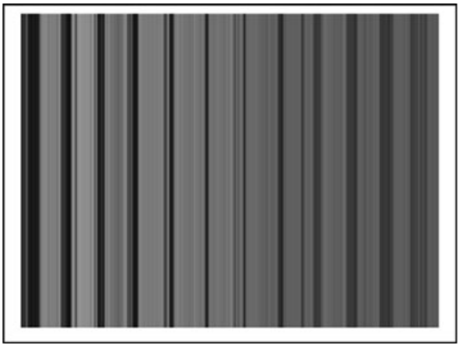
41.55 cmn

Format	*cmn*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Image measurement
Function	Minimum intensity for each column in the image
Definition	The *i*th pixel in each row is the minimum of the intensities in column *i*
Switch images	Yes
Typical uses	Texture analysis
Remarks	Use *yxt*, *cmn*, *yxt* to calculate the minimum intensity for each row
Illustration	
Input image	Wicker-work web
File number	*163*
Command	*cmn*

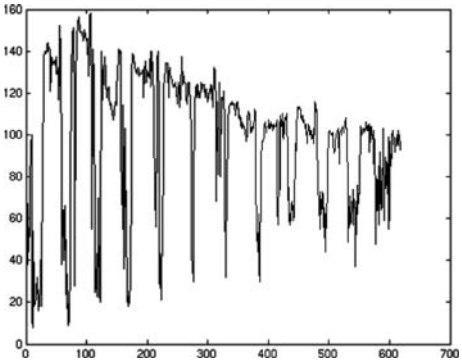
Top-left: original



Top-right: *cmn*



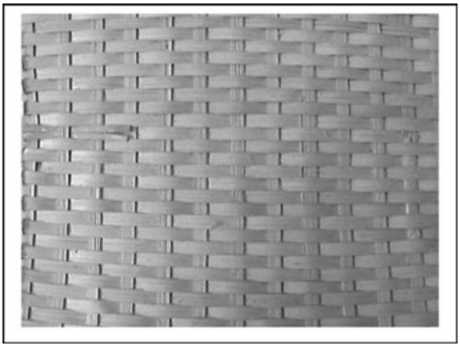
Bottom: *cmn*, *plt*



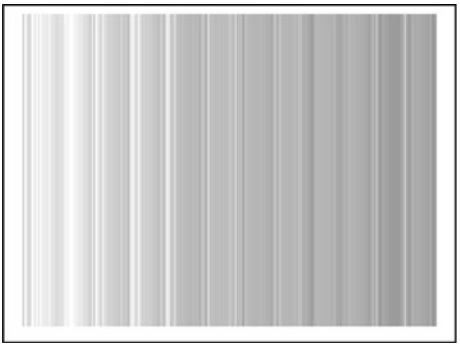
41.56 cmx

Format	*cmx*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Image measurement
Function	Maximum intensity for each column in the image
Definition	The *i*th pixel in each row is the maximum of the intensities in column *i*
Switch images	Yes
Typical uses	Texture analysis
Remarks	Use *yxt*, *cmn*, *yxt* to calculate the maximum intensity for each row
Illustration	
Input image	Wicker-work web
File number	*163*
Command	*cmx*

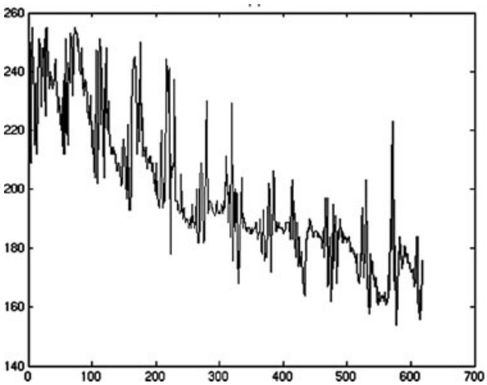
Top-left: original



Top-right: *cmx*



Bottom-left: *cmx*, *plt*



41.57 cnr

Format	*cnr*(*u*, *v*)	
Arguments	*u*	Separation parameter for edge-follower (bugs)
	v	Rescaling parameter, for improving clarity of resulting image
Defaults	*u*	10
	v	20
Output values	None	
Image type	Binary image, single blob, no lakes	
Type	Binary image analysis	
Switch	Yes	
Function	Shade the edge of the blob to highlight corners	
Switch images	Yes	
Typical uses	Feature detection, prior to analysing blob geometry	
Remarks	This operator uses an edge-follower incorporating three “bugs”. The separation of these bugs is controlled by the input parameter *a*. If *a* is large, only major changes in edge direction are detected. Also see *hcd*	

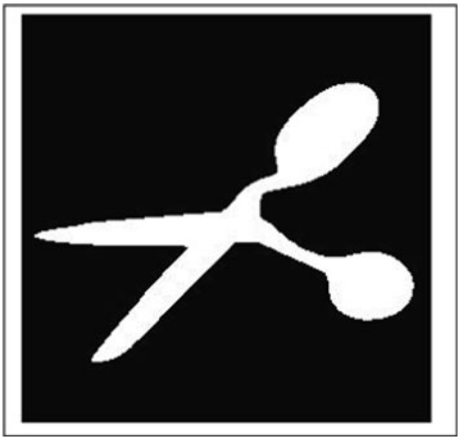
Illustration

Input image *rni*(*49*), *thr*, *blf*

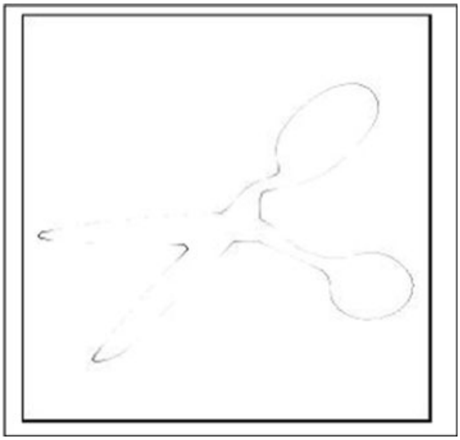
Commands (i) *cnr*, *neg* (*neg* included for sake of illustration)

(ii) *cnr*, *thr*(*100*), *dil*, *rei*('*original*'), *swi*, *adi*, *mxi*

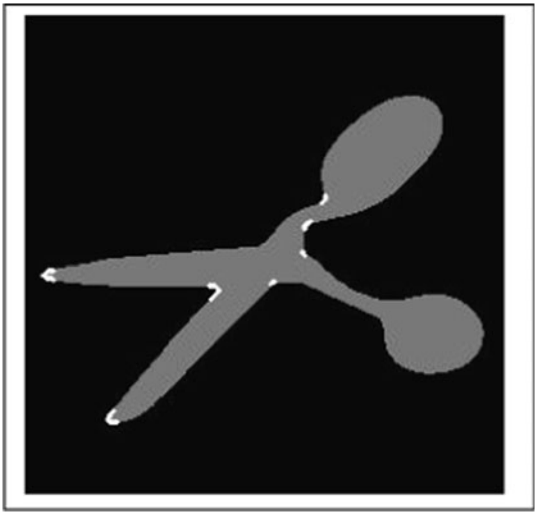
Left: original image



Top-right: *cnr*, *neg*



Bottom: *cnr*, *thr*(*100*), *dil*, *rei*('*original*'), *swi*, *adi*, *mxi*



41.58 cob

Format	*cob*(*a*)
Arguments	*a* Connectivity (4 or 8)
Defaults	*a* 8
Output values	None
Image type	Binary
Type	Miscellaneous
Function	Choose one or more *a*-connected binary objects ($a = 4$ or 8) with the cursor
Switch images	Yes
Typical uses	Interactive selection of blobs in a complex image, prior to more detailed analysis
Remarks	Click on the blob that is to be isolated. Then, press *RETURN*. If more than one blob is to be retained, click on each one in turn, then press *RETURN*. Since it is interactive, the function *cob* is provides a convenient way to avoid a lot of processing, when concentrating on other parts of a complex blob-analysis algorithm

41.59 con/dgr/glc/hgr/lpc/shp/vgr

Format	$[b, c] = con(a)$	
Arguments	*a*	Filter kernel. 1-d or 2-d array
Defaults	None	
Output values	*b*	Multiplicative normalising constant (*k1* in MATLAB listing)
	c	Additive normalising constant (*k2* in MATLAB listing)
Image type	Grey	
Type	Linear filter	

Function: Normalised convolution of a given weight matrix (input array *a*) with the current image. The periphery of the output image is masked to eliminate edge effects

Description: The input array, *a*, defines the weight matrix; the weights can have any real values (positive, zero or negative). First, the normalising coefficients (*k1* and *k2* in the MATLAB function), are computed from the weight matrix. The rescaled weight matrix (*a.k1*) is then convolved with the input image and the constant *k2* is added to the result. Finally, the periphery of the edge is masked (using *edg*), to eliminate edge effects

Switch images: Yes

Typical uses: Low-pass filtering, reducing the effects of noise. High-pass filtering, eliminating slow variations in back-ground intensity

Remarks: Many useful functions can be expressed in terms of *con*:

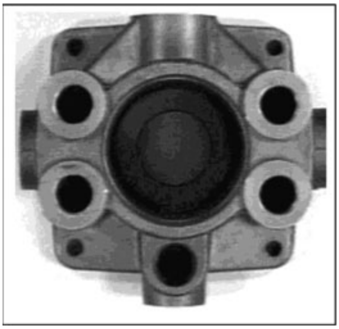
Weight matrix	Operation
[*0, 0, 0; 0, −1, 0; 0, 0, 0*]	Same result as negate (*neg*)
[*0, 0, 0; 1, 0, −1; 0, 0, 0*]	Horizontal gradient (*hgr*)
[*0, 1, 0; 0, 0, 0;0,−l, 0*]	Vertical gradient (*vgr*)
[*−1, −, 1, −1; −1, 8,−1; −1,−1, −1*]	Highlight 1-pixel spots (*lpc*)
[*− l, −, l, −l; −1, 9, −1; −1, −1, −1*]	Sharpen (*shp*)
[*1, 2, 1; 0, 0, 0; −1, −2,−1*]	Horizontal smoothing, vertical difference. (*sed* uses two operators like this)
[*0, 0, 1; 0, 0, 0; 0, 0, 0*]	Shift up one row and right-wards one column
[*1, 0, 0; 0, 0, 0; 0, 0, −1*]	Diagonal gradient (*dgr*)
[*1, 1, 1; 1, 1, 1; 1, 1, 1*]	Count white neighbours. (Binary image operator) Functionally equivalent to *cnw*, *enc*
[*1, 1, 1; 1, −1, −1; 1, −1, 1*]	Detects *top-left* corners. Use *thr*(*255*) after *con*

Note: In some versions of QT for MS-WINDOWS, the mnemonic *con* causes problems and is renamed *glc*.

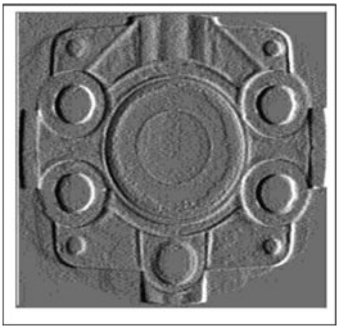
Illustration

Input image	Hydraulics component
File number	37

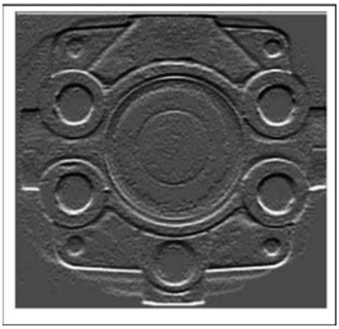
Top-left: original image



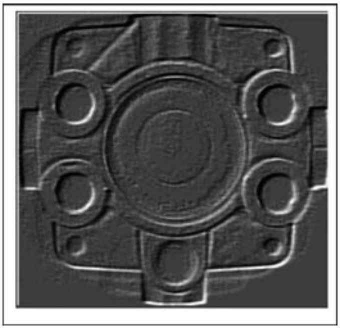
Top-right: *hgr*



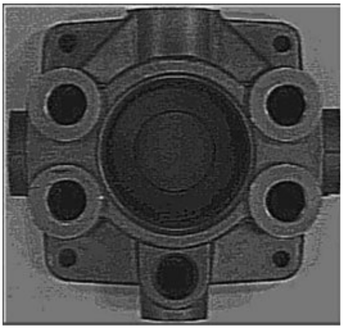
Centre-left: *vgr*



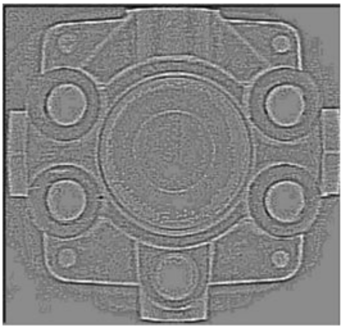
Centre-right: *dgr*



Bottom-left: *con*([−1,−1,−1; −1, 9,−1; −1,−1,−1])



Bottom-right: *con*([−1,−1,−1; −1, 8,−1; −1,−1,−1])



41.60 cpy

Format	*cpy*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Any
Type	Miscellaneous
Function	Copy the current image to the alternate image

41.61 crk

See *dil.*

41.62 crp

Format [*a*, *b*, *c*, *d*] = *crp*(*a*, *b*, *c*, *d*)

Arguments *a* X-coordinate of the left-hand side of the cropping rectangle

b Y-coordinate of the top of the cropping rectangle

c Width of the cropping rectangle

d Height of the cropping rectangle

Defaults *No arguments supplied*: Use the cursor to define the cropping rectangle

Output values *a* X-coordinate of the left-hand side of the cropping rectangle

b Y-coordinate of the top of the cropping rectangle

c Width of the cropping rectangle

d Height of the cropping rectangle

Image type Grey-scale or binary

Type Miscellaneous

Function Crop the image; keep only those pixels that lie within the cropping rectangle

Definition *No arguments specified*: Cursor defines diagonal corners of the cropping rectangle

Four arguments specified: Crop the image to the specified rectangle

Switch images Yes

Typical uses Defining a region of interest (ROI), so that subsequent processing is faster

Remarks The cropping rectangle can be defined interactively by the user, static (fixed by specifying constant values within the program), or dynamic (the input parameters [*a*, *b*, *c*, *d*] are calculated within the program)

Illustration

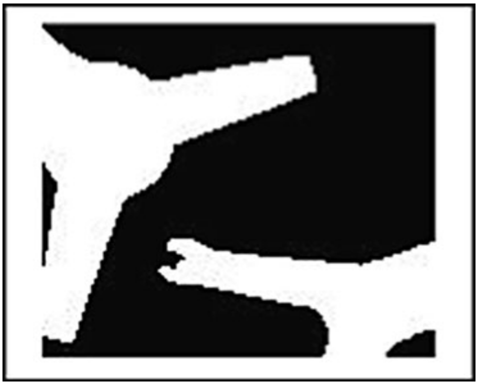
Input image Metal components

File number 46

Command *crp* (User selects the region of interest with the cursor)

Left: original image

Right: *crp*



41.63 crs/dcr/dsq

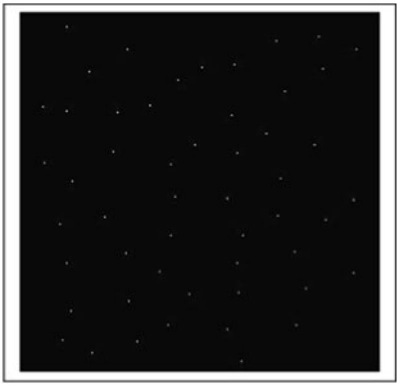
Format	*crs*(*a*) *dsq*(*a*) *dcr*(*a*)
Arguments	*a* Size of symbol to be drawn
Defaults	*a* 3
Output values	None
Image type	Binary image containing one or more isolated white pixels
Type	Drawing
Function	Replace isolated white pixels by symbols that are easier to see: up-right cross (*crs*), diagonal cross (*dcr*), or square (*dsq*)
Switch images	Yes
Typical uses	Displaying results, preparing documents
Remarks	Some operations (e.g., *shk*) generate single, isolated pixels that are difficult to see, particularly. These three functions are used to display them as small symbols that are easier to see

Illustration

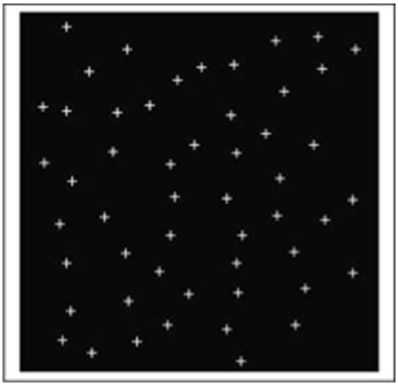
Image generation *rni*(*3*), *enc*, *thr*(*97*), *kgr*(*50*), *rbe*, *shk*

Commands (*i*) *crs*(*3*)
(*ii*) *dcr*(*3*)
(*iii*) *dsq*(*3*)

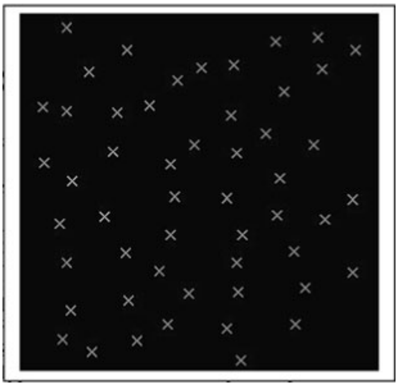
Top-left: original image



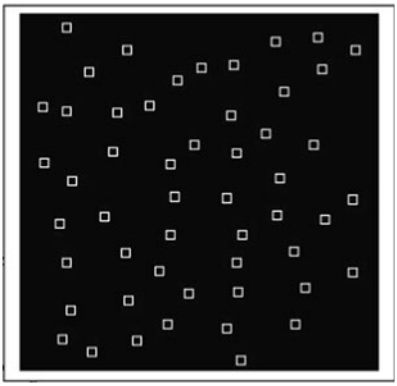
Top-right: up-right crosses, *crs*



Bottom-left: diagonal crosses, *dsq*



Bottom-right: squares, *dsq*



41.64 cse

Format	*b* = *cse*(*a*, *b*)
Arguments	*a* Structuring element type *b* Structuring element size
Defaults	None
Output values	Structuring element: a square logical array, consisting of 0s and 1s
Image type	None
Type	Morphology
Function	Generate a structuring element for either dilation (*gbd*) or erosion (*gbe*)
Switch images	No
Typical uses	Used only in conjunction with *gbd* or *gbe*
Remarks	The function *cse* can be used to generate several frequently used structuring elements:

Input, a	*Structuring element*
1	Horizontal line
2	Vertical line
3	Diagonal line (top-left to bottom-right)
4	Diagonal line (top-right to bottom-left)
5	Up-right cross
6	Diagonal cross
7	Open square
8	L-shape (L is upside-down):
9	V-shape (angle 45°)
10	V-shape (angle 90°)
11	V-shape (angle 135°)
12	Open diamond
13	Open circle

cse has little use apart from generating structuring elements for *gbd*/*gbe*. It is possible to generate structuring elements simply by entering the array via the keyboard, or combine two templates of the same size using a script:

```
a = cse(6, 15)      % Diagonal cross SE, 15*15 pixels
b = cse(7, 15);     % Open squares SE, 15*15 pixels
c = a | b;          % Logical OR of SEs a and b
gbd(c)              % Dilation. Alternatively, use gbe (erosion) here
```

The MATLAB function *strel* is also able to creating structuring elements, including multiple lines, diamond, rectangle, disc and octagon

Also see *gbd* and *gbe*

41.65 csg

Format *csg*(*a*)

Arguments *a* Enable/disable superimposition of plots and control colour of plots

Defaults *a* 0 (plot in red and clear previous plots)

Image type Colour

Type Colour image analysis

Function The user first draws a closed polygonal mask over the image. The colours of pixels within this region are represented by a cloud of points in RGB space. The user is then able to explore the 3D scattergram interactively, using the mouse to choose the viewing point (This is a standard MATLAB feature)

Switch images Yes

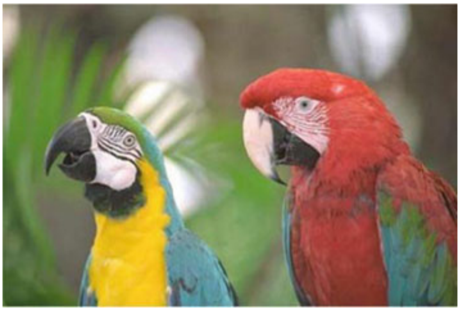
Remarks This function is useful for investigating the clustering of points in RGB space. *csg* is an aid to understanding, particularly when designing colour recognition systems (Also see *pgn* and *mci*)

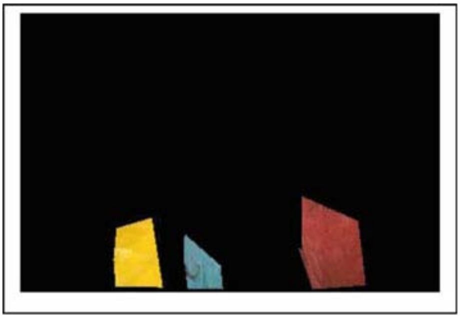
Illustration

Input image	Birds
File number	74
Commands	*rni*(*74*), *csg*(*0*), *wri*(*1*), *rni*(*74*), *csg*(*2*), *rei*(*1*), *mxi*, *wri*(*1*), *rni*(*74*), *csg*(*4*), *rei*(*1*), *mxi*

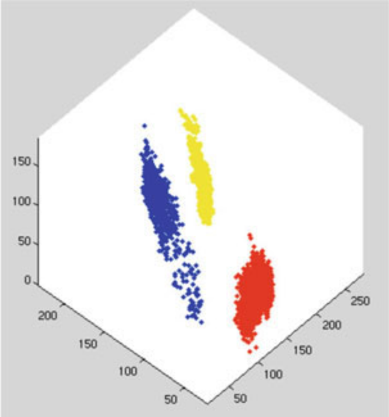
Top-left: original colour image

Top-right: mask regions selected with the mouse





Bottom: RGB scattergram. Notice that there are three clusters; one cluster is generated by each of the three masked regions. Since these clusters are well separated, it is a trivial exercise to design a system/program to recognize these particular samples of red, yellow and blue



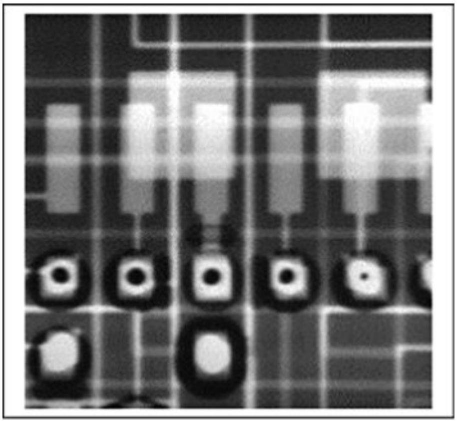
41.66 csh/rsb

Format	*csh*(*a*) *rsb*	
Arguments	*a*	Index number of column to be copied (*csh* only)
Defaults	*a*	Index of right-most column (*csh* only)
Output values	None	
Image type	Grey-scale or binary	
Type	Global transform	
Function	*csh*:	Copy given column to all other columns; spread this column horizontally
	rsb:	Copy the bottom row into every other row
Switch images	Yes	
Typical use	Drawing an intensity profile as an image	
Remarks	Also see *kgr*	

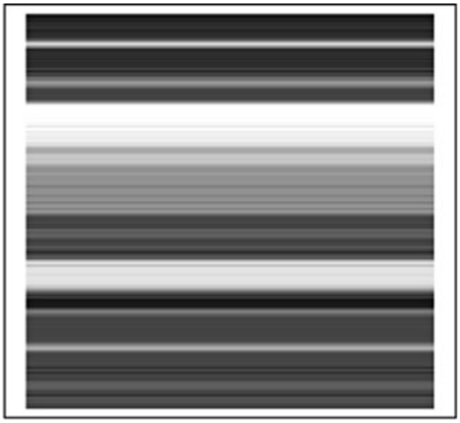
Illustration

Input image	Printed circuit board, x-ray image
File number	135
Commands	*enc, csh*

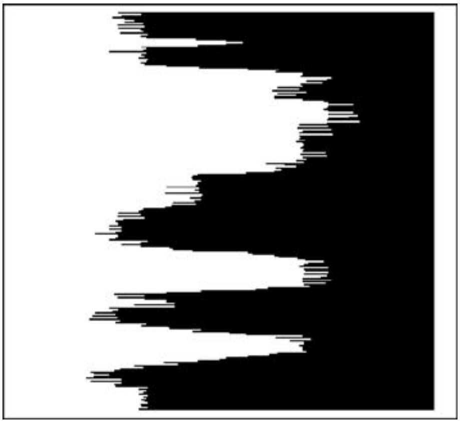
Top-left: original image



Top-right: *enc, csh*



Bottom-left: *[a, b] = cim, csh(a), wgx, sub, thr*



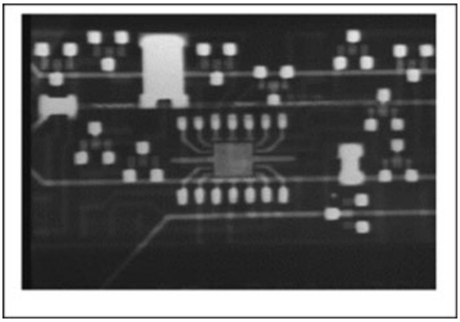
41.67 csi

Format	*[w, k1, k2] = csi(a, b, c, d)*	
Arguments	*a*	Processing window, X-coordinate of top-left-most point
	b	Processing window, Y-coordinate of top-left-most point
	c	Processing window, width
	d	Processing window, height
Defaults	None	
Output values	*w*	Convolution weight matrix
	k1	Normalisation coefficient (See *con*)
	k2	Normalisation coefficient (See *con*)
Image type	Grey scale or binary	
Type	Linear filter	
Switch	Yes	
Function	Convolve a sub-image with the whole image	
Definition	Crop the input image by selecting the rectangle defined by *[a, b, c, d]*. Use this sub-image to generate a weight matrix (*w* in the MATLAB listing). Convolve this with the (whole) input image	
Switch images	Yes	
Typical uses	Identifying repeated small features	
Remarks	Use *crp* to select the cropping rectangle	

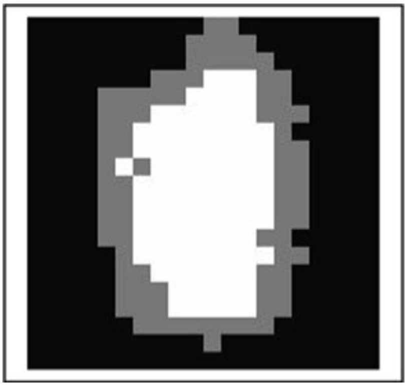
Illustration

Input images PCB x-ray
File numbers 96
Command: *csi*(*140*, *160*, *20*, *20*)

Top-left: original

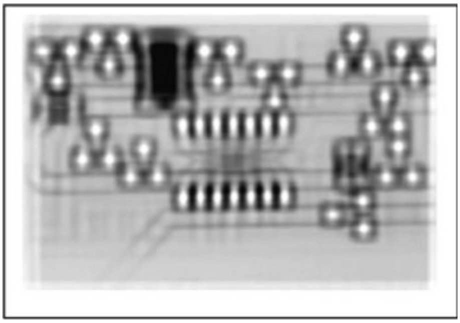


Top-right: weight matrix generated by *csi*(*140*, *160*, *20*, *20*)

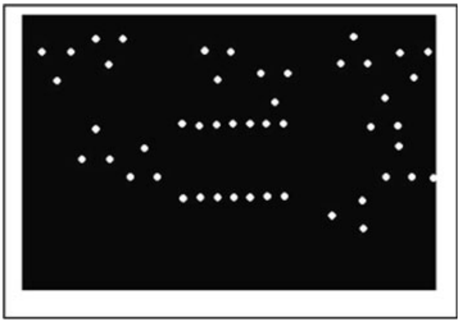


Key	
Intensity	**Weight**
Black	−1
Grey	0
White	+1

Bottom-left: after *csi*



Bottom-right: peak intensities, found by thresholding



41.68 csk

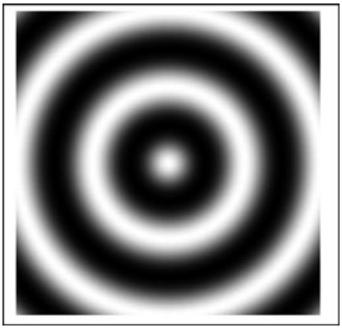
Format	*csk*
Arguments	None
Output values	None
Image type	Any
Type	Stack operator
Function	Clear the image stack. The image files are not destroyed; each image in the stack (i.e., *stack_image*(*i*), where *i* = *0*, *2*, .., (*stack_size-1*)) is filled with Gaussian noise. These images are all different
Switch images	No
Remarks	Also see *rsk*

41.69 csp

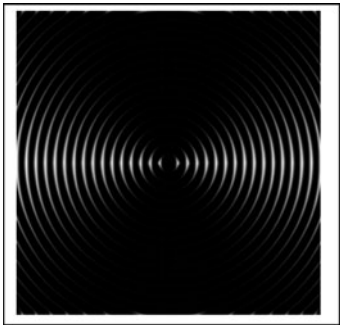
Format	*csp*(*a*, *b*)	
Arguments	*a*	Wavelength
	b	Image size
Default	*a*	Image size/2
	b	Size of current image (larger of height and width)
Image type	Grey scale	
Type	Image generation	
Switch	Yes	
Function	Generate a radially symmetric pattern in which the intensity varies as a cosine function of the distance from the centre of the image	
Switch images	Yes	
Typical uses	Testing/demonstrating image convolution filters and morphology operators	

Illustration

Left: *csp*



Right: *csp*(*16*, *512*), *neg*, *dil_r*(*15*, *3*), *neg*



41.70 ctp

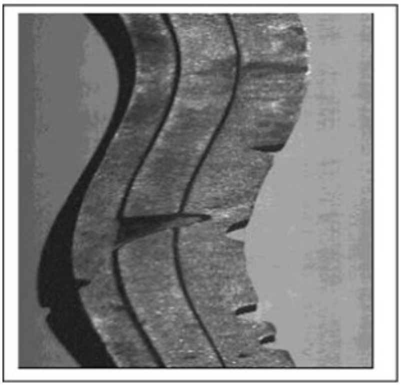
Format	*ctp*(*a*, *b*)	
Arguments	*a*	X-coordinate for the pivotal point for the axis transformation
	b	Y-coordinate for the pivotal point for the axis transformation
Defaults	*a*	X-coordinate of the centre of the image
	b	Y-coordinate of the centre of the image
Output values	None	
Type	Image warping	
Switch	Yes	
Function	Cartesian-to-polar coordinate axis transformation	
Switch images	Yes	
Typical uses	Transforming an image created using a line-scan camera, with the widget mounted on a turntable, so that it appears more natural. For example, a circular object looks circular.	
Remarks	In the illustration below, the black triangular regions at the corners are artifacts of the mapping process and should be ignored. (No information is available for these areas, so they are arbitrarily assigned the value zero.) The centre of the turntable is at the centre of this image, Hence, the displacement of the widget centre relative to that of the turntable can be estimated. *ctp* and *ptc* implement inverse functions	

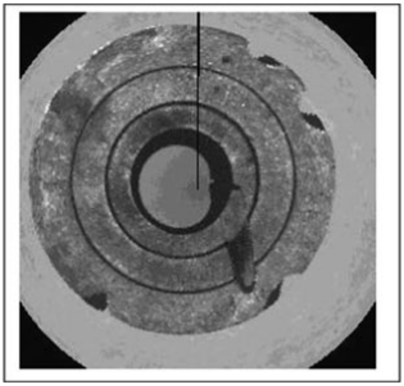
Illustration

Input image	Line-scan camera viewing a circular component mounted on a turntable
File number	99
Commands	*yxt*, *ctp*

Below: original image

Right: processed image

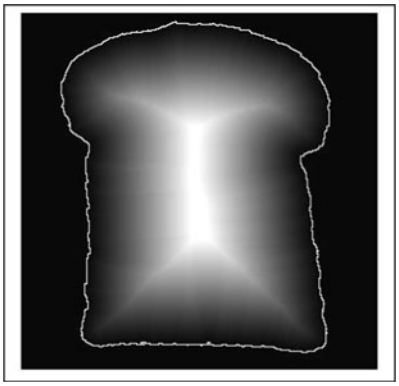




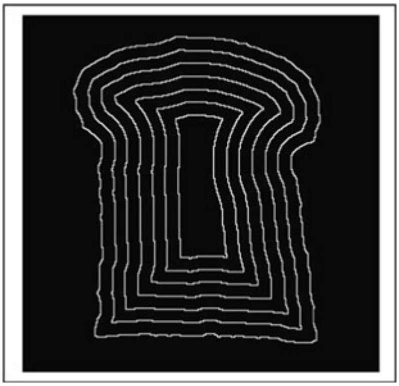
41.71 ctr

Format	*ctr*
Arguments	*None*
Defaults	Not applicable
Output values	None
Image type	Grey scale
Type	Image analysis
Switch	Yes
Function	Draw intensity contours (isophotes) in the current image
Typical uses	Visualising slow intensity variations that are difficult to see otherwise
Remarks	Also see *iso*
Illustration	
Input image	Slice of bread (file number 45), after grass-fire transform (*gft*)
Image generation	*rni*(*45*), *gft*, *enc*
Commands	*ctr*

Left: original (slice outline added for clarity)



Right: after processing



41.72 cur

Format	*[a, b, c] = cur*
Arguments	None
Defaults	Not applicable
Output values	*a* X-coordinate
	b Y-coordinate
	c Intensity value
Image type	Grey or binary
Type	Measurement and analysis
Function	Find the position coordinates and intensity at a point selected interactively, with the cursor
Switch images	No
Typical uses	*cur* is used principally as an aid to understanding the structure of an image and in guiding algorithm design
Remark	The values of *a*, *b* and *c* are defined by the latest clicked position of the cursor before the user presses RETURN

41.73 cwp

Format	*a = cwp*
Arguments	None
Defaults	Not applicable
Output values	*a* Number of white pixels
Image type	Binary
Type	Binary image measurement
Function	Count the number of white pixels in a binary image
Switch images	No
Typical uses	Measuring the area of a single blob, or the average area of a number of blobs (Use *cbl* to count them)
Remarks	Also see *cgp*

41.74 cxa

See *bpa.*

41.75 dab

Format *dab*(*a*)

Arguments *a* *QT command that is to be performed on all blobs in the current image*

Defaults None

Output values None

Image type Binary

Type Controls image processing

Function Perform the image processing operation defined by input parameter *a*, on all blobs

Switch images Yes

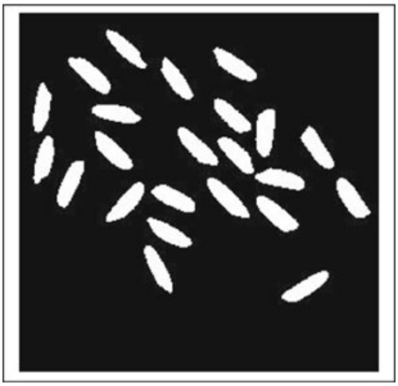
Typical uses Analysing images containing multiple blobs

Remarks Notice that *dab*('*chu*') draws the convex hull of each blob separately; it does not form the convex hull of the complete collection. On the other hand, *chu* draws the convex hull of just one blob. Also see *cgr_all*, *nxy_all*, *npo_all*,

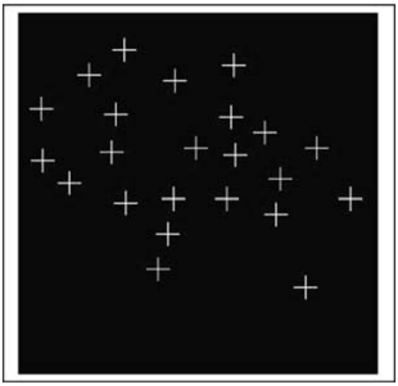
Illustration

Input images	(i) *rni*(*3*), *enc*, *thr*, *rbe*, *kgr*(*300*)
	(ii) *rni*(*46*), *thr*
Commands	(i) *[a, b]* = *cim*; *dab*('*dcg*')
	(*ii*) *dab*('*chu*')

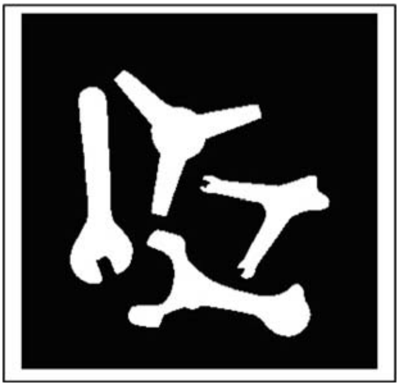
Top-left: rice grains after preprocessing



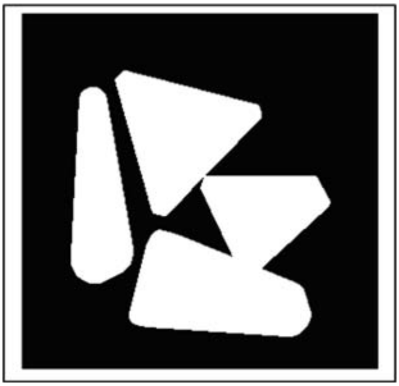
Right: after *dab*('*dcg*')



Bottom-left: Four objects (image 46)



Right: *dab*('*chu*')



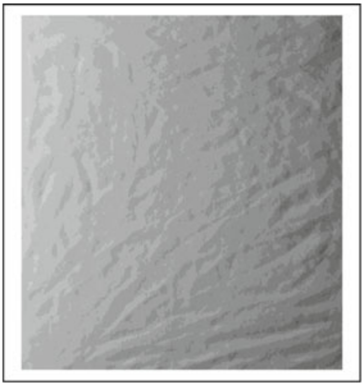
41.76 dbn

Format	*dbn*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey scale
Type	Image filter
Switch	Yes
Function	Direction of brightest neighbour
Definition	Each pixel is assigned a value indicating the direction of the maximum intensity among its 8-neighbours
Typical uses	Enhancing low-contrast intensity variations that are difficult to see otherwise
Remarks	It may be beneficial to apply a low-pass filter (e.g., *raf*, *caf*), or median filter (*mdf*) to attenuate camera noise before using *dbn*

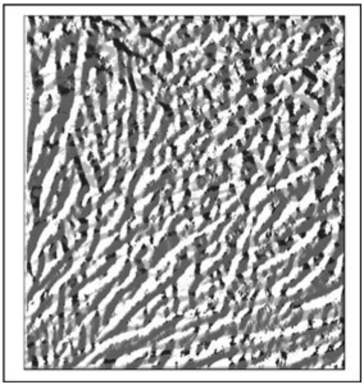
Illustration

Input image	Creased fabric (file number 28), followed by *enc*, *caf*
Image generation	*rni*(*28*), *enc*, *caf*
Commands	*dbn*, *enc*

Left: original



Right: after processing



41.77 dcg

Format	*dcg*
Arguments	None
Defaults	None
Output values	None
Image type	Binary
Type	Drawing
Function	Draw a small cross at the centroid of a blob. If there is more than one blob, the first one found during a raster scan is analysed
Switch images	Yes
Typical uses	Displaying processing results to the user
Remarks	Use *dab*('*dcg*') to show the centroids of all blobs in the input image. (See *dab*)

41.78 dci

Format	*dci*(*a*)	
Arguments	*a*	Control cross size
Defaults	*a*	0 (Draw a large cross)
Output values	None	
Image type	Grey scale or binary	
Type	Image annotation	
Function	Draw a cross (small if $a = 1$; large if $a = 0$) at the centre of the image	
Switch images	Yes	
Typical uses	Emphasising results for the user	
Remarks	Also see *dcl*	

41.79 dcl

Format	*dcl*(*x*, *y*, *a*)	
Arguments	*x*	X-coordinate of the centre of the cross
	y	Y-coordinate of the centre of the cross
	a	Size of the cross
Defaults	*a*	8
Output values	None	
Type	Drawing and image generation	
Switch	Yes	

Function: Draw an 'up-right' cross at a given point, *[a, b]*. The size of the cross is determined by the control parameter *a*

Switch images: Yes

Typical uses: Indicating the position of features detected during processing, e.g., centroid

Remarks: The length of the arms of the cross is given by *a* and may exceed the width and/or height of the image. The arms are one pixel wide. The command sequence *[p, q]* = *cgr*, *hin*, *dcl*(*p*, *q*) plots the position of the centroid of a blob. To draw a diagonal cross, lines 8 and 9 in the MATLAB program should be changed to:

$$vpl((x - a), (y - a), (x + a), (y + a), 255);$$
$$vpl((x + a), (y - a), (x - a), (y + a), 255);$$

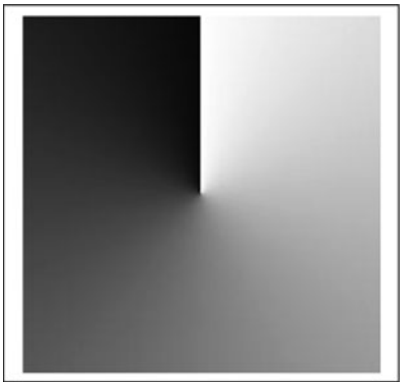
41.80 dcw

Format	*dcw*(*u*, *v*)	
Arguments	*u*	X-coordinate of the centre or 'pivotal point' of the circular wedge
	v	Y-coordinate of the centre or 'pivotal point' of the circular wedge
Defaults	*u*	Centre of the image
	v	Centre of the image
Output values	None	
Type	Drawing and image generation	
Switch	Yes	
Function	Draw a circular wedge centred at *[u, v]*.	
Switch images	Yes	
Typical uses	Measuring angles of point features in an image, relative to some reference point	
Remarks	A circular wedge is an image in which the intensity of a pixel *[p, q]* is proportional to the angle of that line through *[p, q]* and the given point *[u, v]* (the centre or 'pivotal point'), relative to the vertical axis. A set of binary 'wheel spokes' can be drawn using *dcw*, *sca*(4), *sub*, *enc*, *thr*	

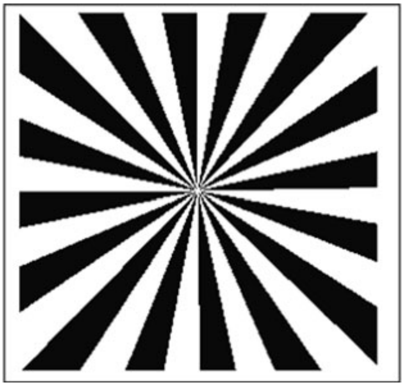
Illustration

Commands Left: *dcw*. *Right*: *dcw*, *sca*(4), *sub*, *enc*, *thr*

Left: circular intensity wedge



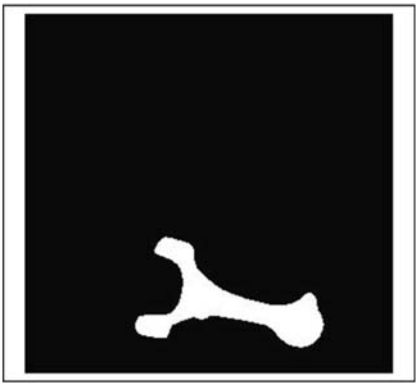
Right: 'Wheel spokes'



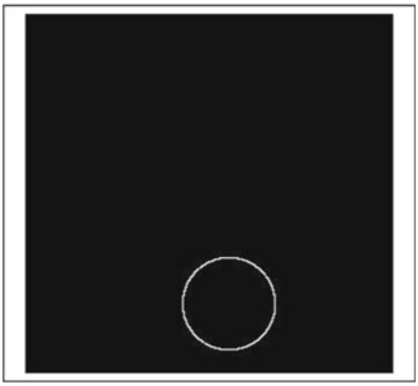
41.81 dec

Format	*dec*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Drawing
Function	Draw an ‘open’ circle of the same area as the blob in a binary image
Definition	Locate the centroid, [*a*, *b*], and calculate the radius (*c*) of the circle that has the same area as the given blob. Draw the ‘open’ circle centred on [*a*, *b*] and of radius *c*, against a white background
Switch images	Yes
Typical use	Simplifying images, when size (area), rather than shape is important. This may be useful for display purposes
Remarks	The circle can be filled using *blf*. Also see *eqd*
Illustration	
Input image	Con-rod: image with four metal components (file number 46) with *big* applied
Image generation	*rni*(*46*), *thr*, *big*
Commands	*dec*

Left: original image



Right: *dec*



41.82 delay

Format	*delay*(*a*)
Arguments	*a* Delay duration, in seconds
Defaults	None
Output values	None
Image type	Not relevant
Type	Program control
Function	Delay the program execution by a defined period (*a* seconds)
Switch images	No
Typical uses	Controlling external devices
Remarks	At any moment of his/her choosing, the user can terminate the timer and recover control, by pressing CTRL/C

41.83 dev

Format	$a = dev$
Arguments	None
Defaults	Not applicable
Output values	*a* Standard deviation of the intensity
Image type	Grey-scale
Type	Grey-scale image measurement
Function	Find the standard deviation of the intensity over the whole image
Switch images	No
Typical uses	Image enhancement. Choosing a threshold parameter
Remarks	Also see *avg* and *gli*

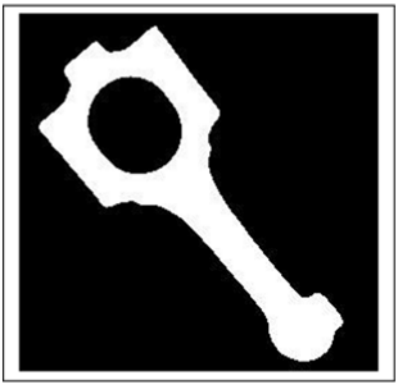
41.84 dfc

Format	*dfc*
Arguments	None
Output values	None
Image type	Binary
Type	Binary image transformation/analysis
Function	Set the intensity each white point to a value equal to the Euclidean distance from the centroid of all white pixels
Switch images	Yes
Typical uses	Object recognition
Remarks	The histogram of grey-level values generated by *dfc* provides a useful way to represent a blob independently of its orientation. Also, consider two sets of spots lying on concentric circles of different radii. Using *dfc*, it is possible to shade the spots, so that one ring can be distinguished from the other, by simple thresholding. Also see *rlc*, *gft*, *dfp* and *pon*

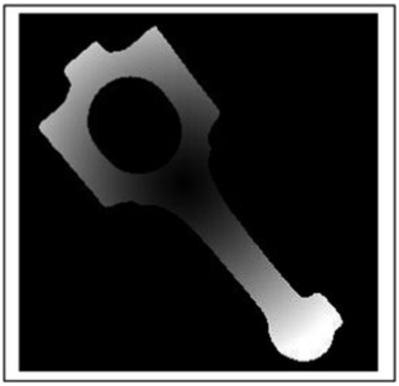
Illustration

Input image	Con-rod, after applying thr to eliminate JPEG artifacts
File number	47
Command	*dfc*, *enc*

Left: original image



Right: *after processing*



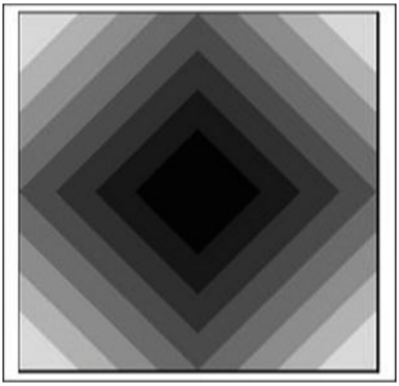
41.85 dfp

Format	*dfp*(*x*, *y*, *r*)	
Arguments	*x*	X-coordinate of the reference point
	y	Y-coordinate of the reference point
	r	Distance metric
Defaults	*r*	2 (Euclidean distance)
Output values	None	
Image type	Any	
Type	Image generation	
Function	Set the intensity each point *[x, y]* to a value proportional to the Minkowski *r*-distance from *[x, y]*. If $r = 1$, the City Block (Manhatten) distance is computed. If $r = 2$, the Euclidean distance is calculated. As *r* tends to infinity (> 20), the Minkowski r-distance approaches the Square distance	
Switch images	Yes	
Typical uses		
Remarks	Contours of constant Euclidean distance form circles. The equivalent figure for $r = 1$, is a 'diamond' (i.e., a square with sides inclined at 45° to the coordinate axes). When *r* is greater than 2, the equivalent figure is a rounded square. When *r* is infinite, the figure becomes a square with sides at 0°/90° to the axes	

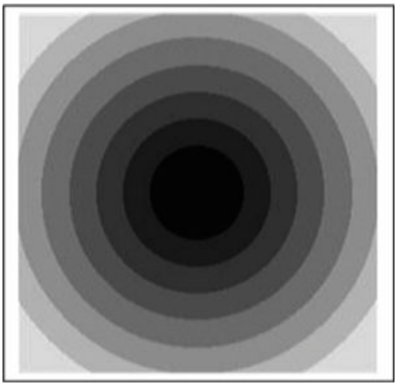
Illustration

Commands See below. *dfp* generates a smooth intensity function. The stepped effect, achieved using *sca*, was introduced here to emphasise the relationship between the metric (set by the value of *r*) and the circle-equivalent figure

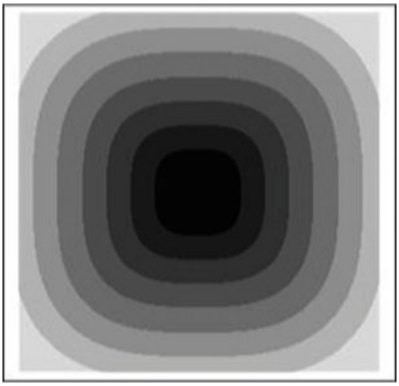
Top-left: *[a, b] = cim*; *dfp*(*a*, *b*, *1*), *enc*, *sca*(*3*)



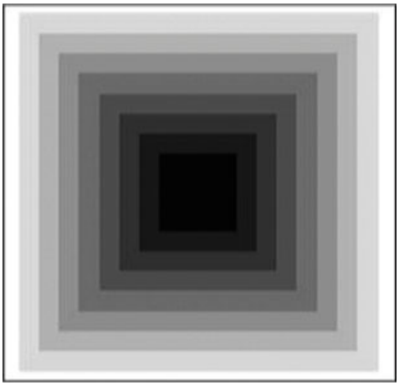
Top-right: *[a, b] = cim*; *dfp*(*a*, *b*, *2*), *enc*, *sca*(*3*)



Bottom-left: *[a, b] = cim*; *dfp*(*a*, *b*, *3*), *enc*, *sca*(*3*)



Bottom-right: *[a, b] = cim*; *dfp*(*a*, *b*, *30*), *enc*, *sca*(*3*)



41.86 dgr

See *con.*

41.87 dgw

Format	*[a, b] = dgw*	
Arguments	None	
Defaults	Not applicable	
Output values	*a*	Image width (number of columns)
	b	Image height (number of rows)
Image type	Any	
Type	Miscellaneous	
Function	Find the size of the image	
Switch images	No	

41.88 dic

Format *dic*(*u*, *v*)

Arguments *u* X-coordinate of the ‘apex’ of the cone

v Y-coordinate of the ‘apex’ of the cone

Defaults *u* Centre of the image

v Centre of the image

Output values None

Type Drawing and image generation

Switch Yes

Function Draw an intensity cone centred at *[u, v]*.

Switch images Yes

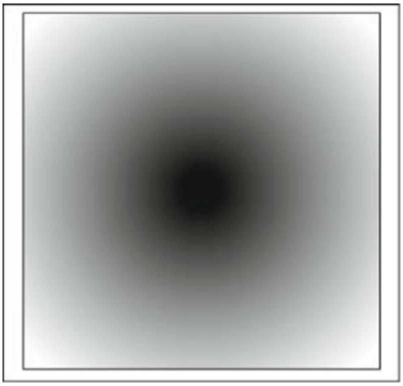
Typical uses Measuring Euclidean distance from a given point in an image

Remarks An intensity cone is a pattern in which the intensity is equal to the distance from the centre point, *[u, v]*. A set of concentric circles can be drawn using *dic*(*u*, *v*), *sca*(*3*), *sed*, *thr*(*8*)

Illustration

Command *dic*, *enc* (Note: *enc* was included to improve the contrast, for display purposes)

Below: intensity cone

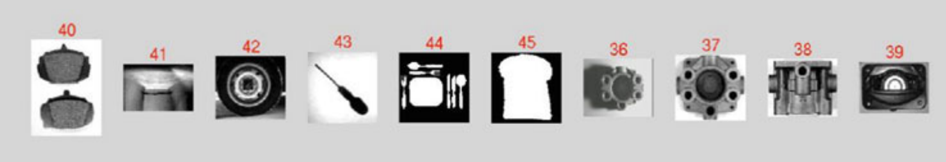


41.89 did

Format	*did*(*a*)
Arguments	*a*
Default	QT_images
Output values	None
Image type	Not applicable
Type	Display of available images
Switch images	No
Function	Dynamic image display. The images held in folder *a* (default *a* = *QT_images*) are read and displayed in ascending numerical order, in Figure 4 (full screen). When all images have been displayed, Figure 4 closes automatically. The user can terminate execution prematurely at any time, by clicking anywhere in Figure 4
Typical uses	Helping the user to identify images

Illustration

Thumbnail images are displayed in numerical sequence as *did* is running. Image 46 will be the next image to be displayed, thereby replacing the thumbnail of Image 36. Following that Image 47 replaces Images 37 and so on.



41.90 dil/cls/crk/ero/nlp/opn

Format *dil*(*a*, *b*)
crk(*a*, *b*)
cls(*a*, *b*)
ero(*a*, *b*)
opn(*a*, *b*)
nlp(*a*, *b*)

Arguments *a* Fixes size of structuring element
b Selector for structuring element

Defaults *a* 1 {See program listing for *dil*}
b 0 (disk structuring element)

Output values None

Image type Grey scale or binary

Type Non-linear filter

Switch Yes

Functions Closing (*cls*),
Crack detection (high pass filtering, *crk*),
Dilation (*dil*),
Erosion (*ero*),
Low pass filtering (*nlp*),
Opening (*opn*)

Definitions Dilation (*dil*)

$$C(i, j) \Leftarrow MAX(A(i + p, j + q) \mid [p, q]\ S),$$

(*S* is the structuring element.) That is, *C*(*i*, *j*) is set to the brightest pixel covered by the structuring element when it is resting on pixel (*i*, *j*). This rule is equivalent to the following:

When the SE rests on pixel (*i*, *j*) *and if any pixel is white within the SE, then C(i, j) is set to white*

Erosion (*ero*)

$$C(i, j) \Leftarrow MIN(A(i + p, j + q) \mid [p, q]\ S),$$

where *S* is the structuring element Closing, opening, crack detection, low-pass filtering, can be expressed as combinations of dilation and erosion, although they are implemented directly in MATLAB to improve speed

Function	Equivalent to
cls(*a*, *b*)	*dil*(*a*, *b*), *ero*(*a*, *b*)
crk(*a*, *b*)	*wri*, *dil*(*a*, *b*), *ero*(*a*, *b*), *ero*(*a*, *b*), *dil*(*a*, *b*), *rei*, *sub*
nlp(*a*, *b*)	*dil*(*a*, *b*), *ero*(*a*, *b*), *ero*(*a*, *b*), *dil*(*a*, *b*)
opn(*a*, *b*)	*ero*(*a*, *b*), *dil*(*a*, *b*)

Typical uses *dil*, *ero*, *cls* and *opn* are useful for identifying blobs/arcs that are above/below a certain size/width. *nlp* are useful for reducing the effects of noise. *crk* is particularly useful for finding crack, splits and thin arc-like defects

Remarks There are several possible shapes of structuring element for each of these functions:

Input parameter b	Structuring element
0	Disk
1	Square
2	Diamond
3	Octagon

All six functions are defined in a similar manner with small changes/ additions at line 33 in the program

Function	*Line 34 in* dil *is replaced by*
ero	`current = imerode(current, c);`

cls	`current = imdilate(current, c);`
	`current = imerode(current, c);`

opn	`current = imerode(current, c);`
	`current = imdilate(current, c);`

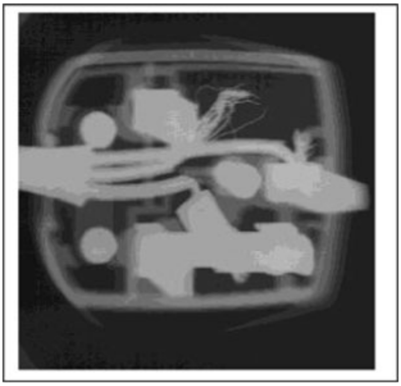
nlp	`current = imdilate(current, c);`
	`current = imerode(current, c);`
	`current = imerode(current, c);`
	`current = imdilate(current, c);`

crk	`wri`
	`current = imdilate(current, c);`
	`current = imerode(current, c);`
	`current = imerode(current, c);`
	`current = imdilate(current, c);`
	`rei`
	`sub`

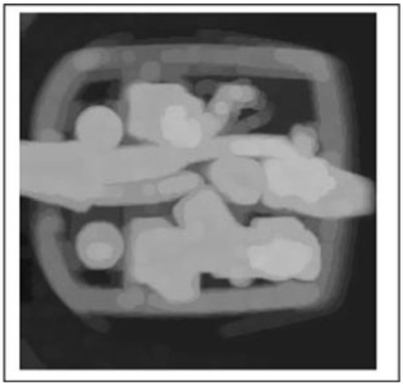
Illustration

Input image	X-ray of standard UK power plug
File number	112
Commands	See below

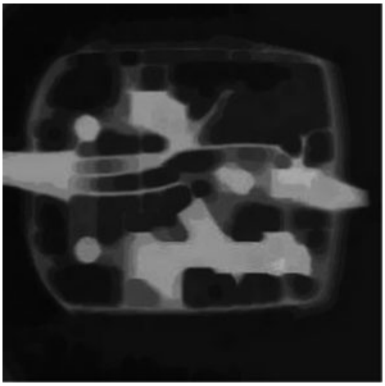
Original image



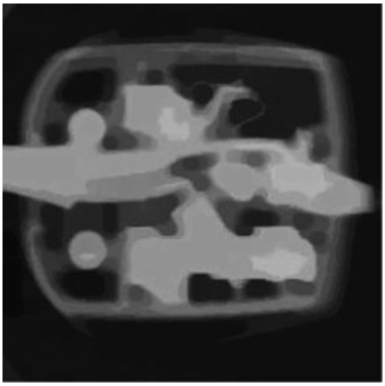
dil(5)



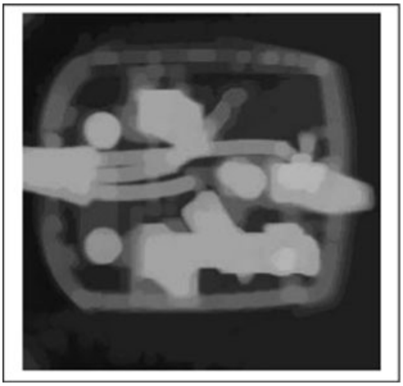
ero(5)



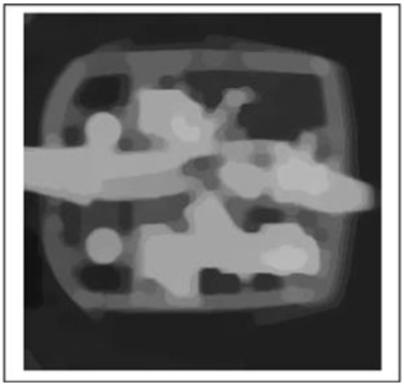
cls(5)



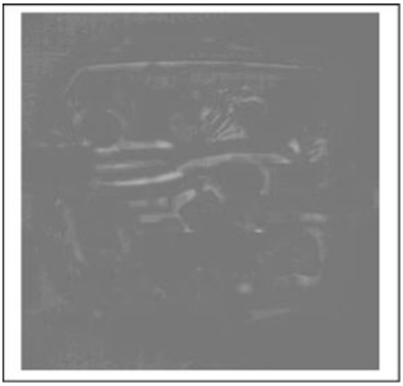
opn(5)



nlp(5)



crk(5)



crk(5), enc

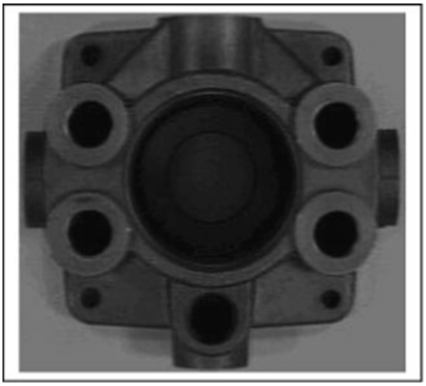
41.91 dim

Format	*[a, b] = dim*
Arguments	None
Defaults	Not applicable
Output values	*a* Image width (number of columns) *b* Image height (number of rows)
Image type	Any
Type	Miscellaneous
Function	Find the size of the image
Switch images	No

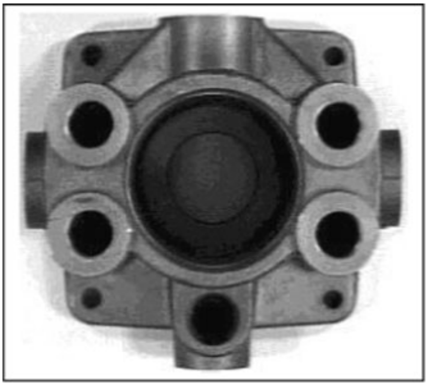
41.92 din

Format	*hin*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Monadic
Function	Double all intensities
Definition	C ⇐ 2∗A
Switch images	Yes
Typical uses	Increasing intensity/contrast, so that faint features can be seen more easily
Remarks	Also see *air*, *blo*, *hin*, *enc* and *heq*
Illustration	
Input image	Hydraulics component with *hin* applied, to generate a low-contrast image
Image generation	*rni(37), hin*
Command	*din*

Left: original



Right: after processing



41.93 dla

Format	*dla*(*a*, *b*, *c*)	
Arguments	*a*	X-coordinate of a point
	b	Y-coordinate of a point
	c	Angle (degrees, measured clock-wise relative to the horizontal)
Defaults	Not applicable	
Output values	None	
Image type	Any	
Type	Drawing	
Function	Draw a line at angle *c* through the point *[a, b]*.	
Switch images	Yes	

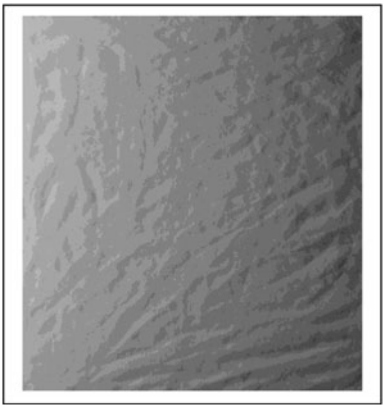
41.94 dmg

Format	*dmg*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey scale
Type	Image filter
Switch	Yes
Function	Direction of maximum gradient
Definition	Each pixel is assigned a value indicating the direction of the maximum of the absolute values of intensity difference between opposing pairs of 8-neighbours
Typical uses	Enhancing low-contrast intensity variations that are difficult to see otherwise
Remarks	It may be beneficial to apply a low-pass filter (e.g., *raf*, *caf*), or median filter (*mdf*) to attenuate camera noise before using *dmg*

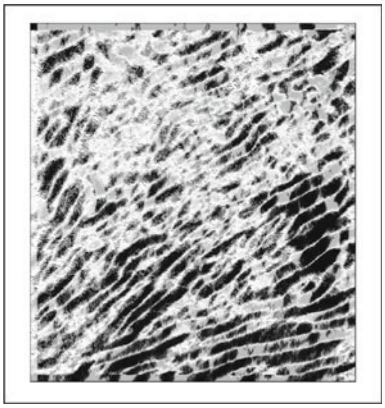
Illustration

Input image	Creased fabric (file number 28), followed by *enc*, *caf*
Image generation	*rni*(*28*), *enc*, *caf*
Commands	*dmg*, *enc*

Left: original



Right: after processing



41.95 doc

Format *doc*(*a*, *b*)

Arguments *a* Name of a QT M-function
b Selects the

Defaults No arguments Print the list of QT commands currently available
b 0, Select local PDF documentation file (PDF)
1, Visit author's web site for PDF documentation file

Output values None

Image type Not relevant

Type Not relevant

Function 1. $b = 0$, Read and display the documentation file (PDF format), held locally
2. $b \neq 0$, Read and display the contents of the PDF documentation file, held on the author's web server, for operator *a*
3. If no argument is specified, all QT command names are listed

Switch images No

Remarks The documentation files explain the function of, how to use and when to apply the given QT M-function. The significance of both input and output parameters is described. Where appropriate, default values for inputs are also defined. In many cases, images are included, to illustrate the operation of the function. The web browser, not MATLAB, will indicate when no appropriate documentation file can be found. In this case, return to MATLAB and try typing *hlp*(*a*) instead. This will indicate whether or not the specified M-function exists and will list it, if it does. Notice that the documentation file will be down-loaded onto your computer. For convenience, set the browser so that it automatically displays PDF files after down-loading

41.96 dpa

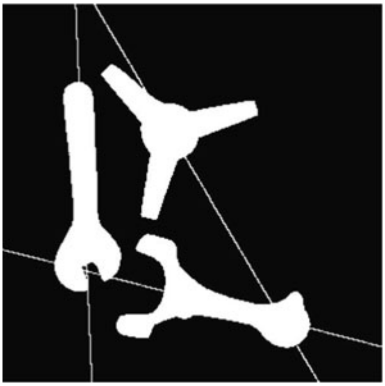
Format	$[x, y, z] = dpa$
Arguments	None
Output values	*x* X-coordinate of the centroid
	y Y-coordinate of the centroid
	z Orientation
Image type	Binary
Type	Feature extraction
Function	Draw the principal axis (axis of greatest second moment) of a single object in a binary image
Switch images	Yes
Typical uses	Visualising actions of a visually guided robot, picking up objects from a table
Remarks	Notice that *dpa* gives good results for objects that have an obvious elongation along one axis. However, an object such as a regular ‘star’ may give spurious results. (see below) Also see *lmi*, *cge*

Illustration

Input image	Four parts
File number	46
Command	*rni*(*46*), *big*(*4*), *exr*, *dab*(‘*dpa*’)

Left: original

Right: after processing



41.97 dps

Format *[a, b] = dps*

Arguments None

Output values *a* X-coordinate of the point selected by the user

b Y-coordinate of the point selected by the user

Image type Any

Function The user selects a point, *[a, b]*, with the cursor. This is then used to define the parameters for a shift with wraparound; the point *[a, b]* is shifted to the top-right corner of the output image

Switch images Yes

Typical uses Analysing binary images

Remarks Also see *cur*, *psw*

Illustration

Input image Stator

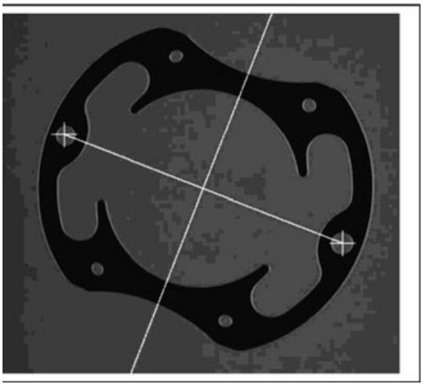
File number 174

Command *[a, b] = cur; [c, d] = cur; test(a, b, c, d)*

where test is defined as follows

```
function test(a, b, c, d)
global current
global alternate
original = current;
pbi(a, b, c, d);
dcl(a, b, 15)
dcl(c, d, 15)
vpl(a, b, c, d, 255)
alternate = original;
subplot(1, 2, 1), imshow(current)
subplot(1, 2, 2), imshow(alternate)
```

Image The user defined the two points indicated by crosses with the cursor

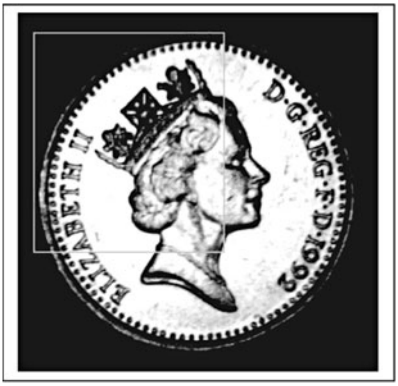


41.98 drd

Format	*drd(x, y, w, h, z)*	
Arguments	*x*	X-coordinate of the top-left corner of the rectangle
	y	Y-coordinate of the top-left corner of the rectangle
	w	Width of the rectangle
	h	Height of the rectangle
	z	Intensity of the rectangle
Defaults	*z*	255 (white)
Output values	None	
Image type	Grey	
Type	Graphics	
Function	Draw an open rectangle of intensity *z*, size *[w, h]* and whose top-left corner is at *[x, y]*.	
Switch images	Yes	
Typical uses	Annotating images	
Remarks	Also see *mar*, *dcd and dsc*	
Illustration		
Input image	Coin	
File number	23	
Command	*drd*(*12*, *15*, *135*, *156*, *128*)	

Left: original

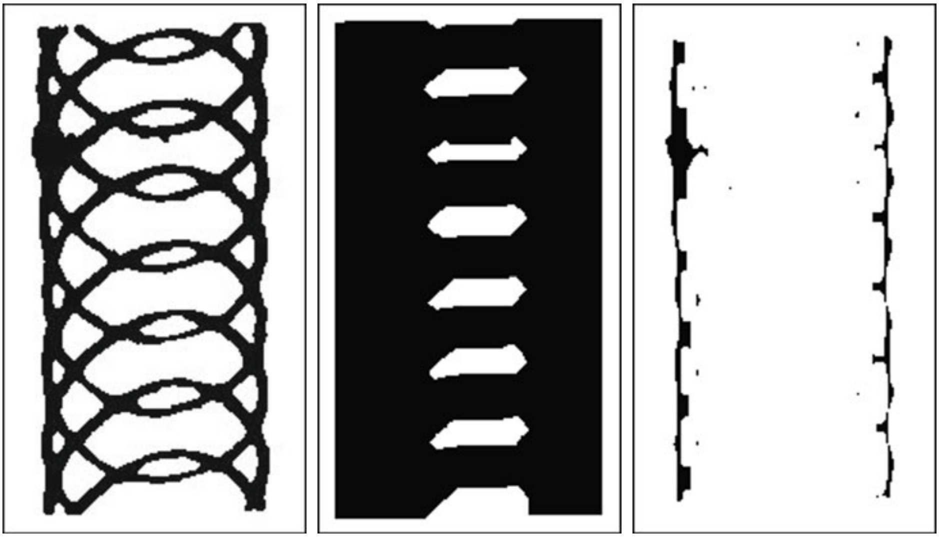
Right: after processing



41.99 dri/eri

Format	*dri*(*a*, *b*)/*eri*(*a*, *b*)	
Arguments	*a*	Structuring element width
	b	Structuring element height
Defaults	*a*	7
	b	1
Output values	None	
Image type	Binary	
Type	Morphology	
Function	Dilate/erode the image using a rectangular structuring element	
Switch images	Yes	
Typical uses	Feature detection	
Remarks	Also see *dil, erl, ero gbd, gbe, lnb, snb*	
Illustration		
Input image	Cake decoration pattern	
File number	55	
Commands	See below	

Left: original *Centre*: after *eri*(*31*, *5*) *Right*: after *dri*(*3*, *15*)



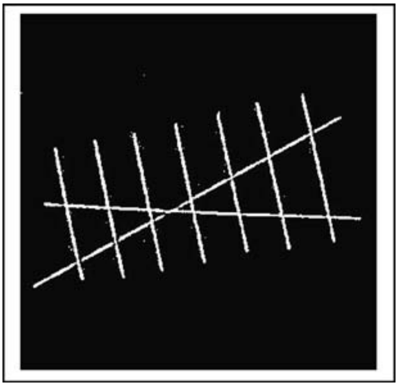
41.100 drp

Format	*drp*
Arguments	None
Defaults	None
Output values	None
Image type	Binary
Type	Image analysis
Switch	Yes
Function	Dominant Radon projection
Definition	First, the Radon transform is calculated. Then, the RT image is integrated in the vertical direction. The column which yields the peak integrated intensity is then identified. This indicates the angle by which the image must be rotated, so that the dominant feature is vertical
Typical use	Detecting linear streaks in noisy images
Remarks	Several short parallel lines, or one long single line, may contribute to the dominant peak in the integrated Radon transform

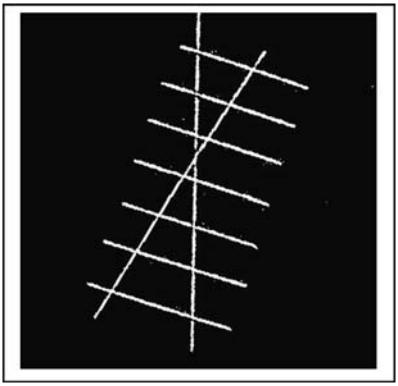
Illustration

Input image	Lines, with pre-processing to convert to binary image
Image generation	*rni*(*160*), *crk*, *thr*(*150*)
Command	*drp*

Left: original



Right: after processing



41.101 dsc

Format	*dsc*(*x*, *y*, *r*)
Arguments	*x* X-coordinate of the centre of the disc *y* Y-coordinate of the centre of the disc *r* Radius of the centre of the disc
Defaults	None
Output values	None
Image type	Any
Type	Image generation
Function	Draw a disc of radius *r* centred at *[x, y]*.
Switch images	Yes
Remarks	To draw an open circle, apply *bed* after *dsc*

41.102 dsz

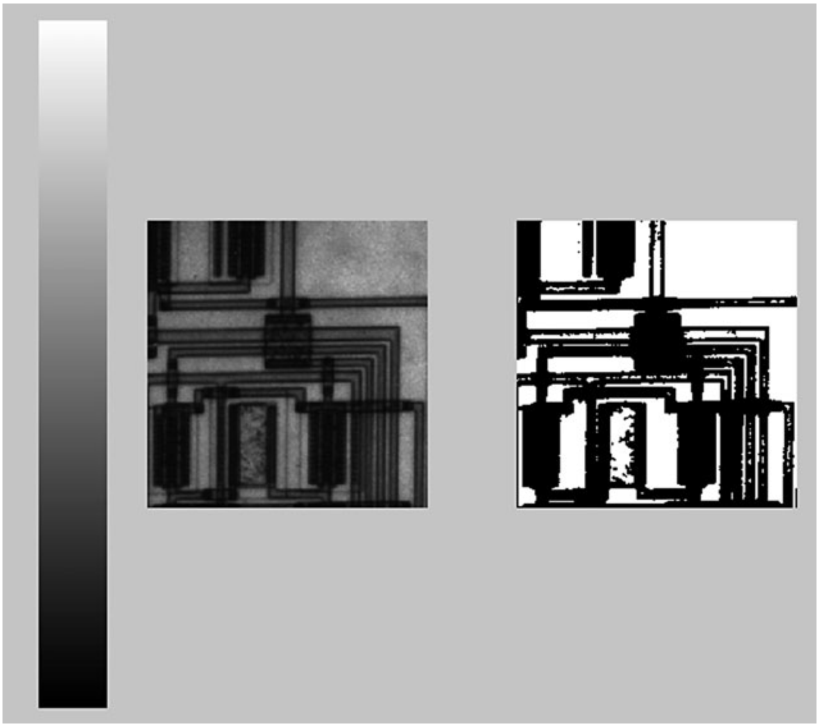
Format	*r*=*dsz*
Arguments	None
Output values	*r* Size of smallest blob to be retained
Image type	Binary
Function	The user selects a size limit (*r*) using the cursor. (The cursor is placed over a grey-scale wedge. so that the user can estimate the size of blobs to be eliminated by relating size with intensity.) Blobs of size smaller than *r* pixels are then eliminated from the current image. Blobs bigger than 255 pixels are always retained
Switch images	Yes
Typical uses	Removing noise from binary images
Remarks	Also see *cur*, *dth*, *kgr*, *kbb*

41.103 dth

Format	r = *dth*(*a*, *b*)
Arguments	None
Output values	r Intensity value
Image type	Grey
Type	Dynamic threshold
Function	Threshold the input image using the cursor to select the threshold parameter
Switch images	Yes
Typical uses	Image segmentation (i.e., isolating an object silhouette from its background). Reducing data content in an image, to make subsequent analysis easier
Remarks	*dth* is an interactive thresholding operator and is useful for exploring the effects of varying the threshold parameter Also see *hgm*, *thr* and *tha* – *thg*

Illustration

The image below is typical of that displayed in *Figure 1* during the execution of *dth.* The user double clicks in the colour bar to select the threshold parameter. The result is shown, almost immediately, in the Alternate Image (on the right). The user can then select a new threshold, or double click elsewhere in *Figure 1* to finish. The final threshold value selected in this way is then returned by *dth.* The program then switches the images, so that the resulting thresholded image is displayed in the Current Image. As usual, the original is displayed in the Alternate Image.

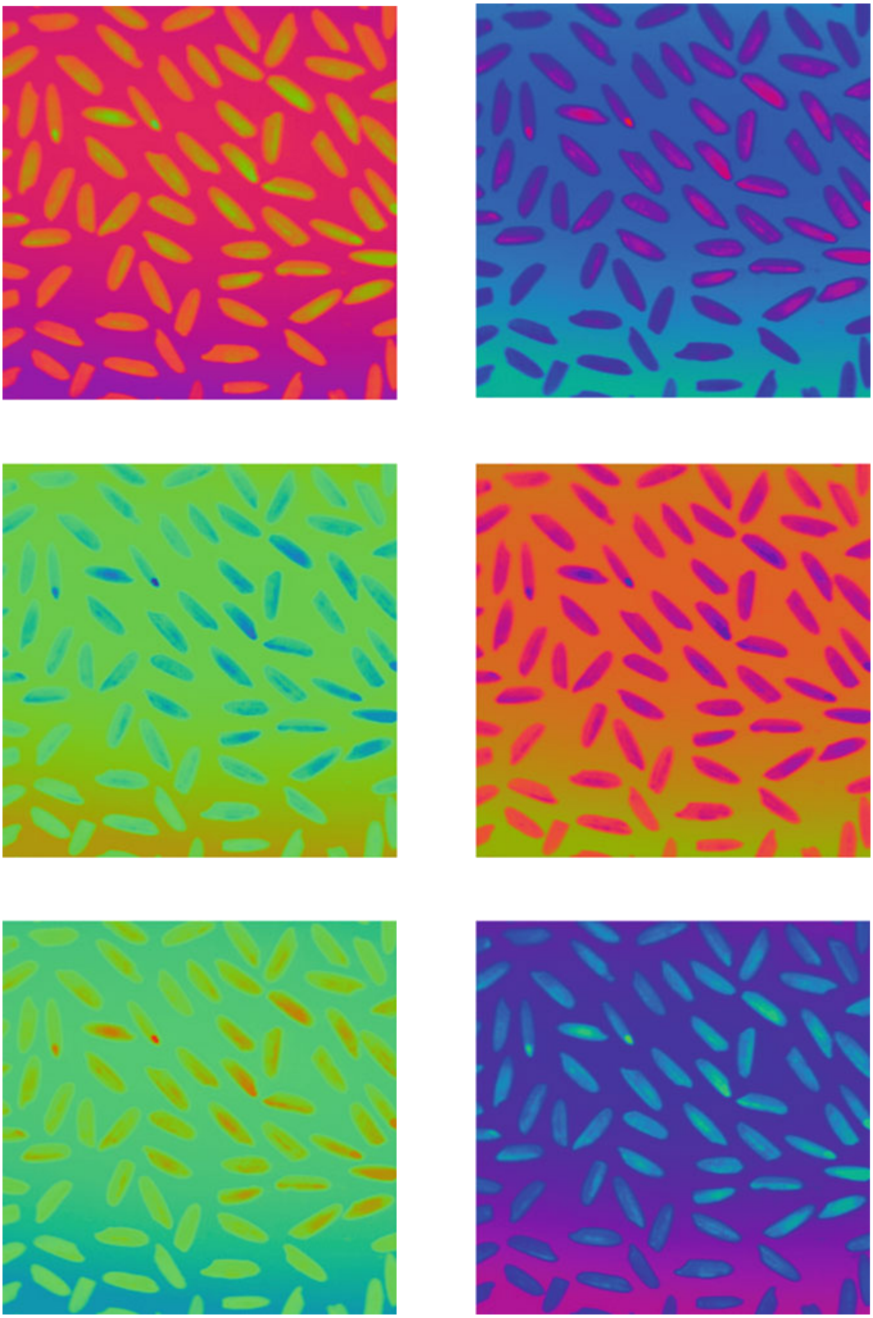


41.104 ecc

Format	*ecc*(*a*, *b*, *c*)	
Arguments	*a*	Colour channel *a* defines new R channel
	b	Colour channel *b* defines new G channel
	c	Colour channel *c* defines new B channel
Default values	*a* = 2 (R channel takes values originally specified by the G channel)	
	b = 3 (G channel takes values originally specified by the B channel)	
	c = 1 (B channel takes values originally specified by the R channel)	
Output values	None	
Image type	Colour	
Type	Monadic	
Function	The RGB channels are exchanged	
	ecc(1, 2, 3)	No change
	ecc(2, 1, 3)	R and G channels are interchanged. B channel is not changed
	ecc(1, 3, 2)	G and B channels are interchanged. R channel is not changed
	ecc(3, 2, 1)	R and B channels are interchanged. G channel is not changed
	Further examples are illustrated below	
Switch images	Yes	
Typical uses	Increasing contrast produced by the pseudo-colour operator (*psc*)	
Remarks	This function has little practical value when processing colour images. However, it is great fun to use; it often produces interesting surreal effects, especially on faces. Also see *aic*	

Illustration

Input image	Rice grains
File number	73
Commands	
Top-left	*rci(73), enc, psc*
Top-right	*rci(73), enc, psc, ecc(2, 3, l)*
Centre-left	*rci(73), enc, psc, ecc, ecc(3, l, 2)*
Centre-right	*rci(73), enc, psc, ecc(l, 3, 2)*
Bottom-left	*rci(73), enc, psc, ecc(2, l, 3)*
Bottom-right	*rci(73), enc, psc, ecc(3, 2, l)*



41.105 ecy

See *bpa.*

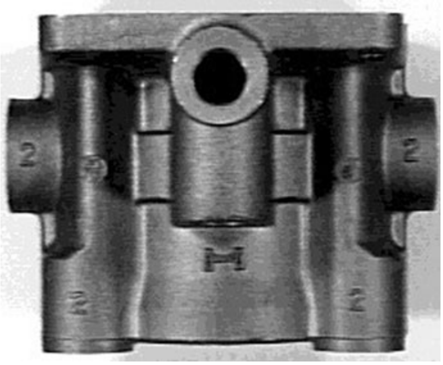
41.106 edd

Format *edd*(*a*)
Arguments *a* Function selector
Defaults *a* 0
Output values None
Image type Grey
Type Edge detector
Function Morphological edge detectors
Definition See MATLAB listing below
Switch images Yes
Typical uses Highlighting edges (i.e., regions of large intensity gradient)
Remarks A range of edge detectors can be constructed by subtracting a dilated/eroded image from the original:

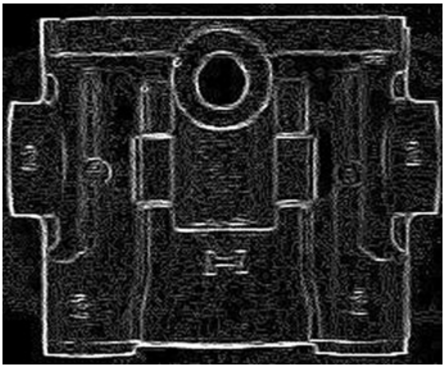
Input parameter *a*	Function
0	Contour is on dark side of dark-bright boundary
1	Contour is on bright side of dark-bright boundary
2	2-pixel wide contour spans dark-bright boundary

Illustration
Input image Hydraulics component
File number 38
Command *edd*

Left: original



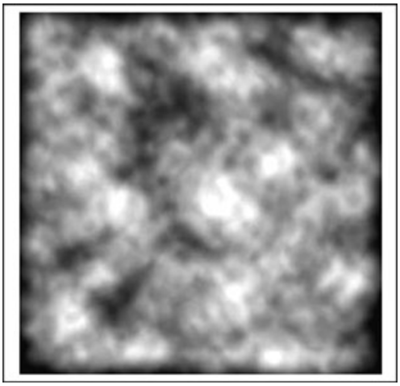
Right: after processing



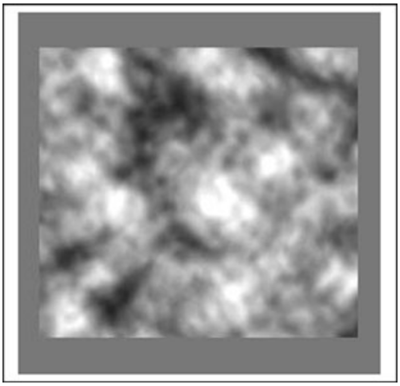
41.107 edg

Format	*edg*(*a*, *b*, *c*)	
Arguments	*a*	Width of the border at the sides of the image
	b	Width of the border at the top and bottom
	c	Intensity value of the border
Defaults	No arguments:	*a = 1*; *b = 1*; *c = 0*
	One argument:	*b = a*; *c = 0*
	Two arguments:	*b = a*; *c* is equal to the second argument supplied
Output values	Not applicable	
Image type	Any	
Type	Drawing	
Function	Set the edge pixels to a defined value	
Definition	A band of pixels of width *a* is set to level *c* at the left- and right-hand sides of the image. Bands of width *b* are also set to this level at the top and bottom	
Switch images	Yes	
Typical uses	Masking edge effects created by local and morphological operators, such as *sed, caf, dil*, etc.	
Remarks	Also see *rbe*	
Input image	Coffee beans (file number 4), after applying a severe blurring filter (*caf*)	
Image generation	*rni*(*4*), *enc*, *caf*(*15*), *enc*	
Command	*edg*(*15*, *25*, *128*)	

Left: original (Notice the edge effects)



Right: after processing



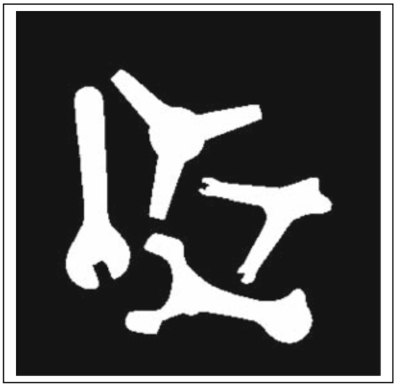
41.108 edx/lrx/tbx

Format	*lrx* *tbx* *edx*
Arguments	None
Defaults	Not applicabale
Output values	None
Image type	Grey
Type	Monadic
Function	*lrx:* Row maximum, going from left-to-right. Result is *I1* Row maximum, going from right-to-left. Result is *I2* Combine *I1* and *I2* using *mni* *tbx:* Column maximum, going from top-to-bottom. Result is *I3* Column maximum, going from bottom-to-top. Result is *I4* Combine *I3* and *I3* using *mni* *edx:* Combine images *I1*, *I2*, *I3* and *I4* using *mni*
Switch images	Yes
Typical uses	Compensating for uneven lighting by removing slow intensity changes in the back-ground
Remarks	These are three rather crude operators. However, they do occasionally produce interesting and useful results. Illustration I shows how these operators behave on a simple binary image. Illustration II demonstrates the practical benefit on a simple grey-scale image. Notice how the background shading is improved by subtracting the result of *edx* from the original image, thus making thresholding simpler

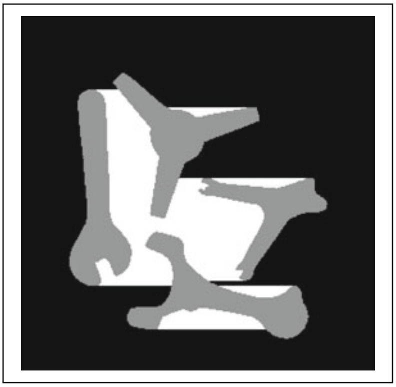
Illustration

Input image	4 piece parts
File number	46
Commands	*lrx* *tbx* *edx*

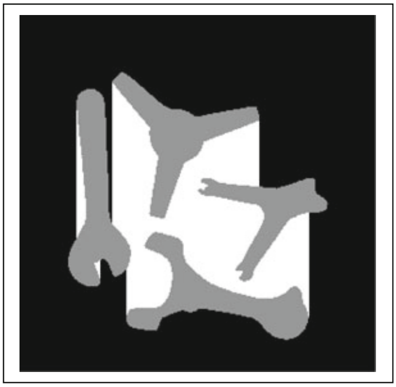
Top left: original



Top right: after *lrx*



Centre left: after tbx



Centre right: after *edx*

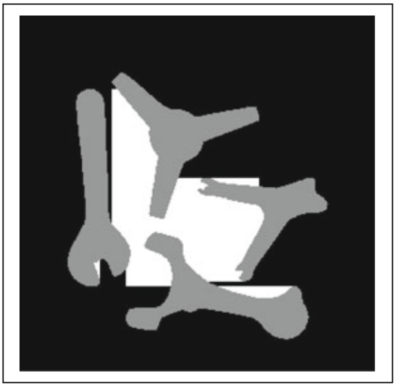
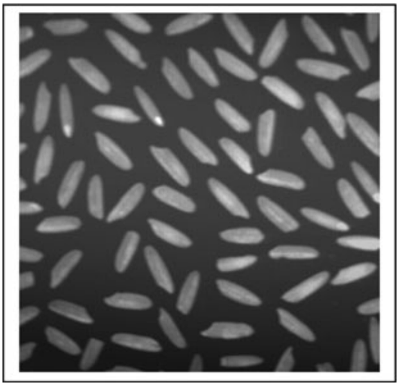


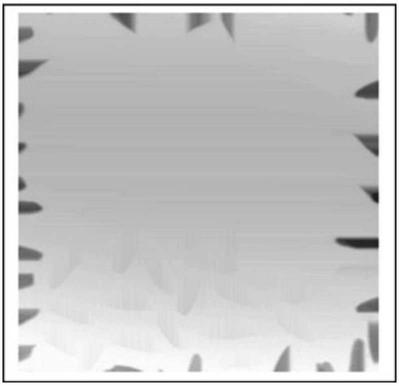
Illustration II

Input image	rice
File number	3

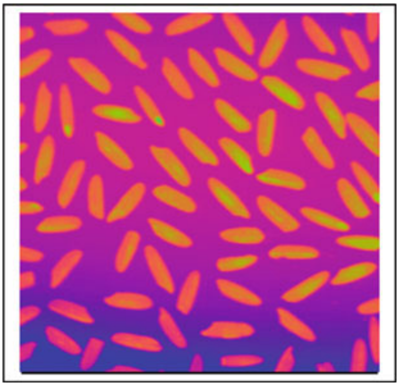
Top left: original



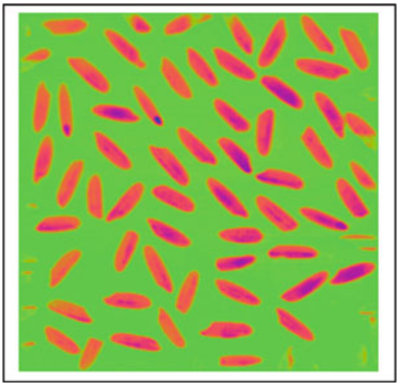
Top right: after *enc*, *neg*, *edx*



Centre left: after *enc*, *psc*



Centre right: after *enc*, *neg*, *edx*, *sub*, *enc*, *psc*



41.109 emn

See *emx*.

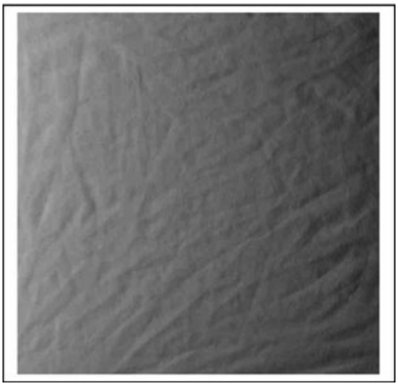
41.110 enc

Format	*enc*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Monadic
Function	Enhance contrast by applying a shift and linear stretch to the intensity scale. In the resulting image, the darkest pixel is black and the brightest pixel is white
Definition	$C \Leftarrow 255*(A - A_{min})/(A_{max} - A_{min})$
Switch images	Yes
Typical uses	Increasing contrast
Remarks	*enc* is functionally equivalent to *[a, b] = gli, air*(*a, b*). Notice that *enc* performs a linear stretching of the intensity scale, while *heq* is non-linear. Also see *blo*, *acn*, *din*, *hin*

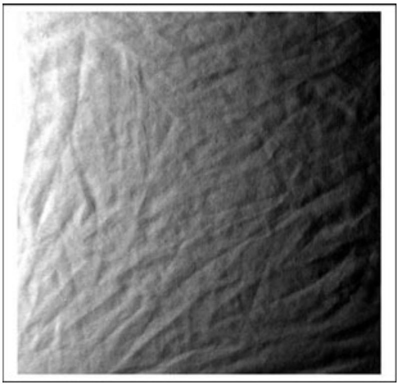
Illustration

Input image	Creased fabric
File number	28
Command	*enc*

Left: original



Right: after processing



41.111 eqd

See *bpa*.

41.112 eri

See *dri*.

41.113 ero

See *dil*.

41.114 ersf

Format *ersf*(*a*, *b*, *c*)

Arguments *a* URL for a remote script file

b Time limit (in seconds). Execution automatically terminates after *b* seconds

c Time delay (in seconds) before the next cycle begins

Defaults None

Output values None

Image type Any

Type Program control

Function Execute the script in a defined file (URL *a*), repeatedly, until a specified time limit (*b*) has been reached. At the end of each cycle, wait for a defined time (*c*), before executing the script again. The program recovers automatically if a faulty script is defined and then corrected

Switch images Undefined; operation depends on the script defined by *a*

Typical uses Prototype interfacing function

Remarks The user can modify the function(s) being executed, at any time, by altering the contents of the file specified by input parameter *a*. This is the URL for a script file, which may held be locally (i.e., same computer as QT), or on a remote computer. If the first parameter (*a*) is simply a number, the file name is built automatically and a local file is consulted. In either case, the contents of the script file can be changed 'on the fly', either by a remote user with a simple text editor, or an autonomous program, running asynchronously with MATLAB. Consider the command

```
ersf('file:///Users/bruce/MATLAB/QT/QT_work_files/script1')
```

The following actions ensue:

(i) Start the timer. This will run from this moment for *b* seconds

(ii) Read the file specified by parameter (*a*)

(iii) Execute the command sequence in file *script1*

(iv) Display the results in MATLAB's *Command Window*

(v) Wait for *c* seconds before continuing

(vi) One of two possible actions follows next:

(a) If the time delay specified by parameter *b* has not yet been reached, return to step (ii) and repeat

(b) Otherwise terminate execution

Notice that execution can be terminated at any time by the user pressing CTRL/C

Example

Command

```
ersf('file:///Users/bruce/MATLAB/QT/QT_work_files/script1',
10,0.5)
```

URL of script file

'file:///QT_work_files/script4532'

Contents of that script file

```
grb(3); enc; thr(200); dil; blf; exr; big; blf; [x, y] = cgr;
display(['Centroid:', int2str([x, y])]);
display(['End of loop']);
```

Output

```
Centroid: 316 228
End of loop
Centroid: 296 187
End of loop
Centroid: 265 168
End of loop
Timer has signalled that the time limit allowed for execution has been
reached – this loop will be completed before execution stops
Centroid: 242 198
End of loop
```

41.115 esr

Format [*c*, *d*] = *esr*(*a*, *b*)

Arguments *a* Synchronisation code (integer, not used internally)
b Command sequence to be evaluated

Defaults *a* *12345*

Output values *c* Results vector. Consists of 7 elements:
1 No. of valid output values generated
2–7 Results obtained by performing operation *b*
d Output results string

Image type Any

Type Program control

Function Execute a command defined by *b*. The following results are then stored in the folder specified by the global variable *results_url*:

Current image	File *current.jpg*
Alternate image	File *alternate.jpg*
Results, formatted text	File *results* (See example below)

Switch images Undefined. Operation depends on the operation defined by *b*

Typical uses Inspecting the results of a free-running QT program using web browser
Interfacing to another program

Remarks Normally, *esr* runs inside an infinite loop. (Execution can be terminated at any time by the user pressing CTRL/C.) *esr*(*a*, *b*) performs operation *b*. Outputs *c* and *d* are made available for debugging purposes but would not normally be used otherwise. The resulting *current* and *alternate* images and a text file containing the time-stamped, numeric results of that operation are then saved. These three files can then be examined using a standard web browser. Viewing the results does not affect the execution cycle. Alternatively, the results of operation *b* can be analysed further, by another program (*P*), running asynchronously locally, or on a remote computer. In order to facilitate synchronisation between QT and *P*, the first parameter (*a*) is simply copied into the text file (*results*). When the command

esr(*3582*,‘*mar*’)

was executed. the following data was generated and stored in the *results* file:

Code:	3582						
Command:	mar						
Date and time:	2004	11	25	16	51	48	
Results:	4	1	13	2	14	0	0

(The first element in this list indicates that there are 4 valid answers (1 13 2 14) for the operation *mar.*) Notice that commands that calculate numeric answers (e.g., *mar*, *avg*, *gli*, *cbl*, *cwp*, *eul*, etc.) are included in the call, without specifying output parameters. (The command *esr*(*3582*, ‘[*a*, *b*, *c*, *d*] = *mar*’) works perfectly satisfactorily but does not assign values to any output variables (*a*, *b*, *c* or *d*).) Individual QT commands that do not calculate output values (e.g., *neg*, *thr*, *enc*, *heq*, etc.) generate output lists in *results* consisting of seven zeroes. Compound commands are permitted but commands that return values must be executed individually. Thus, we may use

esr(*123*, ‘*rni*(*4*), *enc*, *thr*, *dil*, *ero*’)

However, the command

esr(*123*, ‘*rni*(*4*), *enc*, *thr*, *dil*, *ero*, *cbl*’) performs the desired calculation but does not return any significant values. Instead, we should use two commands:

esr(*123*, ‘*rni*(*4*), *enc*, *thr*, *dil*, *ero*’), *esr*(*124*, *cbl*)

41.116 eul

Format	$b = eul(a)$
Arguments	a Connectivity (4- or 8-connected. Hence, $a \in \{4, 8\}$)
Defaults	a 8
Output values	b Euler number
Image type	Binary
Type	Binary image measurement
Function	Calculate the Euler number for a binary image
Switch images	No
Typical uses	Counting blobs or lakes
Remarks	The Euler number is the number of blobs minus the number of holes. Hence, if it is known that there are no holes, the number of blobs can be estimated. On the other hand, if there is just one blob, then the number of holes can be found. Notice that *cbl* computes the number of blobs directly

41.117 exm

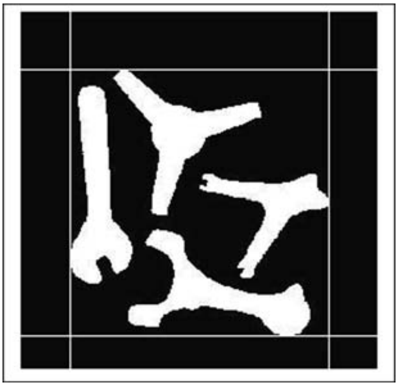
Format	$[x1, x2, y1, y2] = exm(a)$	
Arguments	*a*	Control
Defaults	*a*	1 (Draw *X* and *Y* limits)
Output values	*x1*	Minimum value of *X*
	x2	Maximum value of *X*
	y1	Minimum value of *Y*
	y2	Maximum value of *Y*
Image type	Binary	
Type	Image measurement	
Switch	Yes	
Function	*a = 1*:	Calculate and draw the minimum and maximum *X* and *Y* bounds, considering all white points
	a = 2:	Calculate and draw the minimum and maximum *X* and *Y* bounds. Then, crop the image at these limits
Switch images	Yes	
Typical uses	Reducing execution time for subsequent by limiting processing to a region of special interest	
Remarks	*exm* can find the limits for multi-blob images, whereas the related functions *bbx* and *mar* operate on just one blob at a time. Use the following command sequence to calculate the *X*- and *Y*-limits from a binary image and then crop the grey-scale image from which it was derived: $thresh, [x1, x2, y1, y2] = exm(0), swi, crp(x1, y1, x2 - x1, y2 - y1)$ The unspecified function *thresh* might, for example, be *thr*(*Q*), where *Q* is some positive integer. The important point to note is that the grey-scale image is held in the alternate image and the binary image derived from it is in the current image	

Illustration

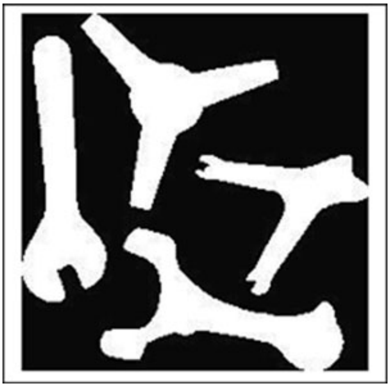
Input image	Four metal parts
File number	46
Command	*thr*, *exm*(*1*) and *thr*, *exm*(*2*)

Top-left: original

Top-right: after *thr*, *exm*(*1*)



Bottom-left: after *thr*, *exm*(*2*)



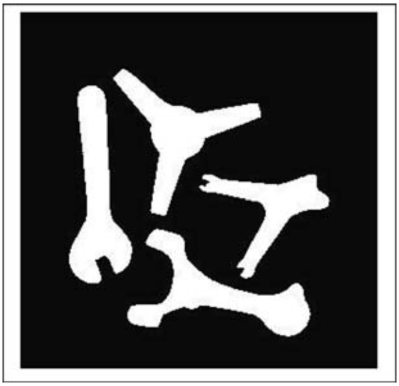
41.118 exr

Format	*exr*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Dyadic
Function	Exclusive OR of pixels at corresponding addresses in the current and alternate images
Switch images	Yes
Typical uses	Comparing binary images
Remarks	Applying the following command sequence selects the *second* largest blob in a binary image: *big*, *exr*, *big*

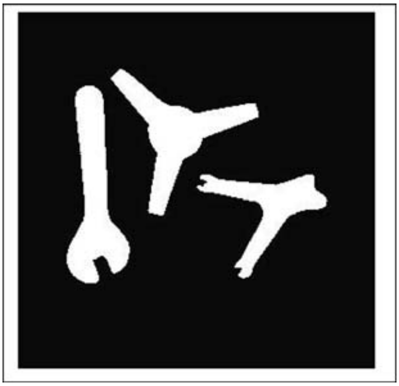
Illustration

Input images	Four metal components and the same image with the largest blob isolated using *big*
File number	46
Command	*big, exr*

Left: original



Right: after processing



41.119 ext

See *bpa.*

41.120 fap

Format	*fap*(*x*, *y*, *z*)	
Arguments	x	Vector containing X-coordinates of pixels to be set to level z
	y	Vector containing Y-coordinates of pixels to be set to level z
	z	Vector containing intensity values
Defaults	None	
Output values	None	
Image type	Grey or binary image is generated	
Type	Data display	
Function	Set the pixels addressed by the vectors *x* and *y* to intensity values defined by *z*. All other pixels are set to black	
Switch images	Yes	
Typical uses	Converting edge contour data (e.g., filtered chain code) into an image	
Remarks	Also see *gap*	

41.121 far

See *bpa*.

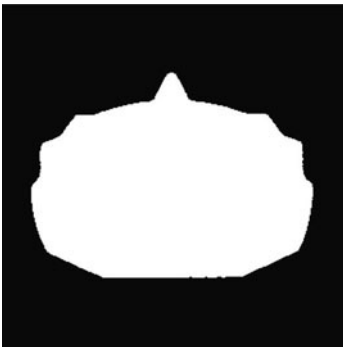
41.122 fbl

Format	*n* = *fbl*
Arguments	None
Output values	*n* Rotation angle that achieves the best fit
Image type	Binary
Type	Binary image analysis
Function	1. The current and alternate images are both normalised so that the centroid of each image is moved to the middle of the image 2. The current image is then rotated so that it achieves the best fit on the alternate image (Angle *n*) 3. The angle for best fit defines the output variable
Switch images	Yes
Typical uses	Fitting a binary template
Remarks	

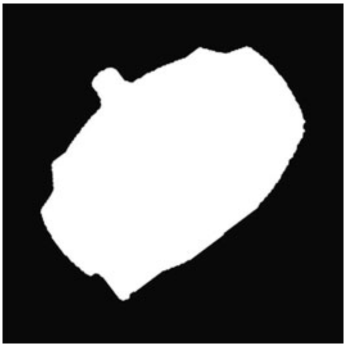
Illustration

Input image	Automobile break pads
File number	40 (split in two, to separate the brake pads)
Command	*fbl, adi*

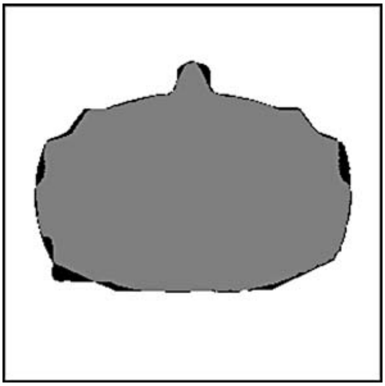
Top-left: binary image: metal base-plate for an automobile brake pad. Call this Part A



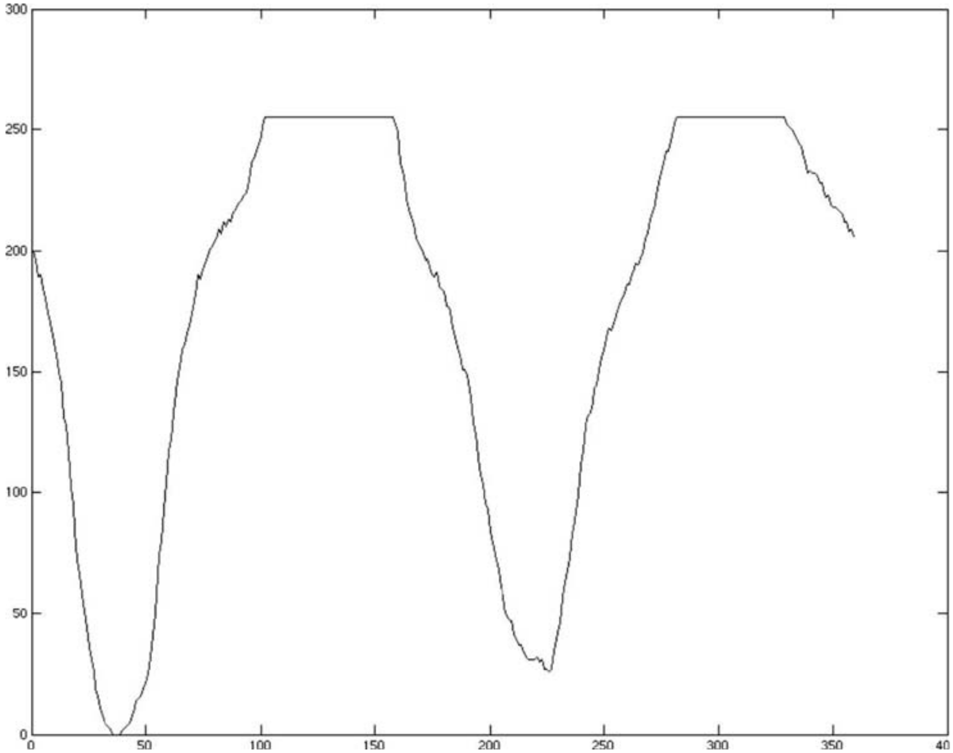
Top-centre: binary image from another slightly different object, Part B



Centre: after applying [*rei*('*B*'), *rei*('*A*'), *fbl*], Part B has been rotated and shifted so that it 'fits' Part A
Black areas indicate the differences between the two objects



Bottom: goodness of fit (ordinate), plotted against angle of rotation (abscissa). This graph is plotted in an additional MATLAB window: *Figure 5*



41.123 fcc

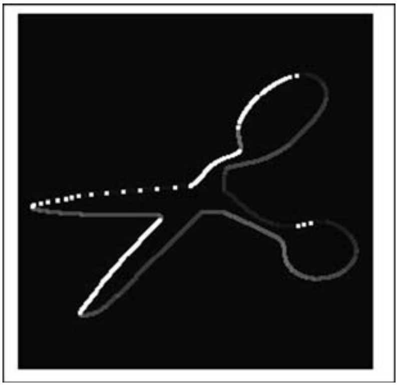
Format	*[x, y, z] = fcc*	
Arguments	*None*	
Output values	*x*	Vector containing X-coordinates of edge pixels
	y	Vector containing Y-coordinates of edge pixels
	z	Vector of chain-code values
Image type	Binary	
Type	Blob transform	
Function	Shade the image to indicate the chain-code value for each edge pixel. The chain code indicates the path taken to exit that pixel to visit an adjoining edge pixel. The edge follower travels clock-wise around the blob. All pixels other than those on the edge are set to black. The starting point is the top-left-most, non-black pixel. Only the largest blob is analysed and the edges of any lakes are ignored; other blobs and/or lakes must be analysed separately	
Switch images	Yes	
Typical uses	Analysing blob shape. Further analysis is performed on the vector z, not the image	
Remarks	A single blob can be processed using *seb*. If any other blobs are present, the result is an error. Other blobs and any lakes must analysed separately. Also see *seb*	

Illustration

Input image	Scissors (after applying *thr* to eliminate JPEG artifacts)
File number	49
Commands	*thr*, *fcc*, *lnb* (*ln*b broadens the edge contour, making it easier to see)

Left: original

Right: after processing



41.124 fce

Format *[x, y, r] = fce(u, v, w)*

Arguments *u* Vertical position of first scan line, relative to the top of the blob

v Vertical position of second scan line, relative to the first one

w Vertical position of third scan line, relative to the second one

Defaults Not applicable

Output values *x* X-coordinate of centre of the fitted circle

y Y-coordinate of centre of the fitted circle

r Radius of the fitted circle

Image type Binary

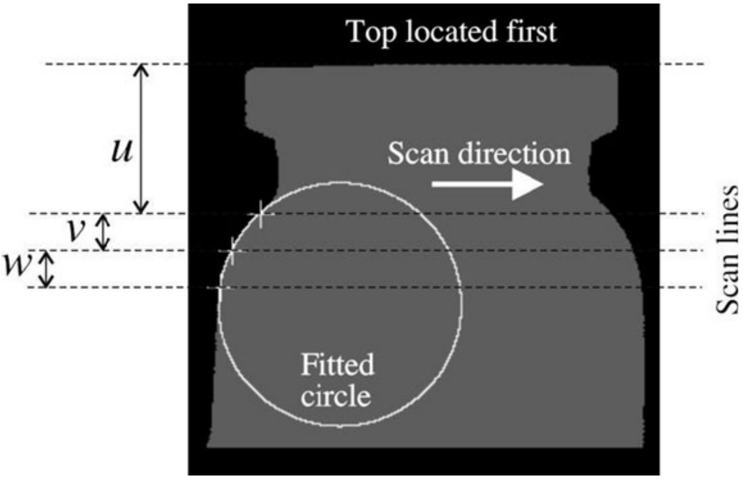
Type Drawing

Function Fit a circle to 3 edge points, found by scanning along 3 rows of the image

Switch images Yes

Typical uses Inspecting bottles, loaves, where radius of curvature is to be calculated

Remarks The algorithm used in this function is explained in the following diagram. First, the top of the blob is found. Then, the 3 scan lines are positioned *u*, ($u + v$) and ($u + v + w$) lines below it. The left-most intersection of each scan-line with the blob is then found. This identifies three edge points, which are then used for fitting the circle. Finally, the circle is drawn

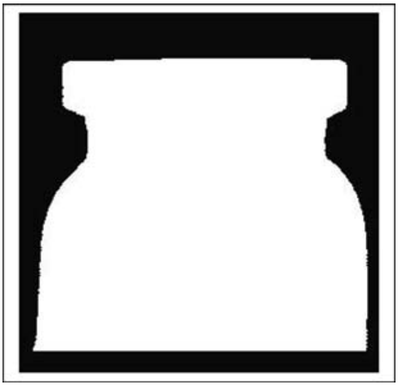


Also see *rbe* and *fcd*

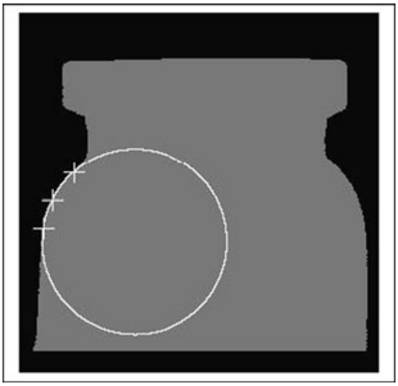
Illustration

Input image	Glass vial (file number 32) with processing to create binary silhouette
Image generation	*rni*(*32*), *rox*, *wri*, *swi*, *lrt*, *rox*, *lrt*, *rei*, *mni*, *thr*(*16*)
Command	*fce*(*90*, *22*, *22*)

Left: original



Right: after *fce*(*90*, *22*, *22*)



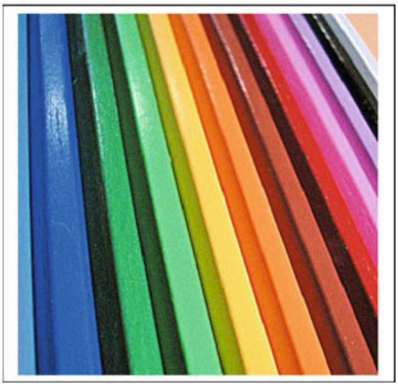
41.125 fcf

Format	*fcf*(*a*)	
Arguments	*a*	Function selection
Defaults	*a*	1
Output values	None	
Image type	Colour	
Type	Colour filter	
Function	Apply one of a set of standard programmable colour filters	
Switch images	Yes	
Typical uses	Segmentation of colour images	
Remarks	See *pcf* for a more detailed explanation	

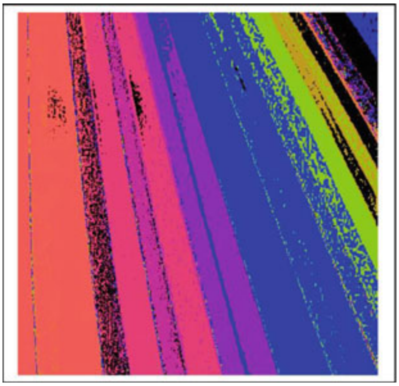
Illustration

Input image	Pencils
File number	270
Command	*fcf*(7)

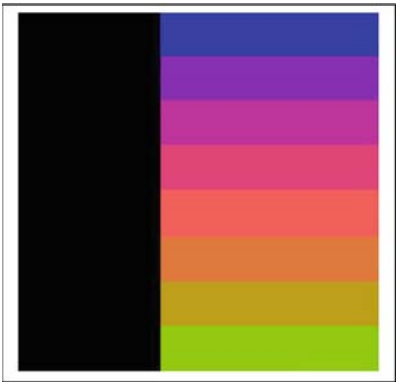
Top-left: original



Top-right: after processing



Bottom-left: mask image



41.126 fcl

Format *[x, y, r] = fcl*

Arguments None

Output values *x* X-coordinate of the centre of the circle

y Y-coordinate of the centre of the circle

r Radius of the circle

Image type Binary

Type Binary image analysis

Function Fit a circle to a set of points, using the least-squares algorithm

Switch images Yes

Typical uses Analysing binary images with 'broken' edge contours derived from circular features

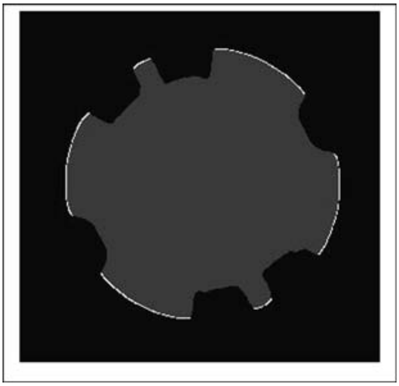
Remarks In the illustration below, a set of disjoint arcs was derived from the outer edge contour of a metal cam, using the perimeter of its convex hull as a mask. The circle found by *fcl* provides an accurate approximation to the outline of the cam

Illustration

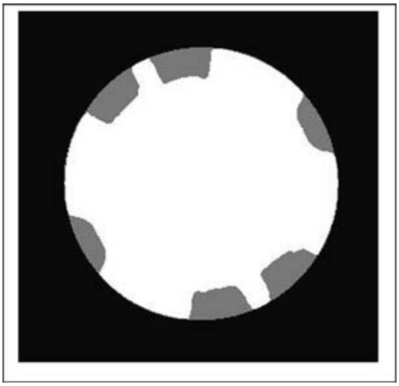
Input generation *rni*(*169*), *enc*, *thr*, *neg*, *blf*, *bed*, *wri*, *swi*, *chu*, *bed*, *rei*, *mni*

Commands *fcl*, *adi*, *mxi*, *sqr* (Generated the white arcs in the image below, left)

Left: original image (white) superimposed on the cam.



Right: circle fitted by *fcl* using the least-squares algorithm, superimposed on the cam



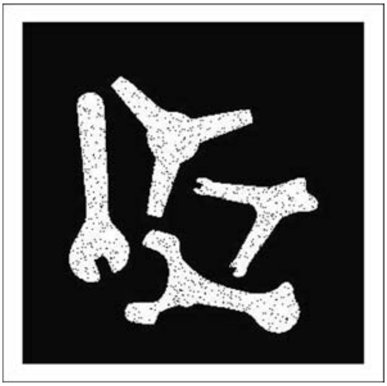
41.127 fib

Format	*fib*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Binary image filtering
Switch	Yes
Function	Fill isolated black pixels
Switch images	Yes
Typical uses	Noise reduction in binary images
Remarks	When there are very few black pixels, it may be advantageous to use *fib* rather than *blf*

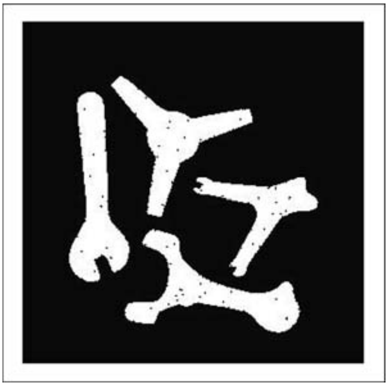
Illustration

Input image	Four metal parts with salt-and-pepper noise added
Image generation	*rni*(*46*), *spn*(*0.1*), *mni*
Command	*fib*

Left: original



Top-right: after processing



41.128 force

Format *force*(*a*, *b*)

Arguments *a* First command/command sequence

b Second command/command sequence

Defaults None

Output values None

Image type Not relevant

Type Program control

Function Execute command/command sequence *a*. If this generates an error, then execute command/command sequence *b*

Switch images Depends upon input arguments

Typical uses Useful when interfacing QT to other software, as the outcome can be predicted

Remarks The following are all valid commands

orce(‘rni(4)’,‘qwerty’) % Nonsense second parameter

force(‘qwerty’,‘neg’) % Nonsense first parameter

x = ‘rni(34)’; y = ‘qwerty’; force(x,y) % Command sequence

x = ‘rni(34)’; y = ‘qwerty’; force(y,x) % Command sequence

x = ‘rni(34),neg,thr’; y = ‘qwerty’; force(y,x) % Assigned variables

force(‘qwerty’,‘ ’) % Successful execution (but does nothing)

41.129 fov

Format *fov*

Arguments None

Output values None

Image type Not applicable

Type Design tool (*fov* is not intended to be used with other QT commands)

Function *fov* (*field-of-view Calculator*) is a design tool that estimates the following parameters for a target vision system:

Function *Field-of-view Calculator. fov* is a design tool that estimates the following parameters for a target vision system:

- The minimum dimensions of the camera's field of view
- The optical magnification required to achieve this
- The pixel size

The width of the smallest linear feature that can be detected reliably. The target system is assumed to perform inspection/measurement functions on piece-parts (widgets) that produce a reasonably high positive contrast with their back-ground. During inspection by the target system, the widgets can be in any orientation but are limited to lie within a defined positional tolerance region [±δx, ±δx]. The user is asked to specify the following input values via the key-board:

- Diameter of the calibration disc (mm, explained below)
- Camera resolution (pixels; the sensor array is assumed to be square)
- Camera sensor size (measured along its diagonal, mm)
- δx, positional tolerance along the X-axis (mm)
- δy, positional tolerance along the Y-axis (mm)

The user is not asked to specify the dimensions of the widget, as these are derived automatically by the QT software. Instead, the user presents the following to QT's camera:

- A circular disc of known diameter (This is called the *Calibration Disc*)
- A sample widget

The calibration disc should be about the same size as the widget (An exact match is not critical and can be judged by eye)

Sample dialogue and output

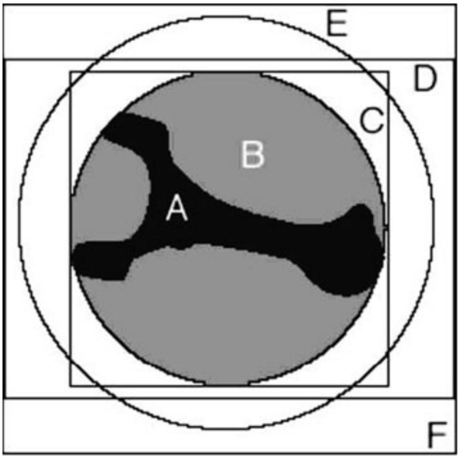
```
FIELD OF VIEW CALCULATOR

Diameter of the calibration disc (mm):157
Camera resolution (pixels):           512
Camera sensor size (diagonal, mm):    6.25
Delta X (mm):                         25
Delta Y (mm):                         5
```

Place the calibration disc fully within the field of view of the camera All activity is suspended until the user clicks on the 'OK' button Place a sample widget fully within the field of view of the camera All activity is suspended until the user clicks on the 'OK' button

RESULTS	
Field of view:	102.75 × 102.75 mm^2
Magnification required:	16.44
Pixel size:	0.20068 × 0.20068 mm^2
Minimum feature width:	0.80273 mm

Image generated by fov



A (black blob):	Widget silhouette
B (inner circle):	Minimum-area-circle enclosing the widget
C (inner rectangle):	Minimum-area-rectangle enclosing B
D (outer rectangle):	Essential pixels
E (outer circle):	Calibration disc
F (outermost square):	Field of view required for square sensor array

N.B. Those pixels in F but not in D are 'wasted' as they will never be used. In this example, only 9.2% of the field-of-view (F) is covered by the widget

41.130 frz

Format	*frz*(*a*, *b*)
Arguments	*a* Switches conversion to grey-scale ON/OFF
Defaults	*a* *1* (*Do not convert to a grey-scale image*)
Output values	*None*
Type	Image acquisition
Function	Acquire an image from a free-running USB camera
Switch images	Yes
Remarks	The operator *frz* illustrates how images from an asynchronous camera may be captured by QT. The camera driver software is set up to load a new image file, called ‘*camera.jpg*’, in the folder ‘QT_work_files’ periodically (typically every second, or so). The previous image file with that name is overwritten. The camera is not synchronised to QT/MATLAB. The author uses a *DinoEye* USB camera (AnMo Electronics, URL *http://www.dino-lite.com*) in conjunction with *EvoCam* software (Evological, URL *http://www.evological.com/*)

41.131 ftm

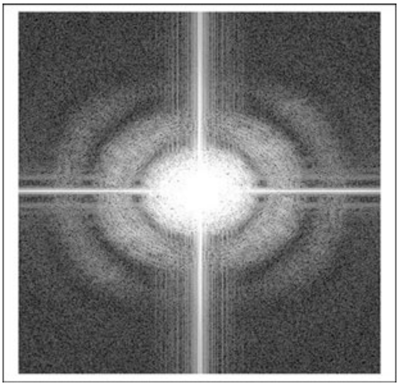
Format	*ftm*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey scale or binary
Type	Integral transform
Switch	Yes
Function	Log-magnitude Fourier transform (FT)
Typical uses	Analysis of particle size, periodic image structures, texture
Remarks	This version of the Fourier transform is intended as a means of describing images. Other applications, such as correlation, filtering and deblurring are better performed in MATLAB, since they require the use of complex numbers, which the designer of QT has ignored. The Fourier phase image can be calculated by changing line 5 to $a = angle(fft2(current));$ Two examples are shown below. In the first one, the FT of a spiral image, derived from an internal screw thread using a conical mirror, is shown. The spacing between the circular rings indicates the thread pitch. The second example shows the isophotes of the FT derived from the image of a pile of coffee beans. The FT image is merely a bright fuzzy spot, whose diameter is an indicator of the size of the coffee beans. This can be estimated by calculating the area enclosed within one of the contours (function *iso*), or by measuring the width of the peak in the intensity plot (*plt*). In both examples, the median filter (*mdf*(*7*)) was applied before *ftm*, to reduce the effects of noise

Illustration

Input image	Screw thread (file number 20) Coffee beans (file number 4)

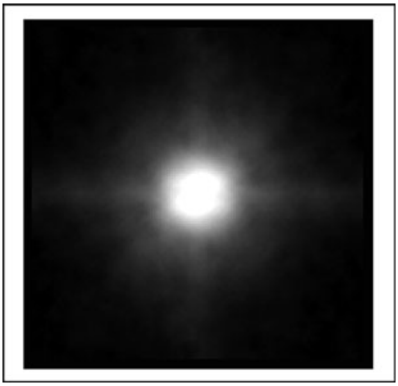
Top-left: screw thread image (file number 20)

Top-right: after *rni*(*20*), *mdf*(*7*), *ftm*

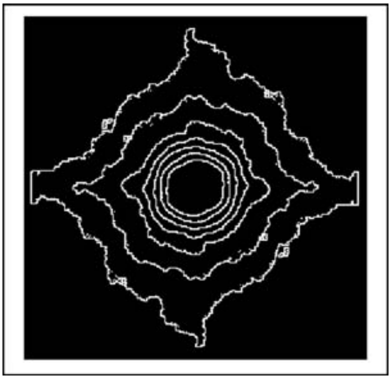


Centre-left: coffee beans image (file number 4)

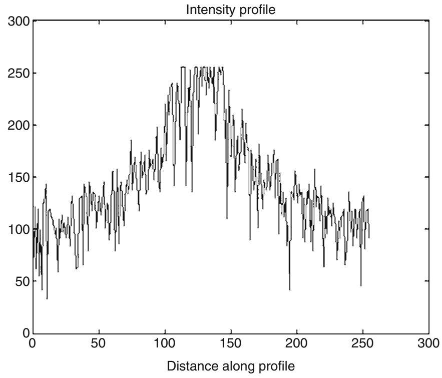
Centre-right: after *rni*(4), *mdf*(*7*), *ftm*, *enc*



Bottom-left: after *mdf*(*7*), *ftm*, *enc*, *sca*(3), *sed*, *thr*(4)

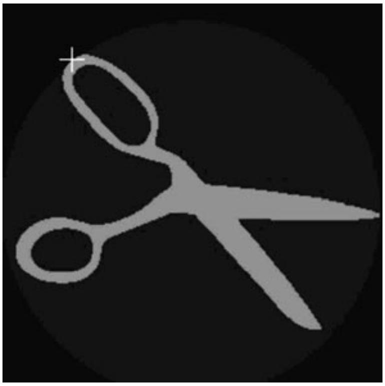


Bottom-right: after *mdf*(*7*), *ftm*, *enc*, *plt*



41.132 fwp

Format	$[u, v, w] = fwp(x, y)$	
Arguments	x	X-coordinate of point
	y	Y-coordinate of point
Defaults	$[x, y]$	Centroid of the object: [*Xcentroid*, *Ycentroid*].
Output values	u	X-coordinate of the white point furthest from *[x, y]*
	v	Y-coordinate of the white point furthest from *[x, y]*
	w	Distance between *[x, y]* and *[u, v]*
Image type	Binary, single object	
Function	Find *[u, v]*, the coordinates of the white point that is furthest from point $[x, y]$. Draw the circle of radius w centred on $[u, v]$, where w is the distance between $[x, y]$ and $[u, v]$.	
Switch images	Yes	
Typical uses	Finding the orientation of a binary object. The orientation of the line joining [*Xcentroid*, *Ycentroid*] and $[u, v]$ is an alternative to the Principal Axis as a means of finding orientation. (The Principal Axis is found by *lmi*)	
Remarks	In its present form, *fwp* will only work properly if $w \leq 255$ pixels	
Illustration		
Input image	Scissors	
File number	48	
Command	$[u, v] = Jwp, adi, hin, dcl(u, v)$	



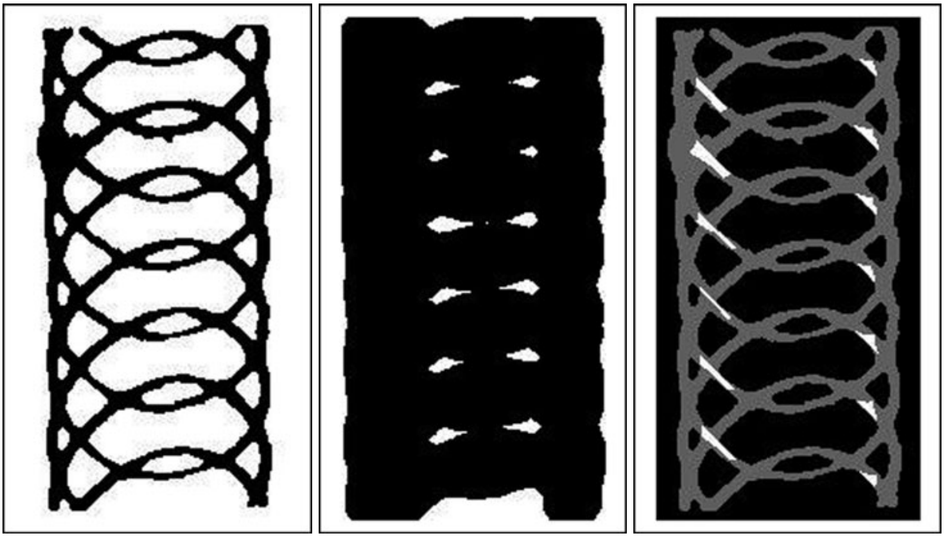
41.133 gap

Format	[*x*, *y*, *z*] = *gap*
Arguments	None
Output values	*x* Vector containing X-coordinates of non-black pixels *y* Vector containing Y-coordinates of non-black pixels *z* Vector containing intensity values of non-black pixels
Image type	Grey or binary
Type	Image measurement
Function	Get the [X, Y]-coordinates and intensity values for all non-black pixels and store them in three separate output vectors
Switch images	No
Typical uses	*gap* can be used in conjunction with *bed*, *ult*, *shk*, etc. to reduce an image to just a small amount of (non-image) data
Remarks	The sequence *[a, b, c]* = *gap*; *fap*(*a*, *b*, *c*) was tested on a 256∗256 pixel image that contains no black pixels. The image was copied successfully but the operation is very slow

41.134 gbd/gbe

Format	*gbd*(*a*) *gbe*(*a*)
Arguments	*a* Structuring element: a 2-dimensional array consisting of 0 s and 1 s
Defaults	None
Output values	None
Image type	Binary
Type	Morphology
Function	General binary dilation (*gbd*)/erosion (*gbe*)
Switch images	Yes
Typical uses	Analysing textured and patterned surfaces. Detecting specific features in an image
Remarks	The function *cse* can be used to generate several frequently used structuring elements: lines (at 0, ± 45° and 90°), upright and diagonal crosses. The MATLAB function *strel* can be used instead, to create structuring elements
Illustration	
Input image	Cake decoration pattern, with thresholding *(thr)* applied to avoid JPEG artifacts
File number	55
Command	Sequence 1: *c* = *cse*(*13*, *21*), *gbe*(*c*) Sequence 2: c = *cse(3, 23), gbe(c), adi*

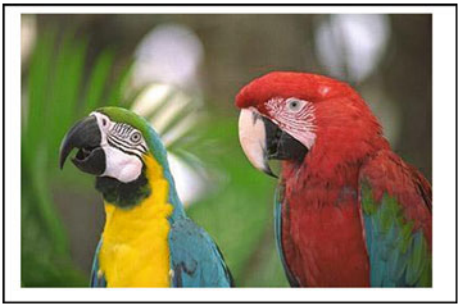
Left: original *Centre*: after sequence 1 *Right*: after sequence 2



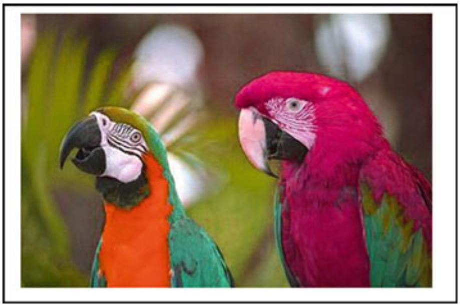
41.135 gcc

Format	*gcc*(*a*, *b*, *c*)
Arguments	*a* Additive adjustment made to the *H* channel *b* Multiplicative adjustment made to the *S* channel *c* Multiplicative adjustment made to the *V* (or I, intensity) channel
Defaults	None
Output values	None
Image type	RGB colour image
Type	Image enhancement
Function	General colour control; adjust the hue, saturation and intensity for each pixel
Definition	$hue = hue + a$, mod 1.0 $saturation = saturation * b$ (Hard limit applied at level 1) $intensity = intensity * c$ (Hard limit applied at level 255)
Typical uses	Image enhancement for visual examination of a scene with low visibility features
Remarks	Also see *gma*, *cga*
Illustration	
Input image	Birds
File number	144
Command	*test1*(−*0.08*, *1.3*, *1*)

Left: original



Right: after processing



41.136 gft

Format *gft*(*a*)

Arguments *a* Choose distance metric. ($0 \leq a \leq 3$)

Defaults *a* 0 (Euclidean distance metric)

Output values None

Image type Input image is binary. The output is a grey-scale image

Type Blob transform

Function Grass-fire transform (Also called *Prairie Fire Transform*)

Description Imagine that the blob is made of combustible material. A fire is ignited simultaneously at all edge points, including inner edges surrounding holes. The intensity in the grass-fire transform image indicates how long the fire takes to reach a given point within the blob

Switch images Yes

Typical uses Blob shape analysis

Remarks The grass-fire transform indicates the distance to the nearest edge pixel. Hence, it is useful for designing morphological operators (i.e., deciding on the size of dilation/erosion structuring elements.) Distance can be measured in a number of ways:

Input parameter, a	Distance metric
0	Euclidean
1	City Block or Manhatten
2	Chessboard
3	Quasi-Euclidean

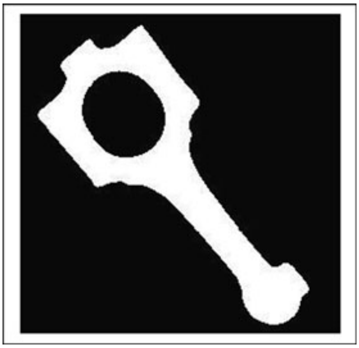
Illustration

Input image Con-rod

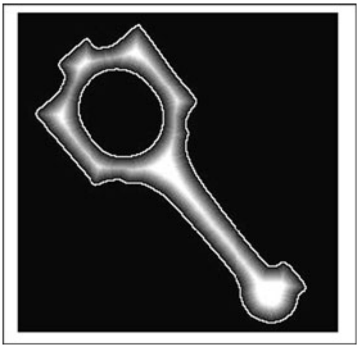
File number 47

Command wri(1), *gft*, *enc*, *wri*(*2*), *rei*(*1*), *bed*, *rei*(*2*), *mxi* (The outer edge was superimposed for clarity)

Left: original



Right: after processing



41.137 gis

Format *gis*(*n*, *d*, *u*, *l*)

Arguments
n No. of images to be stored
d Delay in seconds between successive images
u URL of a web-enabled camera
l Image label

Output values None

Image type Not relevant

Type Image acquisition

Function Capture a series of *n* images from camera *u* at intervals of *d* seconds. The images are stored with labels defined by *l* (e.g., *image1*, *image2*, *image3*, . . .)

Switch images Yes

Typical uses Capturing experimental data from a factory for off-line analysis later

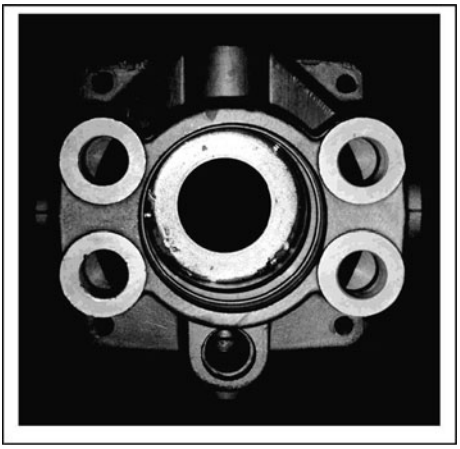
Remarks Precise timing may be a problem when network traffic is heavy and/or the camera is heavily loaded. Provided the parameter *d* is greater than the delay in updating the camera image, the best that can be accomplished is capturing one new image within each *d* second time slot

Sample image sequence captured from a camera in New York City, USA. The computer capturing the images was in Cardiff, Wales, UK. Default values were used for all parameters.

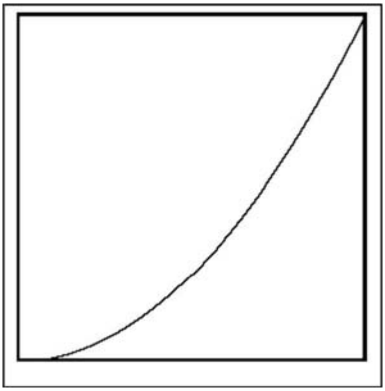
41.138 git

Format	*git*(*a*)
Arguments	*a* Mapping function, specified as a 1D array
Defaults	None
Output values	None
Image type	Grey
Type	Monadic
Function	General intensity transform, using a 256-element look-up table
Switch images	Yes
Typical uses	Image enhancement
Remarks	The input vector, *a*. should have 256 elements and be within the range [0, 255]. Also see *sqr*, *sqt*, *nlg*, *alg*, *enc*, *heq*, *blo*
Illustration	
Input image	Hydraulics manifold
File number	441
Commands	*Centre*: *rni*(*441*), *for i = 1:256*, *x1*(*i*) = (*i* − *1*)∗(*i* − *1*)/*255*; *end*, *git*(*x1*) *Bottom*: *gsn*, *gsn*, *x2* = *hqx*; *rni*(*441*), *git*(*x2*)

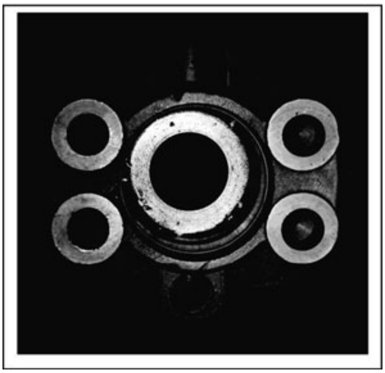
Top: original image



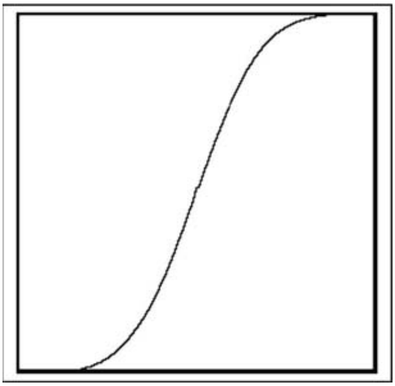
Centre-left: mapping function (*for i = 1:256, x1*(*i*) = (*i* − *1*)∗(*i* − *1*)*/255*; *end* This produces the same result as *sqr*.)



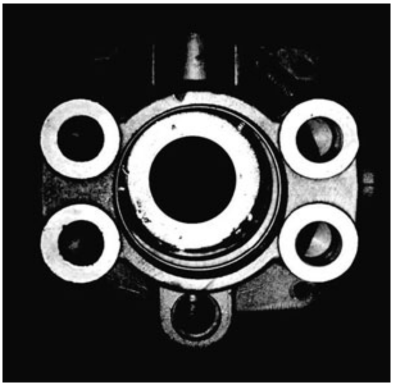
Centre-right: after *git*(*x1*)



Bottom-left: mapping function (*gsn*, *gsn*, *x2* = *hqx*)



Bottom-right: after *git*(*x2*)



41.139 glc

In some versions of QT for MS-Windows, the mnemonic *con* causes problems and is renamed *glc*. See *con*.

41.140 gli

Format	[*a*, *b*] = *gli*	
Arguments	None	
Defaults	Not applicable	
Output values	*a*	Lower intensity limit
	b	Upper intensity limit
Image type	Grey-scale	
Type	Grey-scale image measurement	
Function	Calculate the upper and lower intensity limits	
Switch images	No	
Typical uses	Finding a suitable threshold parameter	
Remarks	Also see *enc*	

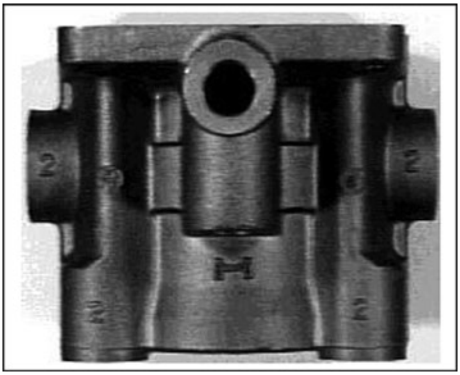
41.141 gma

Format	*gma*(*a*)
Arguments	a Gamma (γ)
Defaults	None
Output values	None
Image type	Grey
Type	Monadic
Function	Square of intensities in the current image
Definition	$C \Leftarrow 255*(A/255)^{\gamma}$, where γ is a constant, typically in the range $0.5 \leq \gamma \leq 2$
Switch images	Yes
Typical uses	Correcting for camera non-linearity
Remarks	The function *gma* is used to improve contrast in bright/dark regions of the image. If $\gamma < 1$, dark regions are enhanced, while bright regions are enhanced if $\gamma > 1$. Photographers (and video engineers) are familiar with the concept of gamma correction: a non-linear mapping of intensity is applied to an image to compensate for film (or video sensor) non-linearity. The function *sqr* is equivalent to *gma*(*2*) and *sqt* to *gma*(*0.5*). Clearly, *gma* provides finer control of contrast than either of these. Also see *alg* and *nlg*

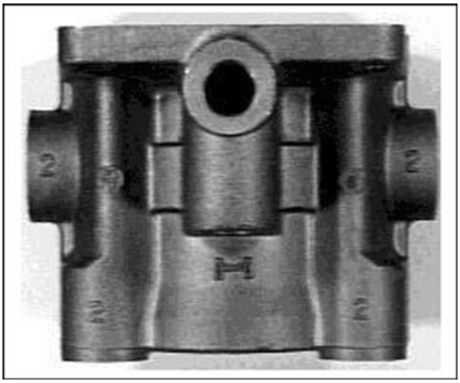
Illustration

Input image	Hydraulics component (side view)
File number	38
Command	*gma*(*0.7*)

Left: original



Right: after processing



41.142 gpt

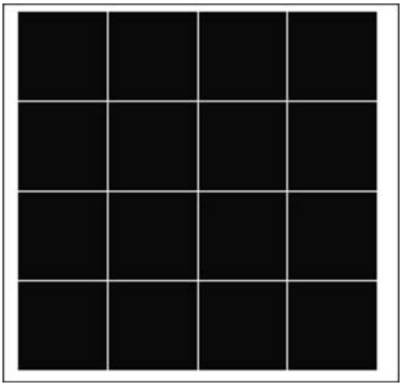
Format	*c* = *gpt*(*a*, *b*)	
Arguments	*a*	X-coordinate of a point
	b	Y-coordinate of a point
Defaults	None	
Output values	c	Intensity at [x, y]
Image type	Greyscale or binary	
Type	Grey-scale/binary image measurement	
Function	Find the intensity (*c*) at the point *[a, b]*	
Switch images	No	
Remarks	Also see *pxv*, and *blp/brp/lbp/...*	

41.143 grb

Format	*grb*(*a*, *b*)	
Arguments	*a*	*Integer, camera selection*
	b	*Grey-scale* (*0*)*/colour* (*1*) *selection*
Defaults	*a*	*0*
	b	*0* (*Convert image to grey-scale format*)
Output values	*None*	
Type	Image acquisition	
Function	Acquire an image from a local camera from an external webcam	
Switch images	Yes	
Remarks	The present version of *grb* illustrates how the M-file is structured. Some of the cameras are outside the author's control, so their availability cannot be guaranteed. From time to time, some of these cameras may be replaced and/or others added. For a good source of information about web cameras, visit *http://www.earthcam.com/* Also see *gwi*	

41.144 grd

Format	*grd*
Arguments	None
Defaults	Not applicable
Output values	None
Type	Drawing and image generation
Switch	Yes
Function	Generate a simple grid pattern
Switch images	Yes
Typical uses	Testing various image processing functions, such as filters
Remarks	Also see *grd*
Illustration	
Commands	*grd*



41.145 grip1 – grip4

The functions *grip1*, *grip2*, *grip3* and *grip4* are included here merely to illustrate the use of QT in robot vision and do not purport to provide comprehensive coverage of this subject. Our intention is to show that an image processor with a few simple graphics functions can be used to good effect in determining where best to place a robot gripper so that it can lift an object and checking whether it is safe to attempt to do so. Four situations are considered and are summarised in the table below.

QT function	Gripper fingers	Inputs	Outputs/draws
grip1	2 rectangular	XY position of gripper	No data output
		Orientation of gripper	Draws gripper as 2 rectangles
grip2	2 rectangular	XY positions of 2 points	XY position of gripper
			Orientation of gripper
		Size of gripper fingers	Draws gripper as 2 rectangles.
grip3	2 circular	Radius of gripper fingers	Positions of 2 fingers
			Draws gripper as 2 circles
grip4	1 circular (suction or magnetic)	None	Position of gripper
			Orientation of gripper
			Draws largest possible gripper

It is assumed that, in each case, one, or more, objects are lying on a plain smooth table that provides a high-contrast background. An overhead camera views the objects, which will then be picked up by a robot. To do so properly, the robot requires the following information:

1. XY coordinates for the centre axis of the robot's gripper.
2. Angular position of the robot's gripper.

No consideration is given here to converting camera coordinates to robot coordinates, nor the initial calibration of the 'hand-eye' system. The task is thus reduced to its simplest form as an exercise in 2D image intepretation.

41.145.1 grip1

Format	*grip1(x, y, z, d, N1, N2)*	
Arguments	*x*	X-coordinate of the centre axis of the gripper
	y	R-coordinate of the centre axis of the gripper
	z	Orientation of the gripper
	d	Separation of the finger of the gripper
	N1	Width of the fingerprint (one finger)
	N2	Length of the fingerprint (one finger)
Defaults	None	

Outputs None

Function *grip1 draws* two rectangles to indicate where the fingers will be placed

Switch images Yes

Typical uses Checking that the intended placement of the robot gripper will not cause a crash

Remarks This is a simple graphics operator, which draws two rectangles to indicate where the fingers will be placed. The gripper is in the *open* position. *grip1* is a component of *grip2*. An illustrative example of its output is shown later (*grip2*)

Illustration See *grip2*

41.145.2 grip2

Format *[x, y, z]=grip2(x1, y1, x2, y2, N1, N2)*

Arguments

Arguments	*[x1, y1]*	Coordinates of one point
	[x2, y2]	Coordinates of another point
	N1	Width of the fingerprint (one finger)
	N2	Length of the fingerprint (one finger)
Defaults	None	
Outputs	*[x, y, z]*	Position and orientation of the gripper

Function *grip2* accepts the coordinates of two points on any object and then calculates the following:

Coordinates of the centre of the gripper *[x, y]*

Orientation of the gripper *(z)*

It then draws the fingerprint of the gripper as two rectangles (Same as *grip1*)

Switch images Yes

Typical uses Checking that the intended placement of the robot gripper will not cause a crash

Remarks The image below shows an electrical inductor (coil) with two flying leads. Our objective is to design a visually guided robot that can locate and then grasp the end of one of the wires. In practice, this application requires 5 cameras, with computer-controlled lighting, and a robot with 6 degrees of freedom. However, we shall simplify the task by considering only the plan view of the coil. In order to take variations of back-ground intensity into account, the following preprocessing sequence may be used:

```
enc,            % Enhance contrast
mdf,            % Median filter
crk,            % ‘Crack detector’: grey-scale closing operator
thr(123)        % Threshold at some appropriate value
```

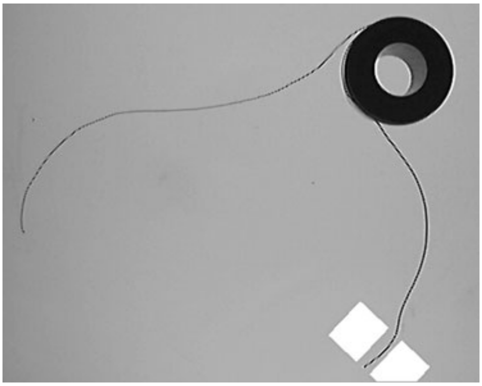
We then use the following sequence applied to the binary image generated above:

```
thn,            % Thinning operator
spr(15),        % Recursively remove the limb-end pixels (15 loops)
exr,            % Exclusive OR - isolates the ends of the wires
```

```
big,                  % Select one blob (i.e., tip of one of the wires)
lmb,                  % Select limb ends
x=gap;
a=angle([x(1), y(1)],[x(2), y(2)])
                      % Orientation of the end of wire 1
grip2((x(1)+x(2))/2, y(1)+y(2))/2, a, 10, 15, 20)
```

The result of processing a typical image of a coil is shown below. The white rectangles show the positions of the finger tips when the gripper is '*open*' (The simple version of *grip1* listed below draws only the outline of the lower rectangle, because it overlaps the edge of the image. This rectangle was 'filled in' manually, to make the illustration clearer)

Illustration The image below shows the coil with two flying leads. Superimposed on this are two white rectangles representing the fingerprints of the gripper in the 'open' state. These were calculated by *grip2*



41.145.3 grip3

Format	*[u1, v1, u2, v2]=grip3(a)*	
Arguments	*a*	Radius of gripper fingers
Defaults	a	24
Outputs	*[u1, v1]*	Coordinates of one of the gripper fingers
	[u2, v2]	Coordinates of the other gripper finger

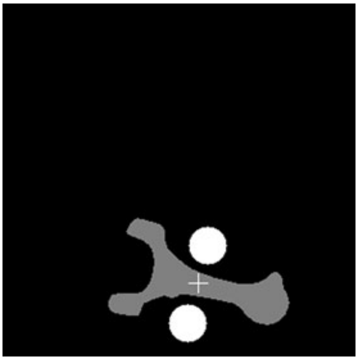
Function *grip3 draws* two solid circles to indicate where the fingers will be placed and returns 4 numeric values indicating their positions. The finger positions are calculated automatically from the centroid and the orientation of the principal axis

Switch images Yes

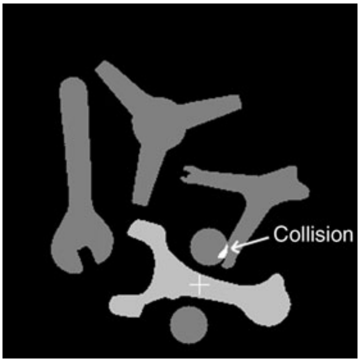
Typical uses Checking that the intended placement of the robot gripper will not cause a crash

Remarks This is a simple graphics operator, which draws two circles to indicate where the fingers will be placed. The gripper is in the *open* position

*Bottom-left***:** Image generated by *grip3*

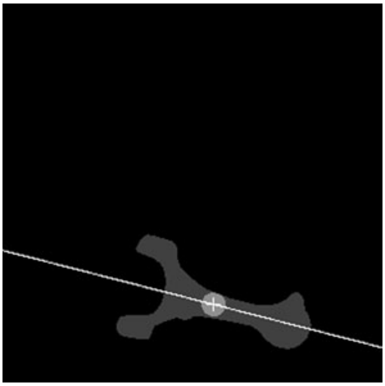


Bottom-right: grip3. Collisions ca detected



41.145.4 grip4

Format	*function [x, y, r]=grip4*
Arguments	None
Outputs	*[x, y, r]* Position and orientation of the gripper
Function	Calculate the coordinates of the centroid and orientation. Then, find the largest circular gripper that can be placed at the centroid without overlapping the edge of the object
Switch images	Yes
Typical uses	Checking that a robot with a magnetic or suction gripper will provide safe lifting
Remarks	This is a simple graphics operator that draws a white disc to show where the gripper wil be placed
Illustration	The light grey disc represents the largest suction or magnetic gripper that can be placed at the centroid (white cross). (To ensure safe lifting, these types of gripper must not overlap the edge of the object.) The orientation of the object is determined by the principal axis (white line)



41.146 grp

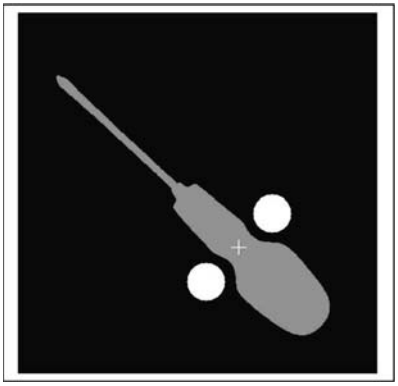
Format	$[u1, v1, u2, v2] = grp(a)$	
Arguments	*a*	Controls gripper spacing and size
Defaults	*a*	24
Output values	*[u1, v1]*	Coordinates of the first gripping point
	[u2, v2]	Coordinates of the second gripping point
Image type	Binary (single blob)	
Type	Robot vision	
Function	Calculate and display the gripping points for 2-finger robot gripper	
Switch images	Yes	
Typical uses	Demonstrates how QT can be used in conjunction with a multi-axis robot	
Remarks	The gripping points are displayed as circular discs. They are located on that line lying at 90° to the axis of minimum second moment (the principal axis) and passing through the centroid. The gripper size is adjusted automatically, to ensure that there will be some space between the object and the gripper. The simple algorithm implemented by *grp* does not always produce good gripping points. For example, on a 3-pointed star, the axis of minimum second moment is not well defined. As a result, the positions of the gripping points are highly variable as they are strongly dependant on edge quantisation effect	

Illustration

Input image	Screwdriver
File number	43
Commands	*thr, neg,*$[x1, y1, x2, y2] = grp(32)$

Left: original

Right: after processing



41.147 grs

Format [*x*, *y*, *u*] = *grs*

Arguments None

Output values *[x, y]* Coordinates of the gripping point (centroid)

u Maximum radius of the gripper to ensure good grip

Image type Binary (single blob)

Type Robot vision

Function Calculate and display the gripping point for a suction or magnetic gripper

Switch images Yes

Typical uses Demonstrates how QT can be used in conjunction with a multi-axis robot

Remarks The position of the gripper is indicated by a small white cross (i.e., the centroid of the blob) and its orientation by a white line (i.e., the axis of minimum second moment). The 'footprint' of the largest gripper that can safely be used to pick up this object is displayed as a circular disc. The simple algorithm implemented by *grs* does not always produce a good gripping point. For example, the axis of minimum second moment of a 3-pointed star is not well defined, so the orientation of the gripper is highly variable

Illustration

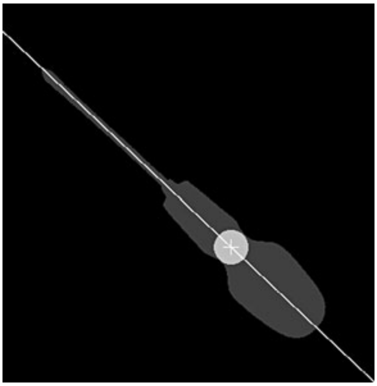
Input image Screwdriver

File number 43

Commands *thr*, *neg*,[*x*, *y*, *u*] = *grs*

Left: original

Right: after processing



41.148 gry

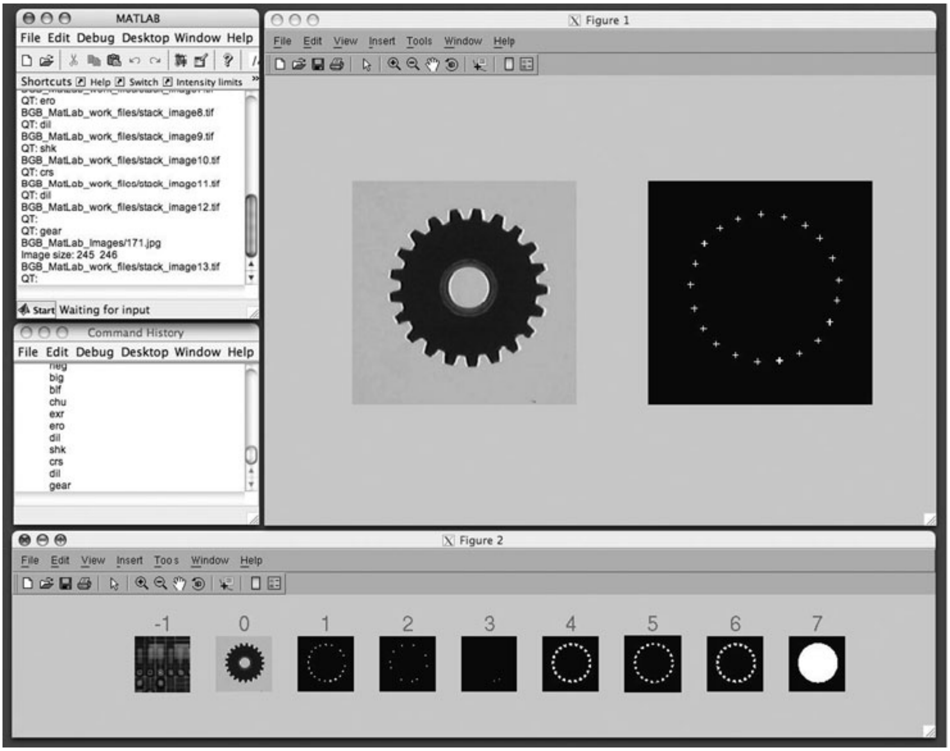
Format	*gry*
Arguments	None
Output values	None
Image type	Colour
Type	Colour transformation
Function	Convert a colour image to grey
Switch images	Yes
Typical uses	Discarding colour information to simplify processing
Remarks	Also see *hsi* and *rgb*

41.149 gsi

Format	*gsi*
Arguments	None
Output values	*None*
Image type	None
Type	System control
Function	Set up Stack Mode so that the user can recover a stack image simply by clicking on it
Switch images	No
Remarks	*gsi* alters the internal configuration but does not have any visible effect

Illustration

Screen layout, Stack Mode (i.e., Image Stack is active and part of it is displayed here). *Top-left*: Command Window; *Centre-left*: command History; *Top-right*: window labelled '*Figure 1*', contains the Current Image (left) and Alternate Image (right); *bottom*: window labelled '*Figure 2*', shows the top 8 elements of the stack and that image which was previously at the top of the stack (labelled '−1'). The user can recover any of the images displayed in *Figure 2*, simply by clicking on it.



```
QT: ero
BGB_MatLab_work_files/stack_image8.tif
QT: dil
BGB_MatLab_work_files/stack_image9.tif
QT: shk
BGB_MatLab_work_files/stack_image10.tif
QT: crs
BGB_MatLab_work_files/stack_image11.tif
QT: dil
BGB_MatLab_work_files/stack_image12.tif
QT:
QT: gear
BGB_MatLab_Images/171.jpg
Image size: 245  246
BGB_MatLab_work_files/stack_image13.tif
QT:
```

Waiting for input

```
neg
big
blf
chu
exr
ero
dil
shk
crs
dil
gear
```

MATLAB
File Edit Debug Desktop Window Help
Shortcuts
Help
Switch
Intensity limits
Start
Command History
File Edit Debug Desktop Window Help
Figure 1
File Edit View Insert Tools Window Help
Figure 2
File Edit View Insert Tools Window Help
-1
0
1
2
3
4
5
6
7

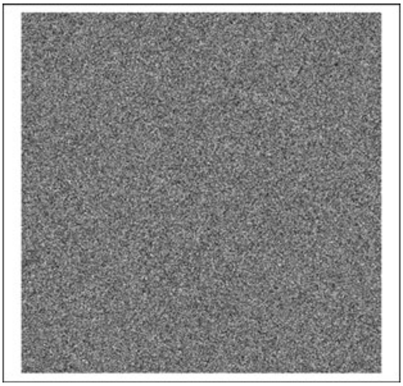
41.150 gsn

Format	*gsn*(*a*)
Arguments	*a* Determines the level of noise added to the image (0 ≤ a ≤ 1)
Defaults	*a* 1
Output values	None
Type	Drawing and image generation
Switch	Yes
Function	Generate an image array containing random numbers with a Gaussian distribution and add this to the current image. The standard deviation of the noise is determined by the input parameter, *a*
Switch images	Yes
Typical uses	Testing prototype image processing functions. Generating other test patterns
Remarks	Use *gsn* and a low-pass (blurring) filter to create a cloud-like pattern. Use a low-pass (blurring) filter and thresholding to create useful a binary test pattern

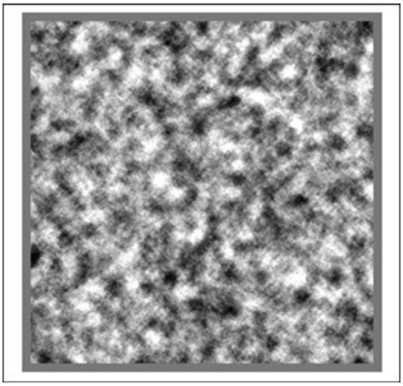
Illustration

Commands Left image: *gsn*. Right image: *gsn*, *caf*, *edg*(*8*, *128*), *enc*

Left: Gaussian noise image



Right: noise image after low-pass filtering and contrast enhancement



41.151 gwi

Format	$[b, c] = gwi(a)$
Arguments	*a* URL of the image file to be loaded (Include *http*:// at the beginning)
Defaults	Not applicable
Output values	*b* Image width (number of columns)
	c Image height (number of rows)
Image type	Image files in BMP, JPEG, PNG or TIFF formats can be read. Colour images are converted to grey-scale, using the intensity component
Type	Input/output
Function	Grab a web image given its URL
Switch images	Yes
Typical uses	Loading images from a web-cam
Sample call	*gwi*(‘*http://gemini.cs.cf.ac.uk/eyeimage.jpg*’)
Remarks	Frequently used web-cams can be accessed more conveniently by re-programming *grb*

41.152 hbt/hbt1/hbt2

Format *hbt*(*n*. *p*)
hbt1(*n*, *p*)
hbt2(*n*, *p*)

Arguments *n* Determines the size of the sampling 'snake' (*length* = *2n* + *1*)
p New image size(after normalising to produce a square image)

Defaults 8

Output values None

Image type Grey scale

Type Image filter

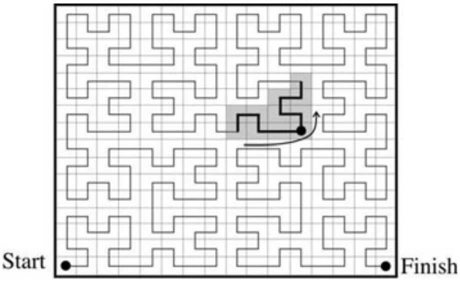
Function *hbt*

1. Reformat the input image so that it is square of size p × p pixels, *p* integer
2. Derive the path followed by the mth order Hilbert curve where $m = log_2(p)$
3. Using a 'snake' of length 2n + 1 pixels, calculate the local average intensity
4. Progressively move the 'snake' one position along the Hilbert curve and repeat step 3, for all pixels along the Hilbert curve (See diagram below)

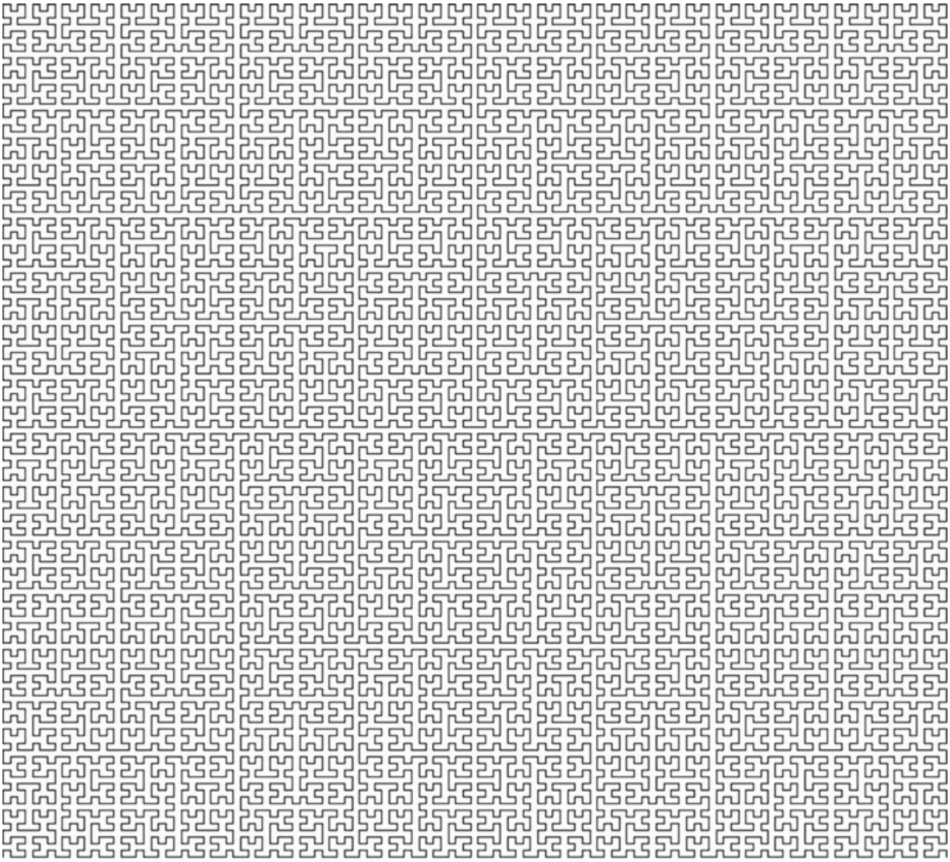
hbt1 As for hbt, except that the maximum intensity is found along the 'snake'.

Switch images Yes but the picture displayed in the Alternate Image is reformatted to be square and of size $2^m \times 2^m$, where *m* is an integer and is equal to $log_2(p)$

Remarks Filtering with the Hilbert curve is relatively crude but it does lend itself to implementation in fast hardware using shift registers. Also see *scr*



Smoothing by averaging along the Hilbert curve. In this simple example the intensities in 11 pixels (shaded) are averaged. That value is then stored, in the output image, at the position indicated by the black spot. The process is repeated by moving the sampling 'snake' progressively along the Hilbert curve. Notice that the region contributing to the estimate of local average intensity changes shape as the 'snake' moves around the Hilbert curve.

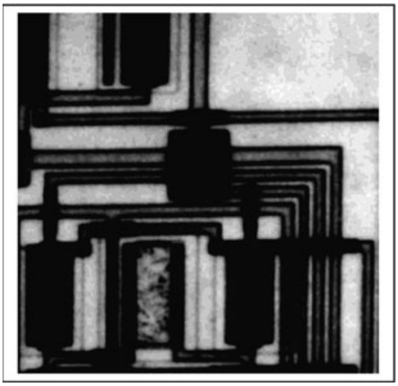


Hilbert curve, 10th order

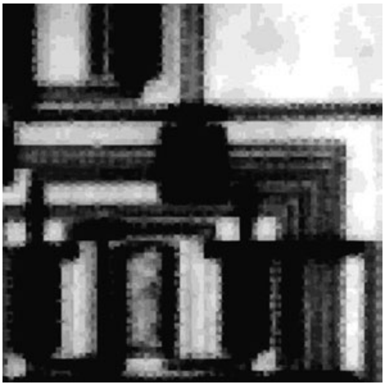
Illustration

Input image	Thick film circuit
File number	136

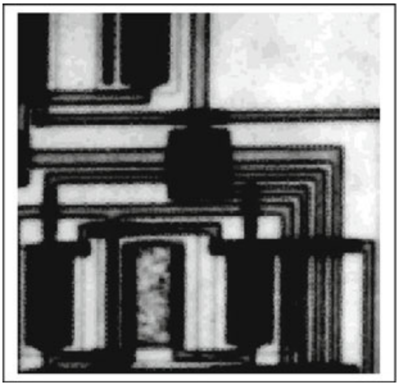
Top-left: original image



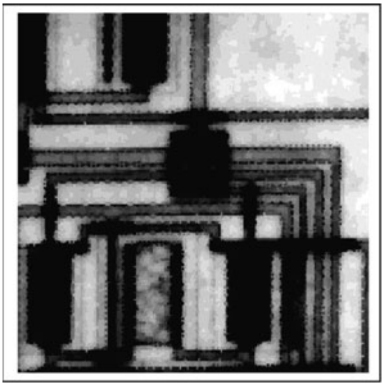
Top-right: after *hbt*(*11*, *256*)



Bottom-left: after *hbt*(*11*, *512*)



Bottom-right: after *hbt1*(*11*, *512*)



41.153 hcd

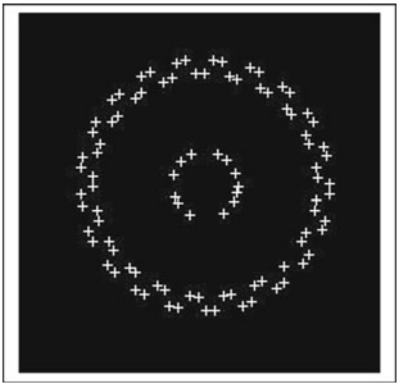
Format	*[d, e] = hcd(a, b, c)*	
Arguments	*a*	Standard deviation of smoothing Gaussian filter
	b	Threshold
	c	Radius of region considered in non-maximal suppression
Defaults	*a*	2
	b	1000
	c	2
Output values	*d*	Row coordinates of corner points
	e	Column coordinates of corner points
Image type	Grey or binary	
Type	Image analysis	
Function	Detect corners	
Switch images	Yes	
Typical uses	Feature detection, prior to measuring distances and angles based on corners. Corners may also be useful in constructing a 'line drawing' from the image	
Remarks	It may be necessary to experiment to obtain the best set of parameter values This operator tends to be 'generous' in detecting subtle features, which can reduce the speed of operation	

Illustration

Input image	Gear
File number	171
Command	*hcd*, *dil*(*3*), *crs*(*3*)

Top-left: original image

Right: after processing

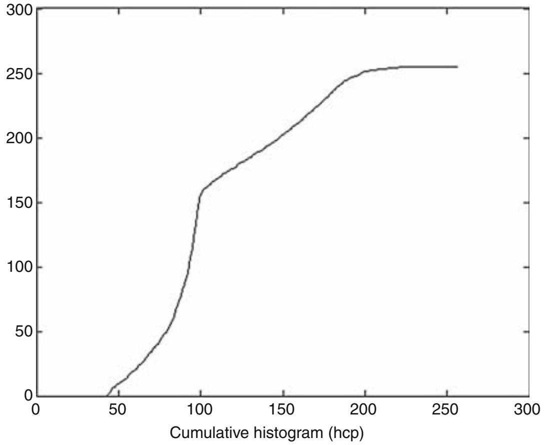


41.154 hcp

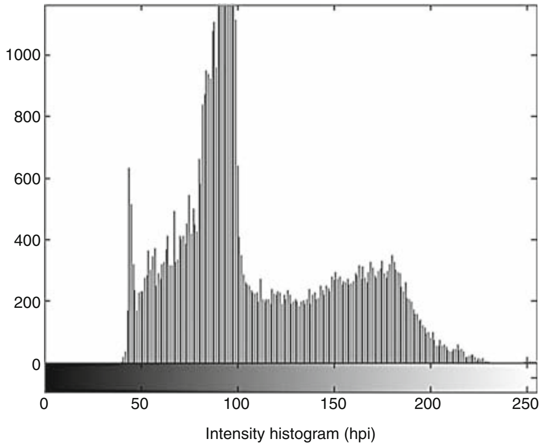
Format	*a* = *hcp*
Arguments	None
Defaults	Not applicable
Output values	*a* Cumulative histogram, rescaled to the range [0, 255]
Image type	Grey
Type	Histogram analysis
Function	Calculate and plot the cumulative intensity histogram. Strictly speaking, this is the function *H*(*X*), where $H(X) = \int_{x=0}^{x=X} h(x).d(x)$ However, this is approximated, in the discrete-variable arithmetic used in QT, by $H(X) = \sum_{i=0}^{X} h(i)$ The output (*a*) generated by *hcp* is the 1-dimensional array: $a(X) = \frac{W.\sum_{i=0}^{(X)} h(i)}{W.H}$ a(X) is the function that performs histogram equalisation on the input image
Switch images	Yes
Typical uses	*hcp* provides useful way to generate the histogram equalisation mapping function. This can be used in a variety of ways, such as contrast enhancement by hyperbolisation. That is, the histogram of the resulting picture has a histogram that has the form *1/X*
Remarks	Also see *hpi*, *hqx*, *mit*

Illustration

Left: histogram equalisation mapping function



Right: intensity histogram



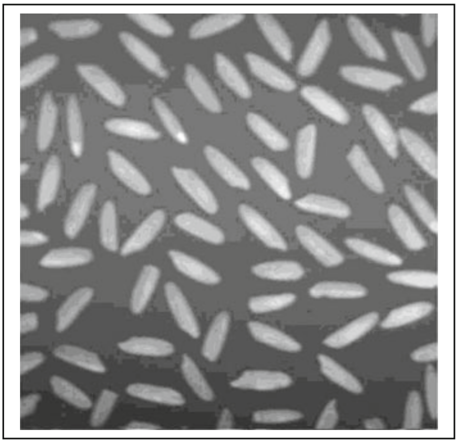
41.155 heq

Format	*heq(a)*
Arguments	*a* Control parameter for plotting the mapping function
Defaults	a 0 Do not plot the mapping function
Output values	None
Image type	Grey
Type	Monadic
Function	Enhance contrast by applying histogram equalisation to the intensity scale. This results in an image whose intensity histogram is nearly flat.
Switch images	Yes
Typical uses	Increasing contrast.
Remarks	The darkest pixel in the output image is black and the brightest pixel is white. However, notice that *enc* performs a linear stretching of the intensity scale, whereas *heq* is non-linear. Also see *blo*, *acn*, *din*, *hcp, hin.*

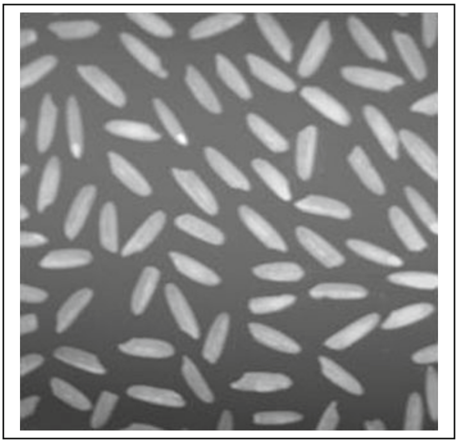
Illustration

Input image	Rice grains
File number	3
Command	*heq*

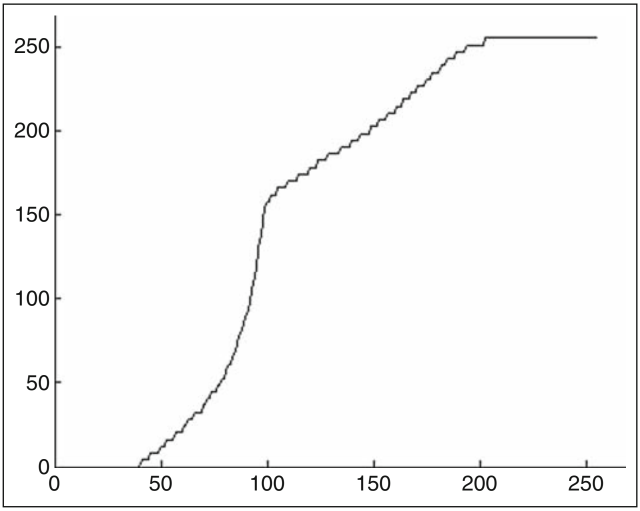
Left: original



Right: after processing



Bottom: Mapping function. Abscissa, input intensity. Ordinate, Output value



41.156 hgm

Format	$b = hgm(a)$	
Arguments	a	Select type of histogram display
Defaults	a	1
Output values	b	Intensity value selected by the user
Image type	Grey	
Type	Histogram	

Function: Plot the intensity histogram of the current image. There are four options, as indicated in the following table

Input	Plotted in	Colour bar (wedge)	Ordinate axis	Interactive	Switch Images
None	Figure 1	No	Linear	No	No
1	Figure 1	Yes	Linear	No	No
2	Figure 1	Yes	Linear	Yes	Yes
3	Figure 1	Yes	Logarithmic	Yes	Yes
4	Figure 3	Yes	Linear	No	No
5	Figure 1	Yes	Cumulative	No	Yes

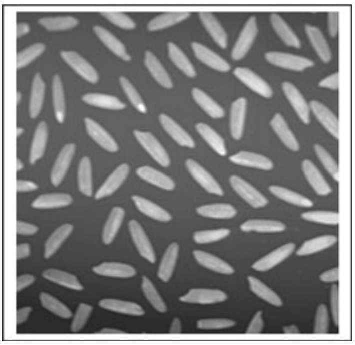
Remarks: *hgm* is used principally as an aid to understanding the structure of an image and particularly in helping the user to select an appropriate threshold parameter. (Thresholding is performed separately, probably using *thr*.) The image of the histogram created by *hgm*(*1*) can be processed like any other grey-scale image. *hgm*(*2*) allows the user to relate peaks and valleys to the intensity scale and then estimate where to place a threshold *hgm*(*3*) is similar but uses a logarithmic vertical scale, to compress very high peaks, so that lower peaks can be examined more closely. *hgm*(*4*) produces a pretty annotated diagram that is suitable for insertion into a report, thesis, book, etc. Also see *dth*, *cur*

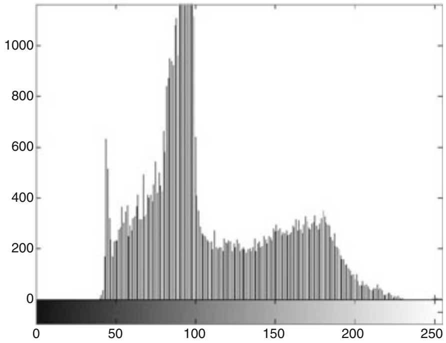
Illustration

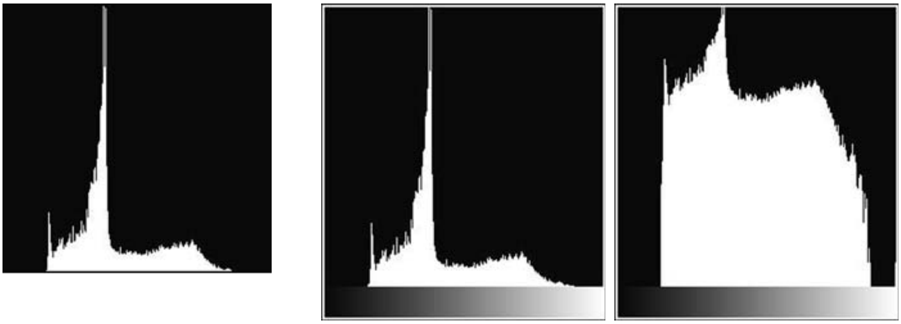
Input image: Rice grains

File number: 3

Command:
Top-left: Original image
Top-right: *hgm*(*4*)
Bottom-left: *hgm*(*1*)
Bottom-centre: *hgm*(*2*)
Bottom-right: *hgm*(*3*)







41.157 hgr

See *con.*

41.158 hil

Format *hil*(*a*, *b*, *c*)

Arguments *a* Lower intensity threshold limit

b Upper intensity threshold limit

c New intensity value

Defaults *c* 255

Output values None

Image type Grey

Type Monadic

Function Highlight a given range of intensity values; all pixels whose intensities lie within the range *[a, b]* are mapped to level c. All other pixels remain unchanged

Switch images Yes

Typical uses Image segmentation (i.e., isolating an object silhouette by suppressing unwanted intensity variations in its background)

Remarks To novice users, thresholding (*thr*) and highlighting are obvious functions to use in many applications. However, both are sensitive to lighting variations and will not produce reliable image segmentation in any but the simplest applications. For this reason, they are almost always used where manual setting of the input parameters is possible, or in conjunction with a suitable filter, edge detector, or intensity-normalisation function

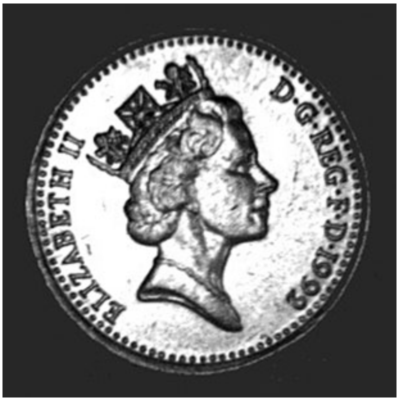
Illustration

Input image Coin

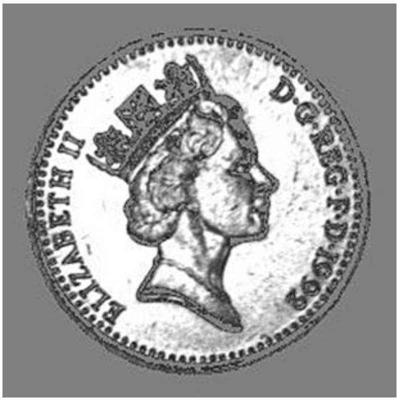
File number 23

Command *hil*(*0*, *50*, *128*)

Left: original



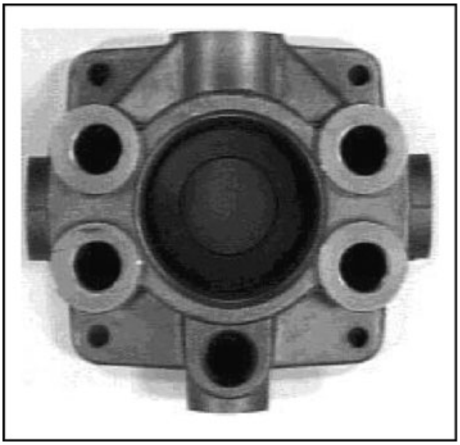
Right: after processing



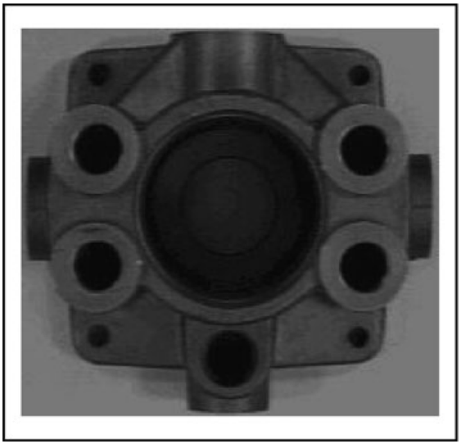
41.159 hin

Format	*hin*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Monadic
Function	Halve all intensities
Definition	C ⇐ A/2
Switch images	Yes
Typical uses	Reducing intensity so that white annotation features can be superimposed (using *mxi*)
Remarks	Also see air, *blo*, *din*, *enc* and *heq*
Illustration	
Input image	Hydraulics component
File number	37
Command	*hin*

Left: original



Right: after processing



41.160 hlp

Format	*hlp*('*topic*') or *hlp topic*
Arguments	topic Character sequence
Defaults	No topic supplied: *hlp* lists all of the M-files in the scripts folder
Output values	None. Data is printed on the screen, in MATLAB's *Command Window*
Type	Miscellaneous
Function	Help facility. QT and MATLAB's own M-files are listed. See example below
Switch images	No
Remarks	The reader is also referred to MATLAB's *help*, *which* and *what* functions
Output:	no argument supplied

```
aab.m     Area of all blobs in a binary image. [1/0]
abd.m     Absolute value of difference in intensities between current and lternate
abv.m     Absolute value of the intensity in the current image. [0/0]
.........
zed.m     Zero-cross edge detector. [1/1]
zer.m     Set all pixels to black. [0/0]
```

The brief explanation listed opposite each command is derived from the first comment line in the MATLAB listing. (The MATLAB documentation refers to this as the *H1* line.)

Typical output: command hlp('radon') or *hlp radon*

```
dlr.m:    % Draw one line identified from a peak in the Radon transform image [0/4]
drp.m:    % Dominant Radon projection [0/0]
irt.m:    % Inverse Radon Transform [0/0]
rdn.m:    % Radon transform [0/0]
IRADON Compute inverse Radon transform     N.B. This is a MATLAB function
RADON Compute Radon transform              N.B. This is a MATLAB function
RADONC Helper function for RADON           N.B. This is a M function
```

41.161 hpf

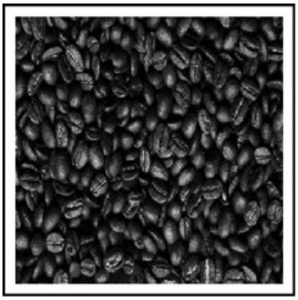
See *lpf*.

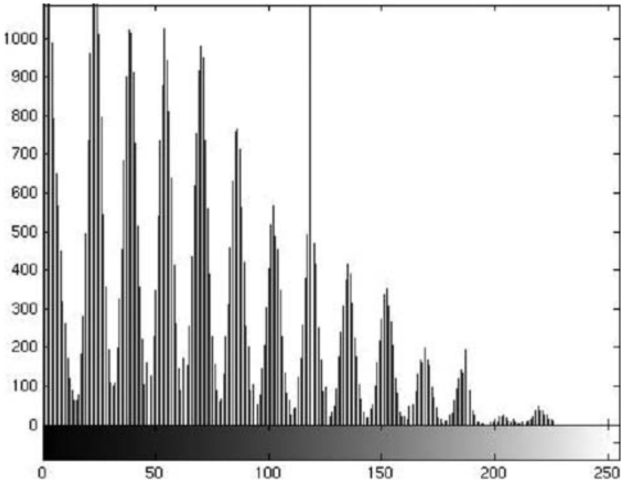
41.162 hpi

Format	*hpi*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Histogram
Function	Calculate and plot the intensity histogram of the current image in Figure 4 (full screen). The user closes Figure 4 by clicking anywhere on it
Switch images	No
Typical uses	*hpi* is used principally as an aid to understanding the structure of image and in guiding the design of an algorithm. It is commonly used is to assist in the choice of a threshold parameter. For example, a bimodal intensity histogram often results from back-lighting an opaque object. A threshold point close to the bottom of the valley between the two peaks in the histogram is often able to isolate the object silhouette
Illustration	
Input image	Coffee beans
File number	4
Command	*hpi*
Remark:	The multiple peaks in the histogram below arise as a result of saving the image in JPEG format (The image was originally stored in JPEG format and later converted to TIF format for this book.)

Left: input image

Right: intensity histogram Abscissa, intensity. Ordinate, frequency (no. of pixels)





41.163 hqx

Format *a* = *hqx*

Arguments None

Defaults Not applicable

Output values *a* Histogram equalisation mapping function, derived from the Alternate image

Image type Current and Alternate images should both be grey

Type Histogram adjustment

Function The mapping function that would perform histogram equalisation on the Alternate image is computed. (This function is returned in the 1-D output array *a*.) This mapping function is then applied to the Current image

Switch images Yes

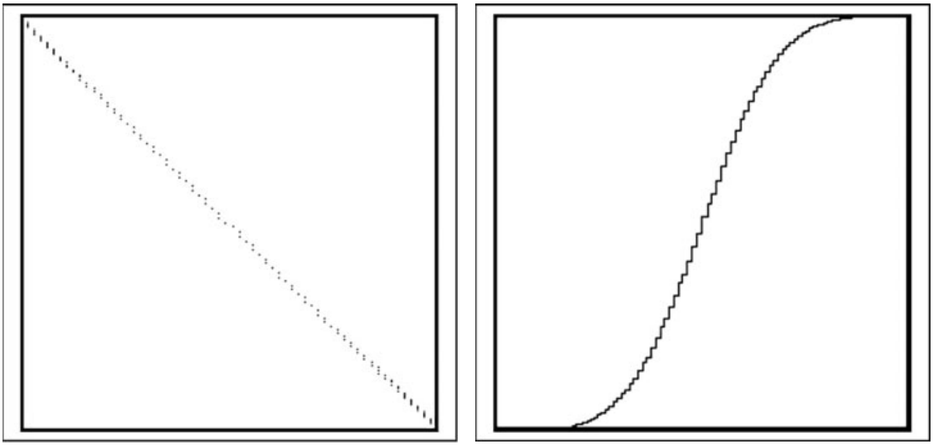
Typical uses *hqx* is useful as a way of modelling near real-time histogram equalisation, where the mapping function derived from frame *I* of a video sequence is applied to frame (*I* + *1*) or (*I* + *2*), not to frame *I*

Remarks The following command sequence should produce a straight line *gsn*, *heq*, *wri*, *swi*, *cpy*, *hqx*, *rei*, *imf*

In fact, [*cpy*, *hqx*] generates an image that differs slightly from that produced directly by *heq*; the differences are slight and appear to be due to rounding errors. (See illustration below.) Also see *hcp* and *hpi*

Illustration

Left: *gsn*, *heq*, *wri*, *swi*, *cpy*, *hqx*, *rei*, *imf* *Right*: mapping function, ‘Gaussian noise’ image



41.164 hsb

Format	*[c, d] = hsb(a, b)*	
Arguments	*a*	Number indicating which row is to be scanned
	b	Minimum intensity value to be detected
Defaults	None	
Output values	*c*	X-coordinate of point found
	d	Intensity of point found
Image type	Grey scale or binary	
Type	Image measurement	
Switch	No	
Function	Scan horizontally along a given row until a bright pixel is found	
Definition	Scan row *a* from left to right. When a pixel with intensity value greater than or equal to *b* is found, stop and return the X-coordinate (*c*) and intensity (*d*)	
Switch images	No	
Typical uses	Fitting curves to edges or other contours	
Remarks	For an example of use (fitting a circle to three edge points), see *fce*	

41.165 hsi/hue/ity/sat

Format *hsi*(*a*)
hue
ity
sat

Arguments *a* Select one of the *H*, *S*, *I* channels (*hsi* only)
hue None
ity None
sat None

Defaults *a* 0 (Display *H*, *S* and *I* components separately. Do not alter the current image)

Output values None

Image type Colour

Type Image display and channel separation

Function *hsi*
a = 0: Separate and display the *H*, *S* and *I* channels
a = 1: Set the current image to be equal to the *hue* channel
a = 2: Set the current image to be equal to the *saturation* channel
a = 3: Set the current image to be equal to the *intensity* channel
hue Same as *hsi*(*1*) (select hue channel)
ity Same as *hsi*(*3*) (select intensity channel)
sat Same as *hsi*(*2*) (select saturation channel)

Switch images *hsi*
a = 0, No
a ≠ 0, Yes
hue, *ity*, *sat*
Yes

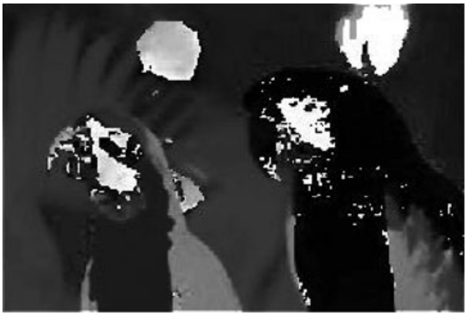
Typical uses Simplifying colour image processing, by reducing it to grey-scale processing

Remarks *hsi*(*0*) displays the 4 images in Figure 4 which is automatically scaled to fill the screen. Figure 4 closes when the user clicks on it. Also see *rgb*

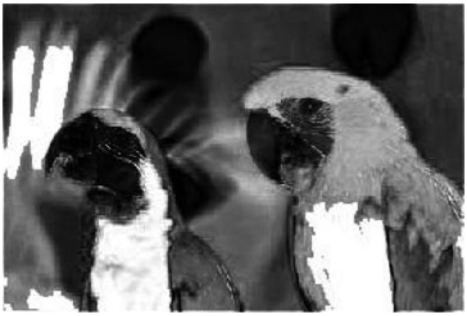
Illustration

Input image	Birds
File number	144
Command	*hsi*

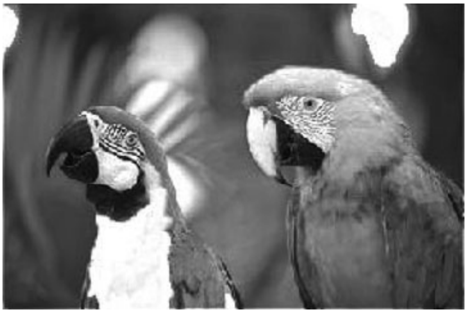
Top-left: hue



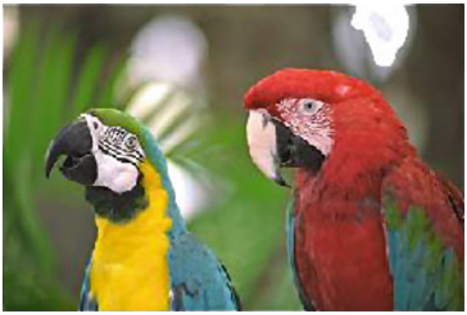
Top-right: saturation



Bottom-left: intensity

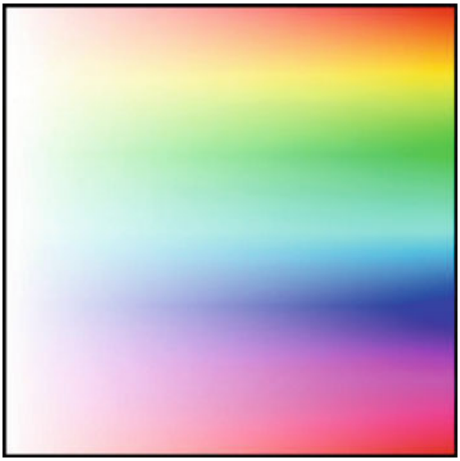


Bottom-right: original colour image



41.166 hsp

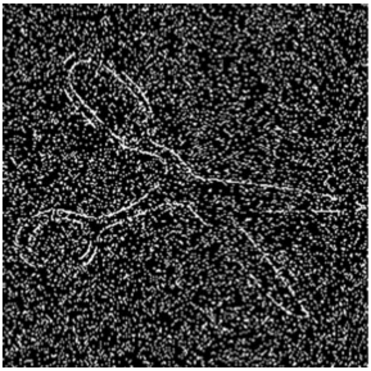
Format	*hsp*
Arguments	*None*
Output values	None
Image type	Colour image generator
Type	Colour image generator
Function	Generate a synthetic image in which the hue and saturation vary linearly with vertical and horizontal position, respectively.
Switch images	Yes
Typical uses	As an aid to interpreting and processing the output from *shs*
Remarks	Also see rgb
Illustration	
Command	*hsp*
Axes	Saturation varies along the horizontal axis. Low saturation at the left. Hue varies along the vertical axis. Notice the cyclic nature of hue



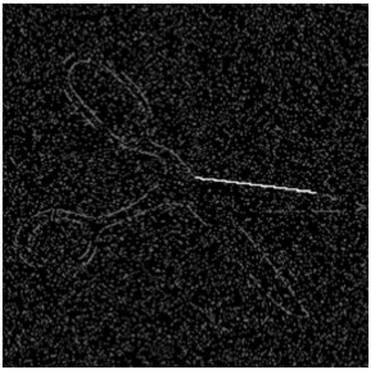
41.167 huf/fsl

Format	*huf*(*a*, *b*, *c*) [*x1*, *y1*, *x2*, *y2*] = *fsl*(*a*, *b*, *c*)		
Arguments	*huf*	*a*	Pseudo-colour display ON/OFF
		b	RhoResolution. See MATLAB documentation, function *hough*
		c	ThetaResolution. See MATLAB documentation, function *hough*
	fsl	*a*	Peak selection. See MATLAB documentation, function *houghpeaks*
		b	*FillGap* in the documentation for MATLAB function *houghlines*
		c	*MinLength* in the documentation for MATLAB function *houghlines*
Defaults,	*huf*	*a*	0
		b	2
		c	0.5
Output values	*huf*	None	
	fsl	[*x1*, *y1*, *x2*, *y2*]	End-point coordinates of the line found from the HT
Image type	Binary		
Type	Feature detection and location		
Function	*huf*	Hough Transform (HT)	
	fsl	Uses HT to extract line segments from a binary image	
Switch images	Yes		
Typical uses	Finding linear segments in noisy images and those where contours demarking large intensity gradients (i.e., found using an edge detector) are ‘broken’		
Remarks	In the image generated by *huf*, the horizontal axis represents angular position. The vertical axis determines distance from the centre of the image. *fsl* was adapted from a program given in the documentation for the MATLAB function. This is a complex function to operate most efficiently. The reader is therefore referred to the following MATLAB function descriptions: *hough*, *houghpeaks*, *houghlines*. The Hough and Radon Transforms (*rdn*) perform similar functions		
Illustration			
Input image	Scissors, deliberately corrupted by noise, generated using *gsn* and *spn*		
File number	48		

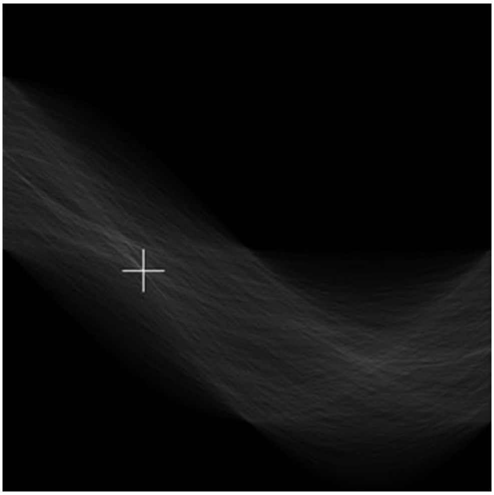
Top-left: original



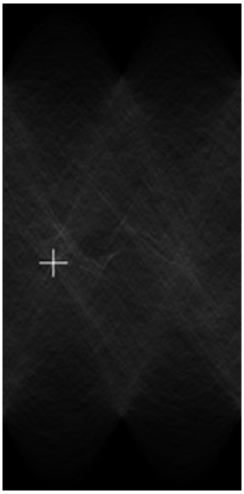
Top-right: after *fsl*



Centre-left: hough transform(huf) (Crosses indicate points of maximum intensity)

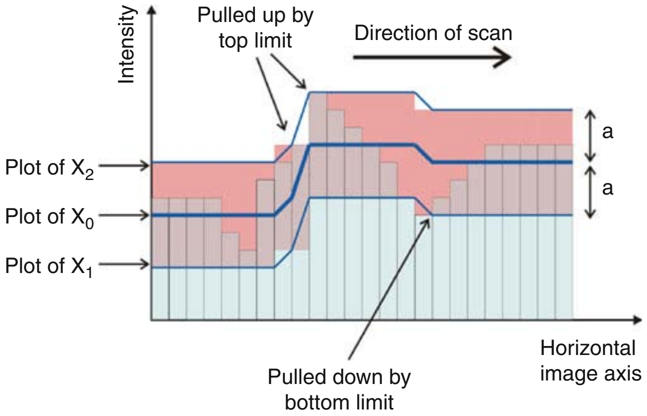


Bottom-right: radon transform (*rdn*)



41.168 hys/hs4

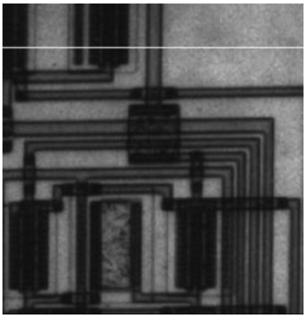
Format	*hys*(*a*) *hs4*(*a*)
Arguments	a Filter control parameter
Defaults	8
Output values	None
Image type	Grey scale
Type	Image filter
Function	*hys*: 1-dimensional hysteresis smoothing (See diagram below) *hs4*: apply *hys* 4 times, scanning in the 4 principal directions. Add the results
Switch images	Yes
Typical uses	*hys* is suitable for processing non-isotropic images generated by laser scanners and line-scan sensors inspecting web product. It may also be suitable where there is strong directional lighting
Remarks	Hysteresis smoothing is a rather crude 1-dimensional method for low-pass filtering. However, this may well suit applications where there is a marked difference between the two image axes, as explained above. It can be implemented in low-cost hardware capable of real-time operation



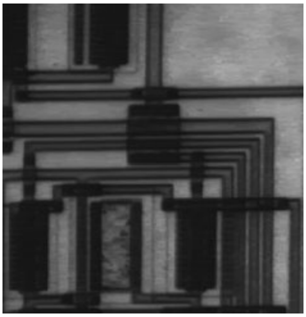
Illustration

Input image	Thick film circuit
File number	136

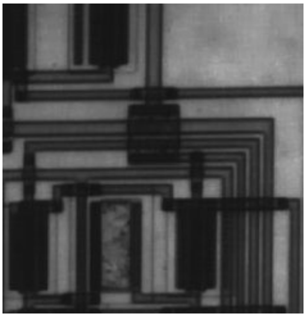
Top-left: *Original image*



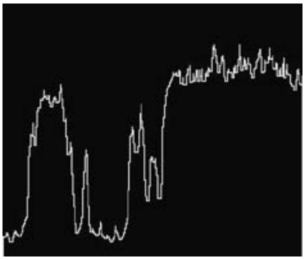
Top-center: *hys*(*8*)



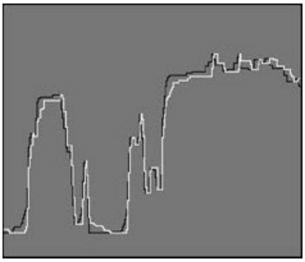
Top-right: *hs4*(*8*)



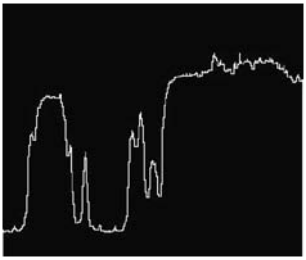
Bottom-left: *intensity profile, original*



Bottom-centre: *intensity profile, hys*(*8*)



Bottom-right: *intensity profile, hs4*(*8*)



41.169 hzs

Format $[x, y] = hzs(a, b, n)$

Arguments *a* Row number indicating start of scan

b Row number indicating end of scan

n Number of scan lines

Defaults None

Output values *x* X-coordinates of points found

y Y-coordinates of points found

Image type Binary

Type Image measurement

Switch No

Function Scan horizontally along a given set of rows. In each case, find the left-most point where the intensity is white (non-black for grey-scale images)

Definition *n* horizontal scan lines are calculated between rows *a* and *b*. Then, for each scan line, find the coordinates of the left-most white pixel

Switch images No

Typical uses Fitting curves to edges, or other contours

Remarks See *pft* and *hsb*

Illustration

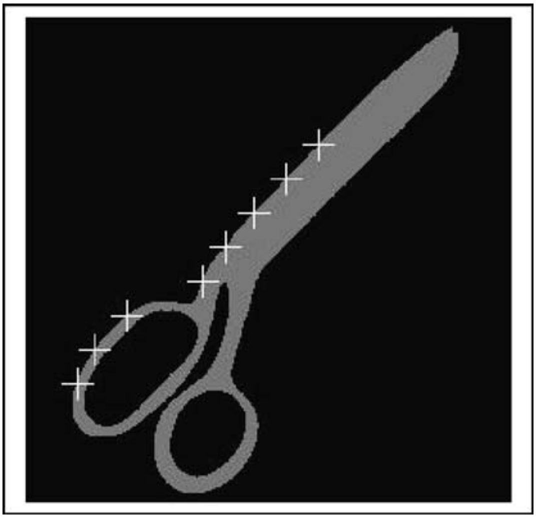
Input image Scissors

File number 50

Command *thr*, [*a*, *b*] = *hzs*(*50*, *200*, *8*)

(*thr* ensures that the image is binary and thereby avoids JPEG coding artifacts)

Left: points found by *hzs*



41.170 icc

Format	a = *icc*	
Arguments	None	
Defaults	Not applicable	
Output values	a	Correlation score. ($-1 \leq a \leq +1$)
Image type	Grey	
Type	Image analysis	
Switch	No	
Function	Cross-correlation between the current and alternate images	
Typical uses	Comparing images. Detecting 2-dimensional periodicity. Analysing texture	

41.171 idn

Format	*idn*
Arguments	None
Output values	None
Image type	Any
Type	Image display
Function	Display the current and alternate images immediately in Figure 1
Switch images	No
Typical uses	Useful during execution of long programs that do not otherwise display intermediate results

41.172 ihm

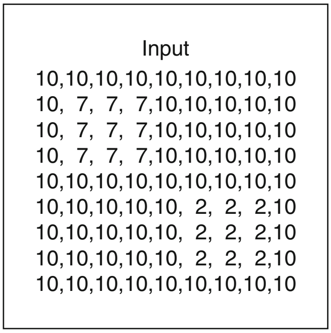
Format	$[b, c] = ihm(a)$	
Arguments	*a*	Control parameter ($a \in \{0, 1\}$. If $a \neq 0$, do not consider black level)
Defaults	a	0
Output values	*b*	Intensity value at the peak frequency in the histogram
	c	Maximum frequency value in the histogram
Image type	Grey-scale	
Type	Grey-scale image measurement	
Function	Find the position and value of the maximum in the intensity histogram. The black level can be ignored by setting $a \neq 0$	
Switch images	No	
Typical uses	Analysing the intensity histogram as part of a function for image segmentation	
Remarks	Also see *cgp* and *hpi*	

41.173 imn

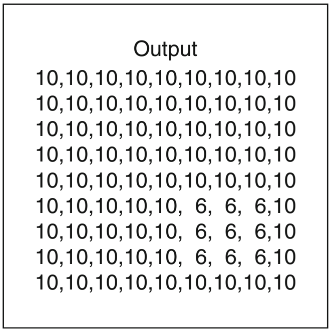
See imx

41.174 imx/imn

Format *imx*(*a*)
imn(*a*)
Arguments *a* *Intensity difference limit* ($a > 0$)
Defaults *a* *128*
Output values None
Image type Grey
Type Morphology
Function Apply the H-maxima/H-minima transform
Switch images Yes
Typical uses Noise reduction, prior to image analysis
Remarks *imx* suppresses all regional maxima whose height, compared its local surroundings, is less than *a*. (A regional maximum is an 8-connected sets of pixels that all have the same intensity value and whose external boundary pixels all have a value less than that value.) Similarly, *imn* suppresses all regional minima. The following example shows the results of applying *imn(4)* to a simple image



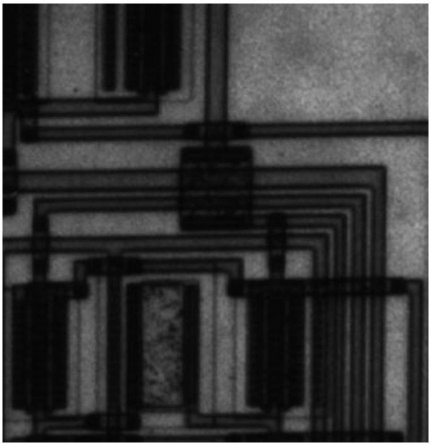
```
Input
10,10,10,10,10,10,10,10,10
10, 7, 7, 7,10,10,10,10,10
10, 7, 7, 7,10,10,10,10,10
10, 7, 7, 7,10,10,10,10,10
10,10,10,10,10,10,10,10,10
10,10,10,10,10, 2, 2, 2,10
10,10,10,10,10, 2, 2, 2,10
10,10,10,10,10, 2, 2, 2,10
10,10,10,10,10,10,10,10,10
```



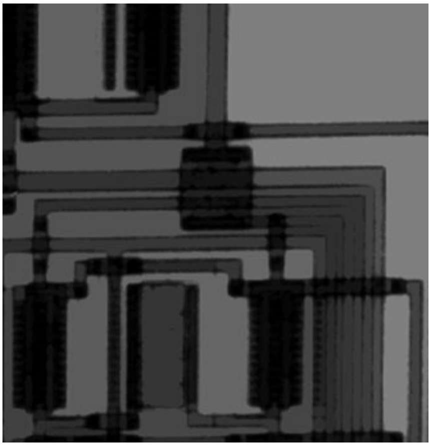
```
Output
10,10,10,10,10,10,10,10,10
10,10,10,10,10,10,10,10,10
10,10,10,10,10,10,10,10,10
10,10,10,10,10,10,10,10,10
10,10,10,10,10,10,10,10,10
10,10,10,10,10, 6, 6, 6,10
10,10,10,10,10, 6, 6, 6,10
10,10,10,10,10, 6, 6, 6,10
10,10,10,10,10,10,10,10,10
```

Illustration
Input image Thick film circuit
File number 136
Command *imx*(*75*)

Left: original



Right: after processing



41.175 ine

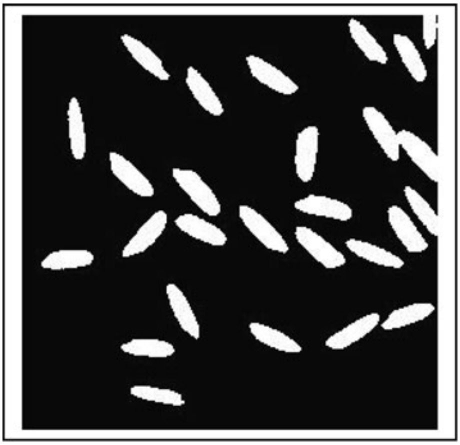
Format	$b = ine(a)$
Arguments	a Tolerance value for intensity difference
Defaults	a 0
Output values	Number of pixels
Image type	Grey
Type	Image comparison
Function	Generate an image in which white pixels indicate that the difference in the intensities between corresponding pixels in the current and alternate images is less than or equal to the given parameter, a. The output b is the number of pixels pairs that are nearly equal
Switch images	Yes
Typical uses	Image comparison
Remarks	*ine* tests for equality of intensity values

41.176 iob

Format	*iob*(*a*, *b*, *c*)	
Arguments	*a*	*X*-coordinate of a point, or a vector containing several values
	b	*Y*-coordinate of a point, or a vector containing several values
	c	Connectivity (4 or 8)
Defaults	c	*8*
Output values	None	
Image type	Binary	
Type	Blob shape analysis	
Switch	Yes	
Function	Isolate that *c*-connected blob (c = 4 or 8), which overlaps a given point [*a*, *b*].	
Typical uses	Blob analysis	
Remarks	*iob* can also be called with vector inputs, as illustrated below. Also see *cob*	
Illustration		
Input image	Rice grains (file number 3) after thresholding	
Image generation	*rni*(*3*), *enc*, *thr*	
Commands	[*h*, *w*] = *size*∗*current*;	% Image size
	i = *round*(*w*∗*rand*(*100*, *1*));	% 100 *X*-coordinates generated at random
	j = *round*(*h*∗*rand*(*100*, *1*));	% 100 *Y*-coordinates generated at random
	iob(*i*, *j*)	% Select up to 100 rice grains at random

Left: original

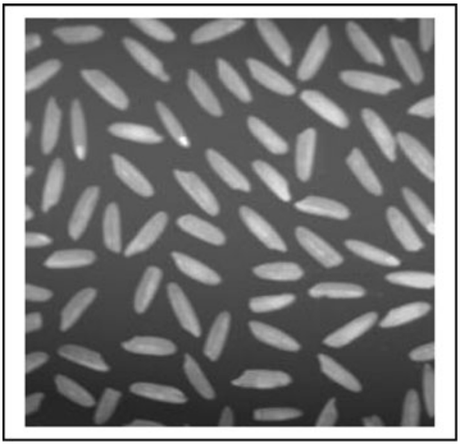
Right: after processing



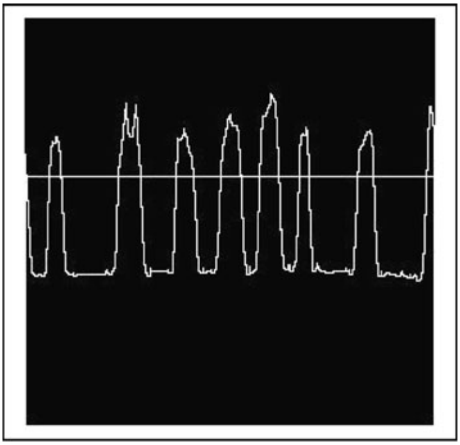
41.177 ipc/ipr

Format	*ipc*(*a*) *ipr*(*a*)		
Arguments	*a*	Column/row for plotting	
Defaults	*a*	Index number for the column/row through the centre of the image	
Output values	None		
Image type	Grey scale		
Type	Measurement and analysis		
Function	Plot the intensity profile along a defined column/row in the current image. The plot is drawn in the current image		
Switch images	Yes		
Typical uses	The two functions are used principally as aids to understanding the structure of an image and in guiding algorithm design		
Remarks	Also see *plt*. *ipc* and *ipr* are suitable for use in the remotely operated versions of QT, whereas *plt* is not. *plt* provides an annotated and more precise graph. Axes:		
	ipc	X-axis	Intensity
		Y-axis	Vertical position in the input image
	ipr	X-axis	Horizontal position in the input image
		Y-axis	Intensity
Illustration			
Input image	Rice grains		
File number	3		
Command	*ipr*(*100*)		

Left: original



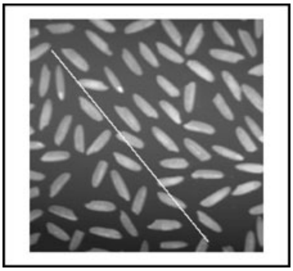
Right: after processing



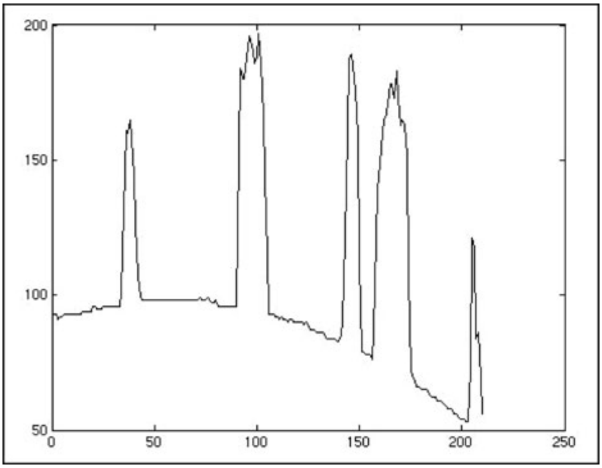
41.178 ipl

Format	*ipl*(*x1*, *y1*, *x2*, *y2*)	
Arguments	*x1*	X-coordinate of one end of the sampling line
	y1	Y-coordinate of one end of the sampling line
	x2	X-coordinate of the other end of the sampling line
	y2	X-coordinate of the other end of the sampling line
Defaults	*x1*	Half image width
	y1	1
	x2	Half image width
	y2	Height of image
Output values	None	
Image type	Grey	
Type	Drawing	
Function	Plot the intensity profile along a defined line in the current image. The plot is drawn in a new figure	
Switch images	Yes	
Typical uses	*ipl* is used principally as an aid to understanding the structure of an image and in guiding algorithm design	
Remarks	If no arguments are supplied, a graph is drawn of the intensities found along a vertical line through the centre of the image. Also see *plt*, *lsc* and *scc*	
Illustration		
Input image	Rice grains	
File number	3	
Command	*ipl*(*25*, *35*, *196*, *245*)	

Left: input image



Right: intensity profile Abscissa, horizontal position. Ordinate, intensity



41.179 ipp

Format	$b = ipp(a)$
Arguments	a Proportion of picture area ($0 \leq a \leq 1$)
Defaults	a 0.5
Output values	b Intensity level
Image type	Grey-scale
Type	Grey-scale image measurement
Function	Calculate the intensity level b such that a proportion a of the pixels have intensity values $\leq b$
Switch images	No
Typical uses	Finding a suitable threshold parameter by analysing the intensity histogram
Remarks	Also see *enc*, *air*, *pth*

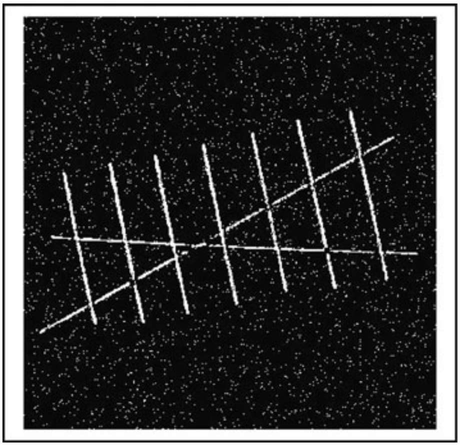
41.180 irt

Format	*irt*
Arguments	None
Defaults	None
Output values	None
Image type	Grey scale or binary
Type	Global transform
Switch	Yes
Function	Inverse Radon transform
Switch images	Yes
Typical use	Isolating individual linear streaks in noisy images
Remarks	Normally, *irt* is used in conjunction with *rdn* (Radon transform). The example shown below illustrates how a group of nearly vertical, parallel lines can be identified. The Radon transform image (RT) is first weighted by multiplying it by the intensity wedge (*wgx*, *neg*). The contrast is then adjusted by applying *sqr*. (This reduces the number of 'noise' points in the final image.) Finally, *irt* is applied (Also see *rdn* and *slf*)

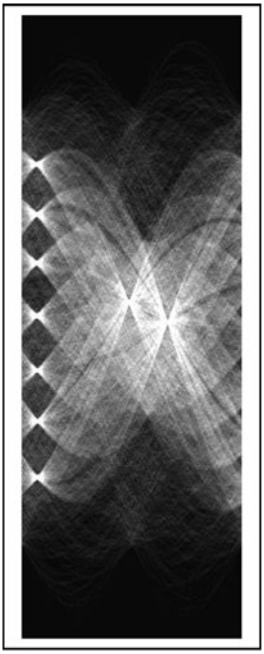
Illustration

Input image	Lines
File number	160
Command	*rdn*, *wri*, *wgx*, *neg*, *rei*, *mul*, *sqr*, *irt*

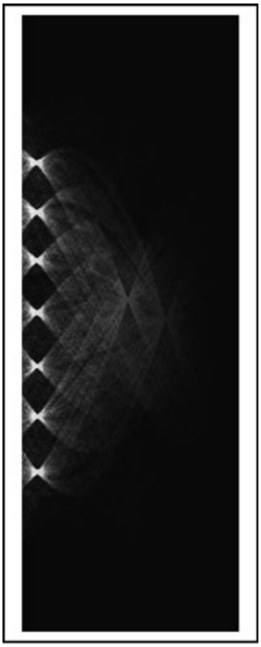
Top-left: original



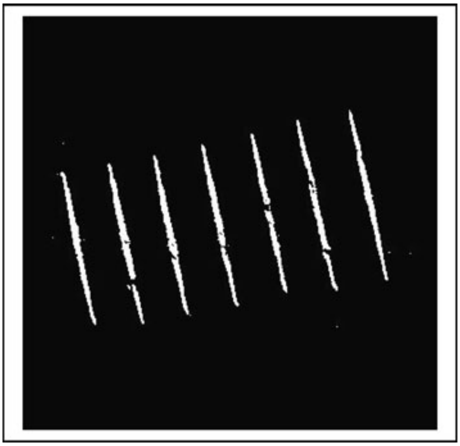
Top-right: just after *rdn*



Bottom-left: after *sqr*



Bottom-right: after *irt*



41.181 isight

Format	*isight*(*a*)
Arguments	*a* *Image file number*
Defaults	*a* *Latest* (*i.e., highest numbered*) *image file present in the folder*
Output values	*None*
Type	Image acquisition
Function	Acquire an image from an iSight camera using iStill software (Apple Computer Inc)
Switch images	Yes
Remarks	The *iSight* camera is no longer available. Use *frz* instead.

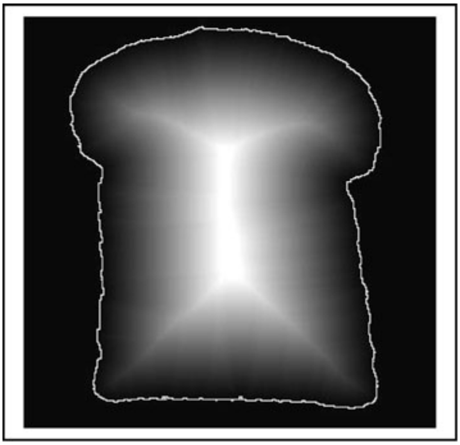
41.182 iso

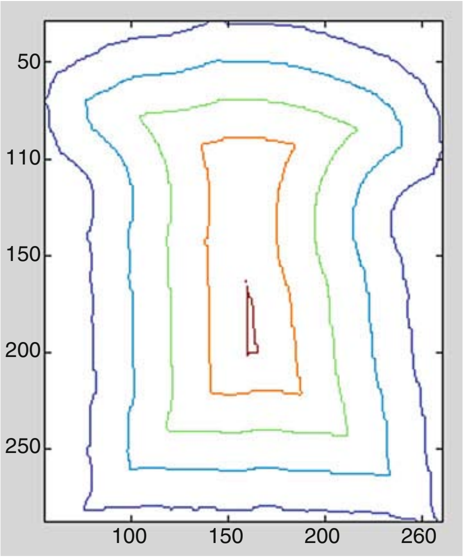
Format	*iso*
Arguments	*None*
Defaults	Not applicable
Output values	None
Image type	Grey scale
Type	Image analysis
Switch	Yes
Function	Draw intensity contours (isophotes) in Figure 6
Typical uses	Visualising slow intensity variations that are difficult to see otherwise
Remarks	The contour image disappears when the user clicks anywhere in Figure 6. Also see *ctr*

Illustration

Input image	Slice of bread (file number 45), after grass-fire transform (*gft*)
Image generation	*rni*(45), *gft*, *enc*
Commands	*iso*

Left: original (slice outline added for clarity) *Right*: after processing





41.183 itl

Format	*[x, y] = M(xl, yl, x2, y2, x3, y3, x4, y4)*	
Arguments	[*xi*, *yi*]	Coordinates of the ith point (*i* = *1*, *2*, *3*, *4*)
Output values	[*x*, *y*]	Coordinates of the point of intersection
Image type	Not relevant	
Type	Geometry	
Function	Find the point of intersection of two lines, each defined by two points. Let *A* be the line joining points [*x1*, *y1*] *and* [*x2*, *y2*]. *B* be the line joining points [*x3*, *y3*] *and* [*x4*, *y4*]. *itl* finds the coordinates of the point of intersection of lines *A* and *B*	
Switch images	No	
Typical uses	Analysing feature geometry	
Remarks	Notice that is either line is vertical, an error will occur	

41.184 itx

Format *itx*(*a*, *b*, *c*)

Arguments *a* String to be displayed

b X-coordinate of the left-most point of the arrow

c Y-coordinate of the left-most point of the arrow

Defaults *b* and *c* Use the cursor to select the position

Output values None

Image type Any

Type Image annotation

Switch No

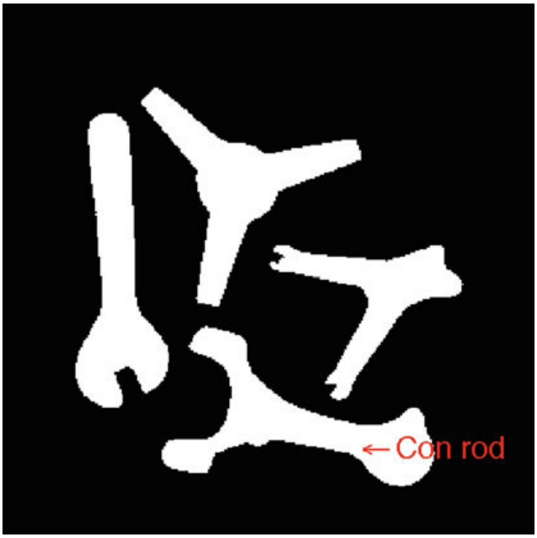
Function Superimpose text over the current image. An arrow to the left of the text allows small feature to be highlighted

Switch images No

Typical uses Annotate the current image

Remarks The text is written into an *alpha channel* associated with the Current image. It is not part of the image and will disappear when the next operation is performed. Neither the Current nor the Alternate image is altered. If only one argument (c) is specified, text with intensity *c* is placed to the right of a point selected by the cursor

Illustration



41.185 ity

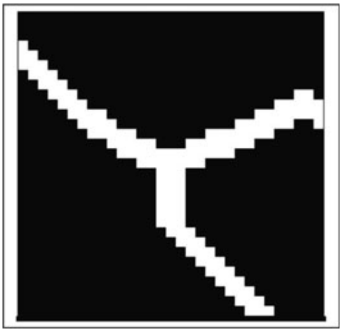
See *hsi*

41.186 jas/las

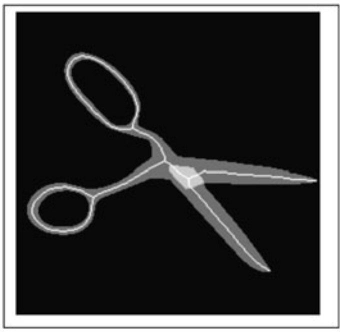
Format		d = jas(a, b, c)
		d = las(a, b, c)

Arguments	*a*	Select which skeleton junction/limb end is to be used
	b	Fixes the size of the region selected (size = $2b \times 2b$)
	c	Dilation control
Defaults	*a*	2
	b	10
	c	2
Output values	*d*	Structuring element (SE)
Image type	Binary	
Type	Designing Structuring Elements for feature recognition	
Function	1.	The *a*th skeleton junction (*jas*) or limb end (*las*) is found (The numbering is the same as that used by *ndo*)
	2.	The blob found in step 1 is shrunk to a single point
	3.	Find the position $[x, y]$ of that point
	4.	Dilate the skeleton using *dil*(*c*), where *c* is the 3rd argument supplied to *jas/las*
	5.	Place a window of size $(2b + 1)$ x $(2b + 1)$ pixels centered on point $[x, y]$.
	6.	Use the contents of this window as the Structuring Element, which is output as *d*
Switch images	Yes	
Typical uses	Noise reduction in binary images; simplifying them, prior to blob measurement/shape analysis	
Remarks	*jas* and *las* are tools for experimenting with erosion/dilation. They are not intended for batch processing. Also see *use*, *gbd*, *gbe*	
Illustration		
Input image	Scissors	
File number	48	
Command	test (See listing above)	

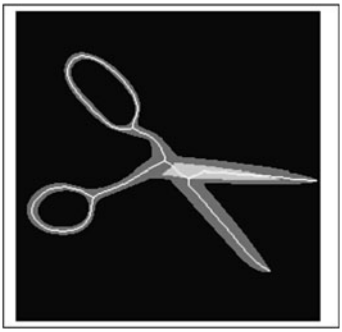
Top-left: SE generated by *jas*



Top-right: result of processing by *jas*(*4*, *15*, *2*)



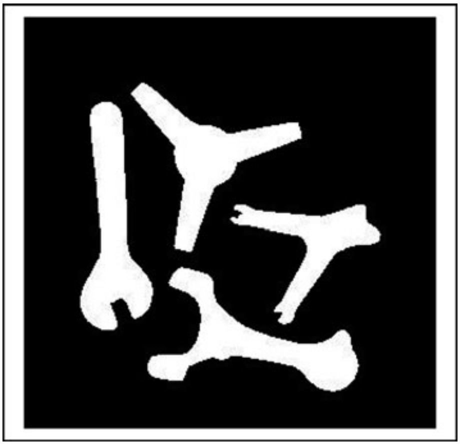
Bottom-left: result of processing by *las*(*2*, *35*, *5*)



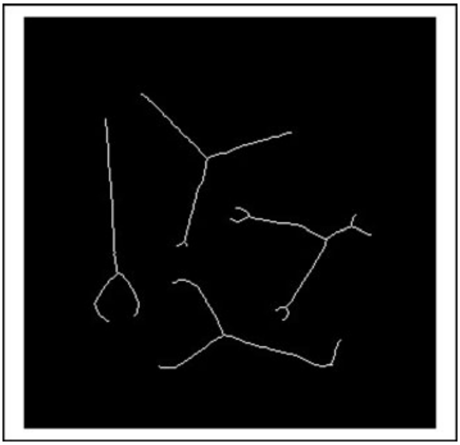
41.187 jnt/lmb

Format	*jnt* *lmb*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob shape analysis
Switch	Yes
Function	Locate the points on a ‘match-stick’ figure (the skeleton), generated by *ske* or *thn*, where three or more limbs meet (*jnt*), or where limbs end (*lmb*)
Typical uses	Representing shapes by measuring distances and angles on the ‘match-stick’ figure. These may be measured between limb ends and/or joints
Remarks	The ‘match-stick’ figure can be ‘broken’, so that each limb is separated from the others, using the command sequence *jnt*, *enc*
Illustration	
Input image	Four metal parts, with *thn* (thinning) applied to create a ‘match-stick’ figure
Image generation	*rni*(*46*), *thr*, *thn*
Commands	Joints: *rni*(*46*), *thr*, *thn*, *wri*, *jnt*, *dil*(*5*), *rei*, *mxi* *Limb ends*: *rni*(*46*), *thr*, *thn*, *wri*, *lmb*, *dil*(*5*), *rei*, *mxi*

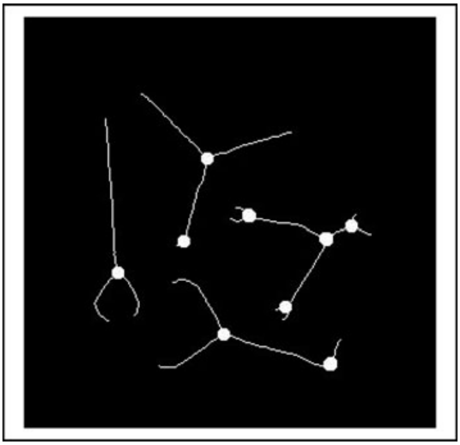
Top-left: original image (file number 46)



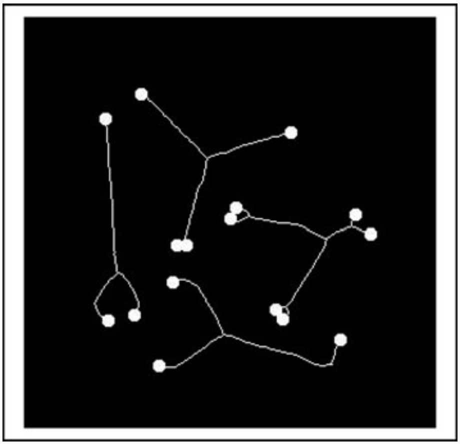
Top-right: rni(*46*), *thr, thn*



Bottom-left: enlarged joints superimposed on the 'match-stick' figure



Bottom-right: enlarged limb ends superimposed on the 'match-stick' figure



41.188 joy

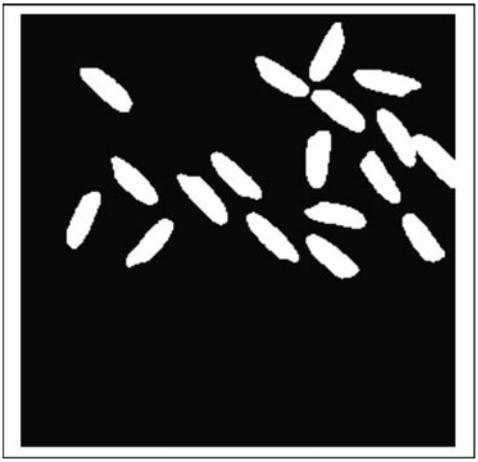
Format	*joy*(*a*)
Arguments	*a* Select recording to play
Default	a = 3, Play 'gong' sound
Output values	None
Image type	Not relevant
Type	User interface
Function	Play one of several pre-recorded sounds
Switch images	No
Typical uses	Providing an audible signal indicating the progress of program execution
Remarks	These recordings are provided as standard with MATLAB and have been played successfully on a Macintosh computer, running OS X.

41.189 kbb

Format	*kbb*(*a*)	
Arguments	a	Rank size of smallest blob to be kept
Defaults	a	2
Output values	None	
Image type	Grey	
Type	Binary image analysis	
Function	Keep the blob of rank size *a* and all blobs larger than it	
Switch images	Yes	
Typical uses	Noise reduction in binary images; simplifying them, prior to blob measurement/shape analysis	
Remarks	Also see *big*, *kgr*	
Illustration		
Input image	Rice grains	
File number	3	
Command	enc, thr, *kbb*(*17*)	

Left: original

Right: after processing

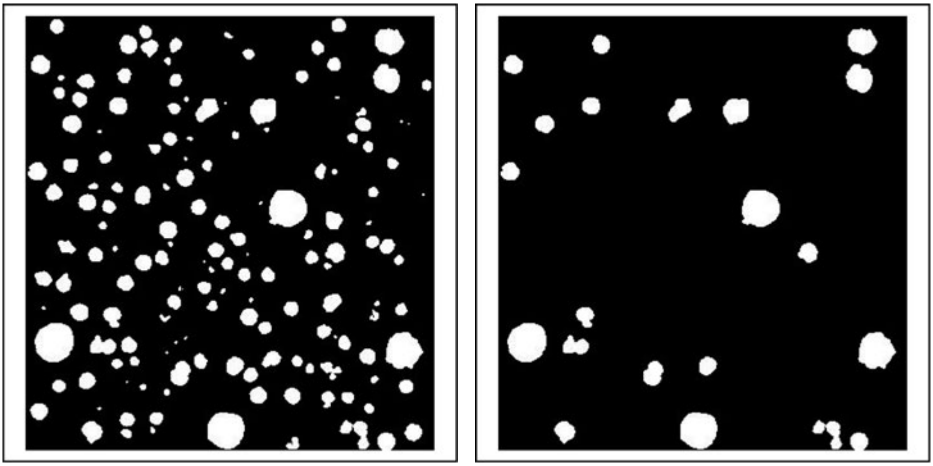


41.190 kgr

Format	*kgr*(*a*, *b*)	
Arguments	*a*	Lower size limit
	b	Connectivity (b takes only values 4 or 8)
Defaults	*b*	8
Output values	None	
Image type	Binary	
Type	Blob transform	
Function	Eliminate all *b*-connected blobs (*b* = 4 or 8) whose area is less than *a* pixels	
Switch images	Yes	
Typical uses	Reducing the effects of noise after filtering and thresholding	
Remarks	Also see *ndo* and *big*	
Illustration		
Input image	Non-standard; contrived for this illustration	
Command	*kgr*(*150*)	

Left: original

Right: after processing



41.191 las

See *jas.*

41.192 led

See *ced*

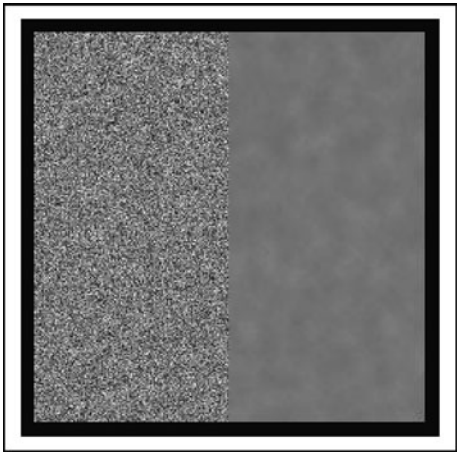
41.193 ley

Format	*ley*(*a*, *b*)
Arguments	*a* Width of the filter window
	b Height of the filter window
Defaults	One argument supplied: b = a
	No arguments supplied: a = b = 9
Output values	None
Image type	Grey
Type	Non-linear local operator
Function	The local entropy of the intensity values within each rectangular neighbourhood of size a × b defines the intensity for the central pixel
Switch images	Yes
Typical uses	Segmenting textured images
Remarks	Also see *lsd*

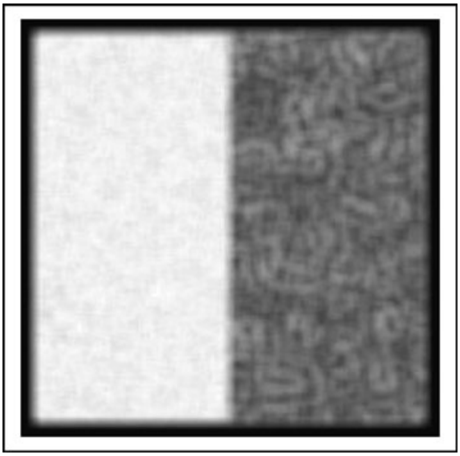
Illustration

Input image	Generated using *gsn* and *caf*. Both halves of the original image shown below have the same average intensity
Command	*ley*(*9*, *9*)

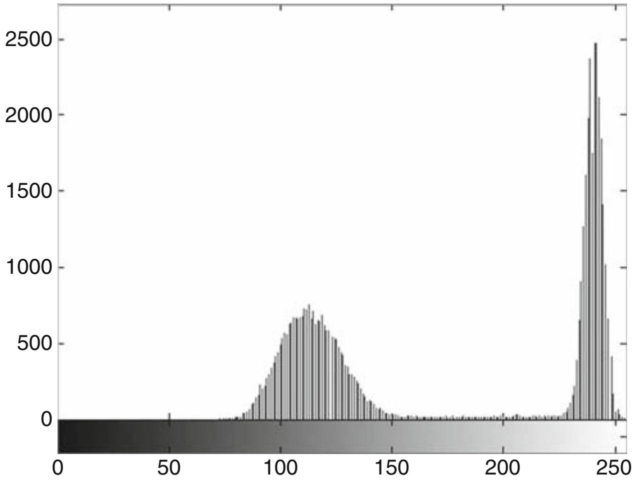
Left: original



Right: after processing



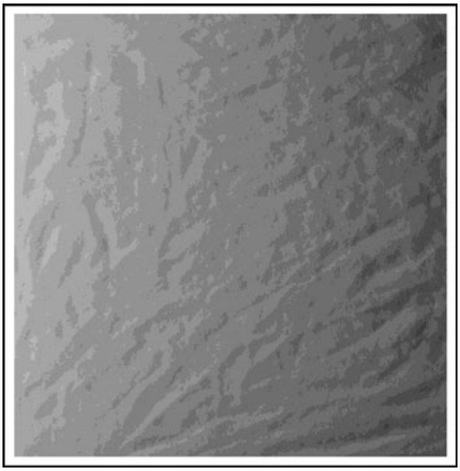
Bottom: Intensity histogram of the processed image (Edge effects were removed by masking first)



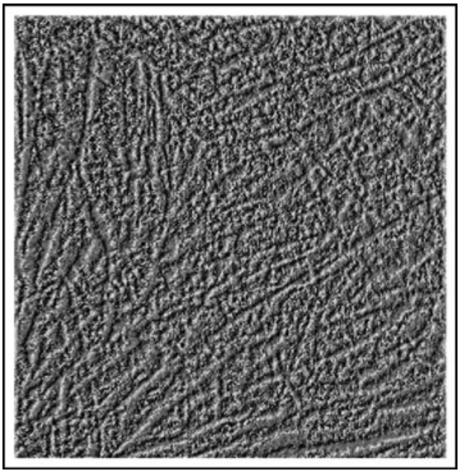
41.194 lhq

Format	*lhq*(*a*, *b*)	
Arguments	*a*	Processing window width (normally an odd number)
	b	Processing window height (normally an odd number)
Defaults	*a*	7
	b	7
Output values	None	
Image type	Grey scale	
Type	Non-linear filter	
Switch	Yes	
Function	Local-area histogram equalisation	
Definition	Within that window of size a × b pixels centred on point (*i*, *j*), *N*(*i*, *j*), the number of pixels that are darker than *A*(*i*, *j*) is computed. After rescaling, this number defines the intensity *C*(*i*,*j*) in the output image	
Switch images	Yes	
Typical uses	Pre-processing for texture analysis	
Remarks	Also see *sdm*	
Illustration		
Input images	Creased fabric	
File numbers	28	
Commands	*lhq*(*15*, *15*)	

Left: original image



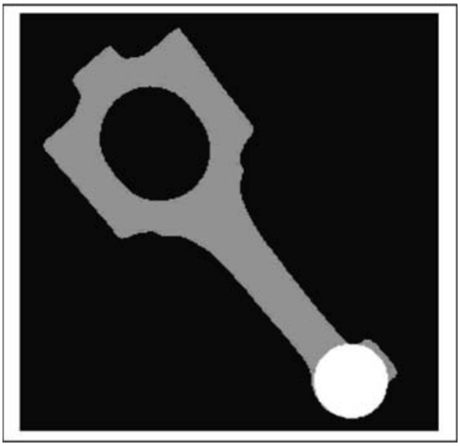
Right: after processing



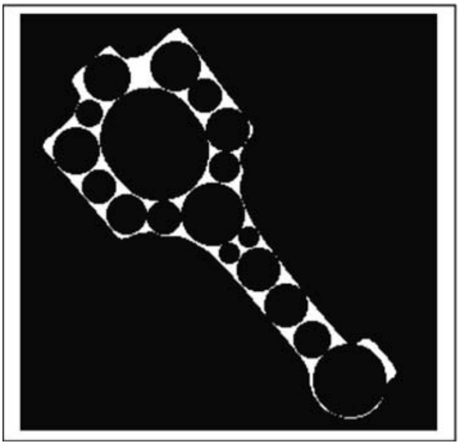
41.195 lic/mic

Format	*[x, y, z] = lic*	
	mic(a)	
Arguments	*a*	Limit on no. of white points to continue recursion (*mic* only)
Default value	a	500
Output values	*x*	X-coordinate of the inscribed circle (*lic* only)
	y	X-coordinate of the inscribed circle (*lic* only)
	z	Radius of the inscribed circle (*lic* only)
Image type	Binary	
Type	Binary image analysis	
Function	*lic*:	Fit the largest inscribed circle into the white regions
	mic:	Apply *lic* recursively
Switch images	Yes	
Typical uses	Placing a magnetic or vacuum gripper Choosing structuring element for binary erosion	
Remarks	*mic* is often slow, since applies *lic* recursively	
Illustration		
Input image	Automobile con-rod	
File number	47	
Commands	*lic*:	*lic, adi*
	mic:	*mic*

Top-left: after *lic, adi*



Top-right: after *mic(500)*



41.196 lmi

Format	$a = lmi$
Arguments	None
Defaults	Not applicable
Output values	*a* Orientation of the principal axis
Image type	Binary
Type	Blob measurement
Function	Calculate the orientation of the principal axis of a blob-like figure
Switch images	No
Typical uses	Finding the orientation of a blob
Remarks	If there is more than one blob, *lmi* computes the orientation of just one: the first blob encountered during a top-to-bottom, left-to-right raster scan. The subsidiary function *blob_parameters* is used in several other functions. The principal axis is also called the axis of minimum second moment

41.197 lnb

Format	*lnb*(*a*)
Arguments	*a* Size of processing window
Defaults	*a* 3
Output values	None
Image type	Grey scale or binary
Type	Non-linear filter
Switch	Yes
Function	Largest (i.e., brightest) neighbour in an $a*a$ window
Typical use	Noise reduction. Component of a number of other operators (e.g., *lnb*, *sub* is a highly effective edge detector)
Remarks	*lnb*(*a*) is equivalent to *dil*(*a*, *1*) and is included in QT to maintain compatability with earlier systems. Also see *snb*

41.198 lpc

See *con*

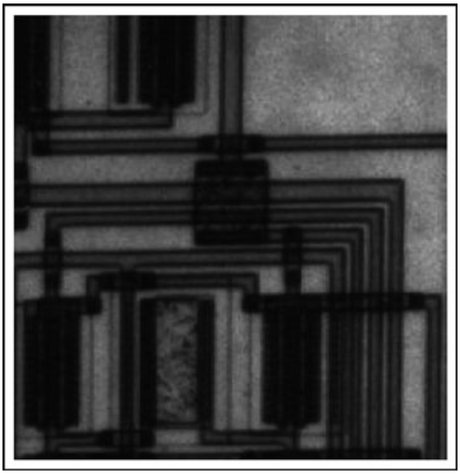
41.199 lpf/hpf

Format	*lpf* *hpf*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey scale
Type	Image filtering
Switch	Yes
Function	*lpf* Local averaging of the intensities within each 3∗3 window *hpf* Result of applying lpf subtracted from the original image
Switch images	Yes
Typical uses	
Remarks	These two functions are included to retain compatability with earlier systems. *lpf* reduces noise slightly but also introduces a slight blurring effect, while *hpf* emphasises sharp edges, small spots, thins arcs/lines and textures. *hpf* also enhances camera noise. *lpf* may be applied repeatedly (typically 2–10 times), to produce a more pronounced blurring effect. Also see *raf*, *caf*

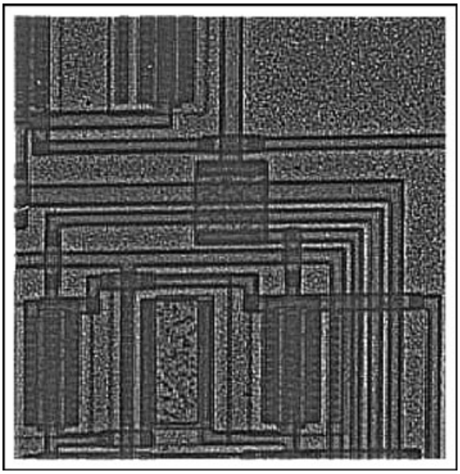
Illustration

Input image	Circuit
File number	136
Command	*hpf*, *enc* (Note: *enc* is included to improve the contrast)

Top-left: original



Top-right: after processing



41.200 lrt

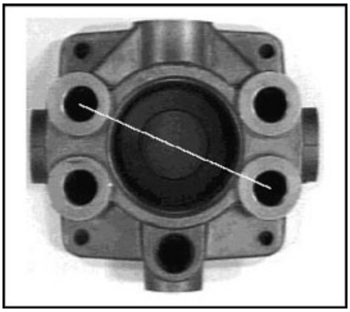
Format	*lrt*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey scale or binary
Type	Affine
Switch	Yes
Function	Invert the horizontal axis
Switch images	Yes

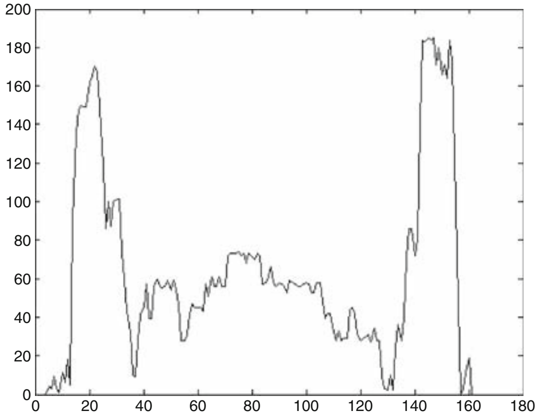
41.201 lsc

Format	*a = lsc(x1, y1, x2, y2, n)*	
Arguments	*x1*	X-coordinate of one end of the scan line
	y1	Y-coordinate of one end of the scan line
	x2	X-coordinate of the other end of the scan line
	y2	Y-coordinate of the other end of the scan line
	n	Number of sample points along this line
Defaults	*n*	Determined automatically from [*x1*, *y1*] and [*x2*, *y2*].
Output values	*a*	Vector containing *n* intensity values
Image type	Grey	
Type	Image analysis	
Switch	No	
Function	Line-scan; sample the intensities uniformly along the line [*x1*, *y1*] to [*x2*, *y2*].	
Typical uses	Reducing processing time, by converting a 2D image into 1D vector. Determining periodicity in known directions	
Remarks	Also see *scc*	
Illustration		
Input image	Hydraulics manifold	
File number	37	
Command	*rni(37), z = lsc(44, 71, 199, 139), figure(2), plot(z), figure(1), vpl(44, 71, 199, 139, 255)*	

Left: original with scan line superimposed by *vpl.* (The line ends were specified manually.)

Right: after *plot*(*z*)
Horizontal axis, scan position
Vertical axis, intensity

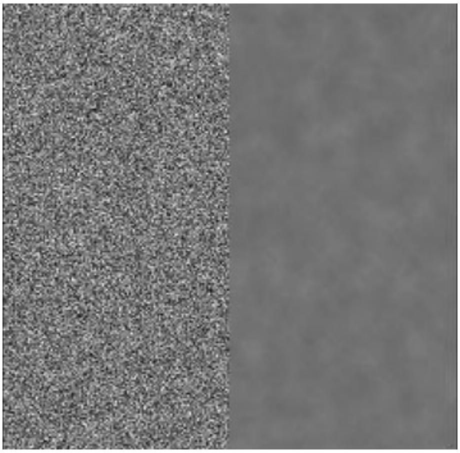




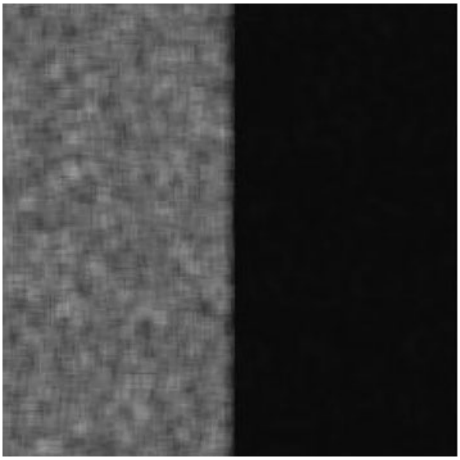
41.202 lsd

Format	*lsd*(*a*, *b*)
Arguments	*a* Width of the filter window
	b Height of the filter window
Defaults	One argument supplied: $b = a$
	No arguments supplied: $a = b = 5$
Output values	None
Image type	Grey
Type	Non-linear local operator
Function	The standard deviation of the intensity values within each rectangular neighbourhood of size a × b defines the intensity for the central pixel
Switch images	Yes
Typical uses	Segmenting textured images
Remarks	Also see *ley*
Illustration	
Input image	Generated using *gsn* and *caf*. Both halves of the original image shown below have the same average intensity
Command	*lsd*(*7*, *7*)

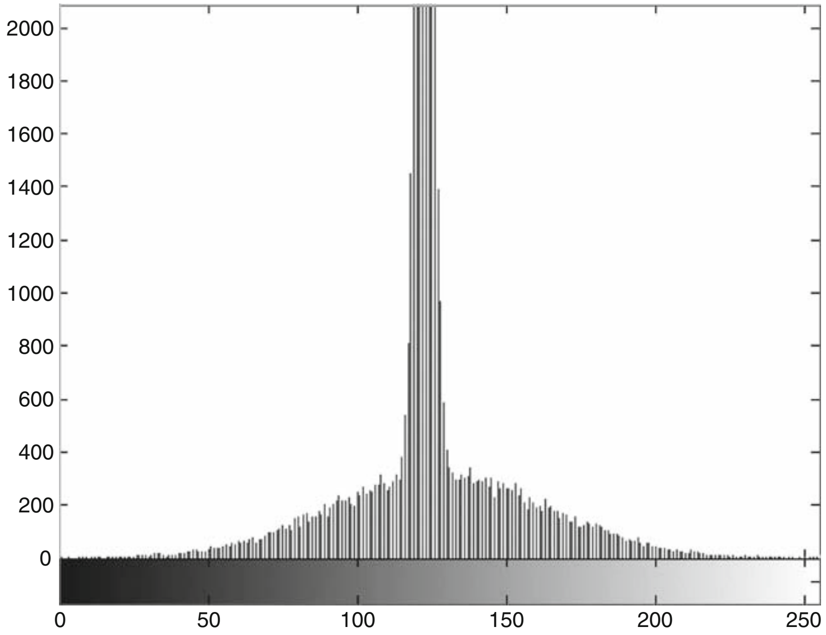
Left: original



Right: after processing



Bottom: *Intensity histogram of the original image*

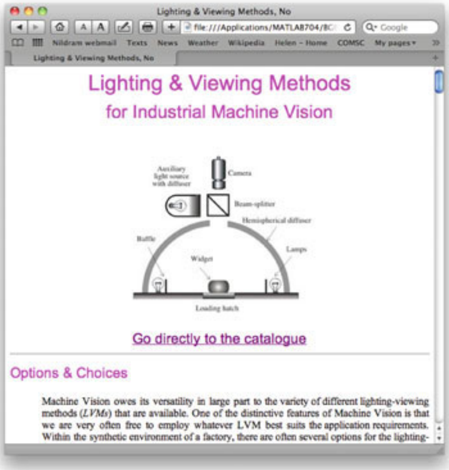


41.203 lvm/lvs

Format	*lvm*(*a*)	
Arguments	*a*	String to be used in the search
Defaults	*a*	View the home page for the Catalogue of Lighting-Viewing Methods (LVM. See Chapters 8 and 40)
Output values	None	
Image type	Not relevant	
Type	Information/advice for the user	
Function	*lvm*	Display the LVM home page
	lvm(‘*string*’)	Search each LVM page for *string*
	lvs({‘*string1*’, ‘*string2*’, . . .})	Search each LVM page for *string1 and string2 and . . .*
Switch images	No	

Typical Web Pages, user typed *lvm*

Top-left:	QT's immediate response to command *lvm*
Top-right:	User clicked on hyperlinked text, '*Go directly to the catalogue*'
Bottom-left:	User clicked on hyperlinked text, '*Omni-directional illumination*'
Bottom-right:	User clicked on hyperlinked text, '*Get it!*' (next to 'Optical set-up')



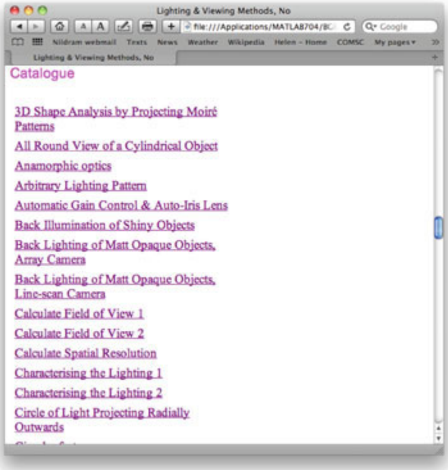
Lighting & Viewing Methods

for Industrial Machine Vision

Go directly to the catalogue

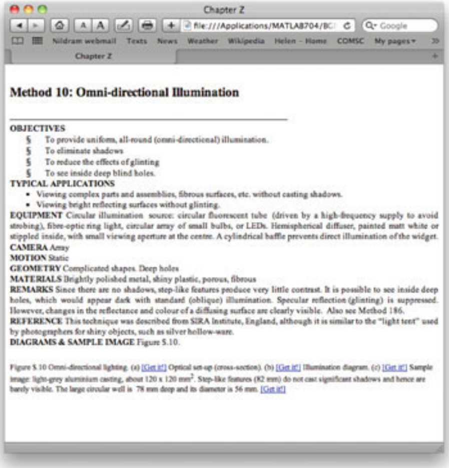
Options & Choices

Machine Vision owes its versatility in large part to the variety of different lighting-viewing methods (*LVMs*) that are available. One of the distinctive features of Machine Vision is that we are very often free to employ whatever LVM best suits the application requirements. Within the synthetic environment of a factory, there are often several options for the lighting-


Lighting & Viewing Methods, No
file:///Applications/MATLAB704/BC
Google
Nildram webmail
Texts
News
Weather
Wikipedia
Helen - Home
COMSC
My pages
Lighting & Viewing Methods, No

Catalogue

3D Shape Analysis by Projecting Moiré Patterns
All Round View of a Cylindrical Object
Anamorphic optics
Arbitrary Lighting Pattern
Automatic Gain Control & Auto-Iris Lens
Back Illumination of Shiny Objects
Back Lighting of Matt Opaque Objects, Array Camera
Back Lighting of Matt Opaque Objects, Line-scan Camera
Calculate Field of View 1
Calculate Field of View 2
Calculate Spatial Resolution
Characterising the Lighting 1
Characterising the Lighting 2
Circle of Light Projecting Radially Outwards



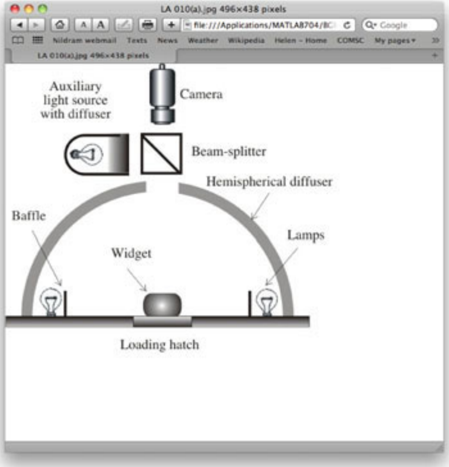
Method 10: Omni-directional Illumination

OBJECTIVES
§ To provide uniform, all-round (omni-directional) illumination.
§ To eliminate shadows
§ To reduce the effects of glinting
§ To see inside deep blind holes.

TYPICAL APPLICATIONS
- Viewing complex parts and assemblies, fibrous surfaces, etc. without casting shadows.
- Viewing bright reflecting surfaces without glinting.

EQUIPMENT Circular illumination source: circular fluorescent tube (driven by a high-frequency supply to avoid strobing), fibre-optic ring light, circular array of small bulbs, or LEDs. Hemispherical diffuser, painted matt white or stippled inside, with small viewing aperture at the centre. A cylindrical baffle prevents direct illumination of the widget.
CAMERA Array
MOTION Static
GEOMETRY Complicated shapes. Deep holes
MATERIALS Brightly polished metal, shiny plastic, porous, fibrous
REMARKS Since there are no shadows, step-like features produce very little contrast. It is possible to see inside deep holes, which would appear dark with standard (oblique) illumination. Specular reflection (glinting) is suppressed. However, changes in the reflectance and colour of a diffusing surface are clearly visible. Also see Method 186.
REFERENCE This technique was described from SIRA Institute, England, although it is similar to the "light tent" used by photographers for shiny objects, such as silver hollow-ware.
DIAGRAMS & SAMPLE IMAGE Figure S.10.

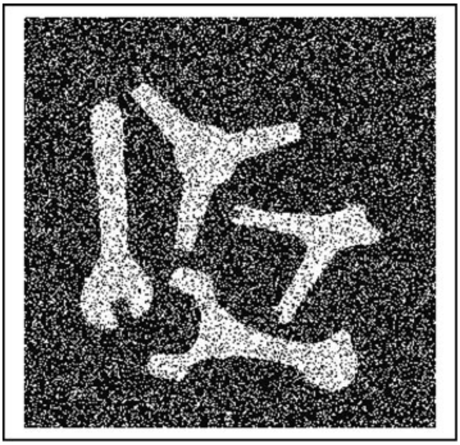
Figure S.10 Omni-directional lighting. (a) [Get it!] Optical set-up (cross-section). (b) [Get it!] Illumination diagram. (c) [Get it!] Sample image: light-grey aluminium casting, about 120 x 120 mm^2. Step-like features (82 mm) do not cast significant shadows and hence are barely visible. The large circular well is 78 mm deep and its diameter is 56 mm. [Get it!]



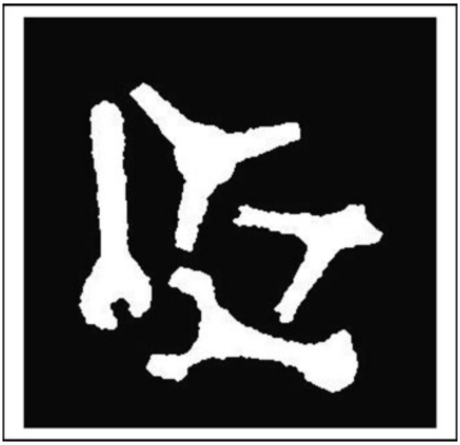
41.204 maj

Format	*maj*(*a*)
Arguments	*a* Multiplier for current image, before addition
Defaults	*a* Indefinitely large
Output values	None
Image type	Binary
Type	Binary image transformation
Switch	Yes
Function	Majority-vote edge smoothing on a binary image
Definition	Majority voting is applied within a 3 × 3 window. The central pixel is set to white, if there are more than 4 white points within the window. If there are four or fewer white pixels, the central pixel is set to black. This process is repeated *a* times. If *a* is omitted, or set to infinity, the process stops when no change takes place
Typical uses	Feature extraction in noisy images
Remarks	
Illustration	
Input image	Four metal parts with salt and pepper noise added
Image generation	*rni*(*46*), *spn*(*0.4*)
Commands	*maj*

Left: original

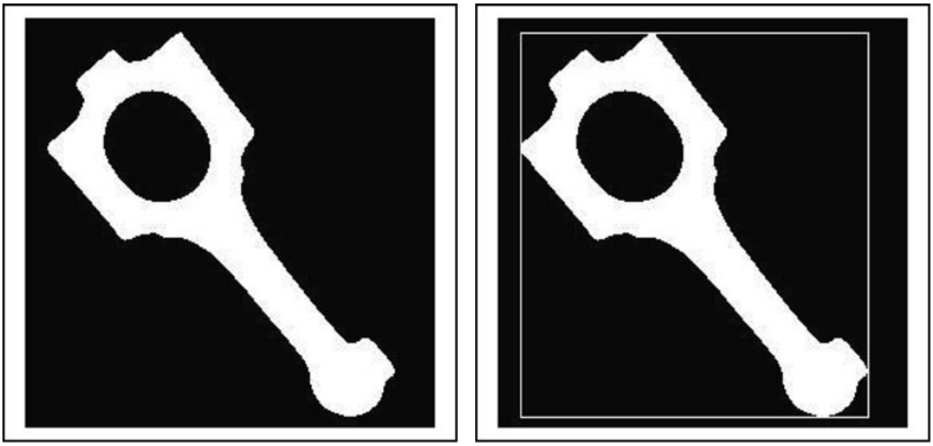


Right: after *maj*



41.205 mar

Format	[*a*, *b*, *c*, *d*] = *mar*	
Arguments	None	
Defaults	Not applicable	
Output values	*a*	Minimum X limit
	b	Minimum Y limit
	c	Maximum X limit
	d	Maximum X limit
Image type	Binary	
Type	Blob transform	
Function	Find the (X,Y)-coordinates describing the bounding box. That is the minimum area rectangle aligned with the sides of the image that encloses all white pixels	
Switch images	Yes	
Typical uses	Blob analysis	
Remarks	To obtain the 'filled' minimum-area rectangle, use *mar*, *blf*. Also see *bbx*	
Illustration		
Input image	Con-rod	
File number	47	
Command	*thr*, *mar* (N.B. *thr* avoids the artifacts created by storing a binary image in JPEG format)	



Left: original

Right: after processing

41.206 mbc

Format $[x, y, z] = mbc(a)$

Arguments *a* Limit on number of iteration cycles within the auxiliary function *mbc_many*

Defaults *a* 5

Output values *x* X-coordinate of the centre of the fitted circle

y Y-coordinate of the centre of the fitted circle

z Radius of fitted circle

Image type Binary, with a single blob

Type Blob shape analysis

Function Minimum bounding circle

Switch images Yes

Typical uses Binary image analysis

Remarks This relies on an approximate heuristic procedure. The auxiliary function *mbc2* is applied first. This finds the size and position coordinates of that circle which passes through the two most distant edge points. In many cases this is sufficient. However, in those situations where it fails to enclose all points within the blob, the function *mbc_many* is applied. Initially, this fits a circle using *ceb*, then finds the protuberances (i.e., regions outside that circle). These are then used to fit another circle, again using *ceb*. This cycle is then repeated a total of *a* times, where *a* is the input parameter

Illustration

Input image Various

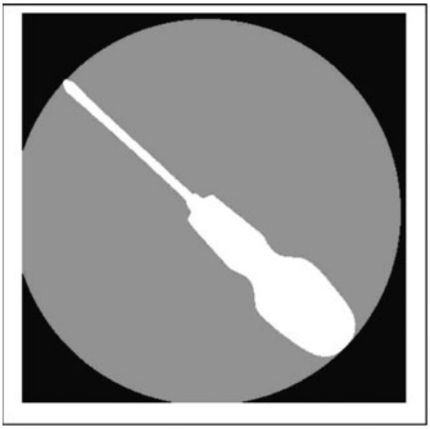
File numbers 43, screwdriver

49, scissors

175, plastic injection mouldings (6 malformed battery tops)

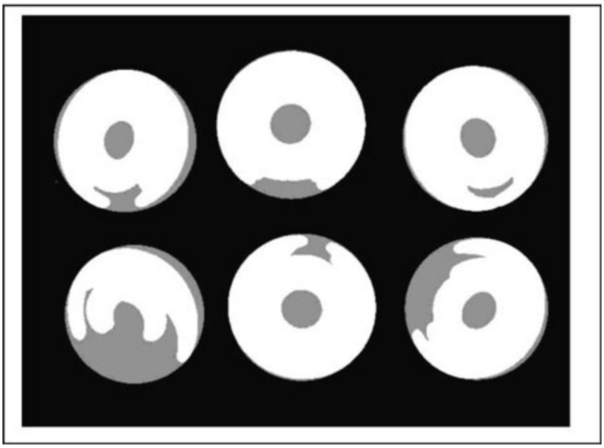
Commands After conversion to a binary image, the command sequence *mbc*, *adi* was applied

Top-left: screwdriver



Top-right: scissors

Bottom: plastic mouldings



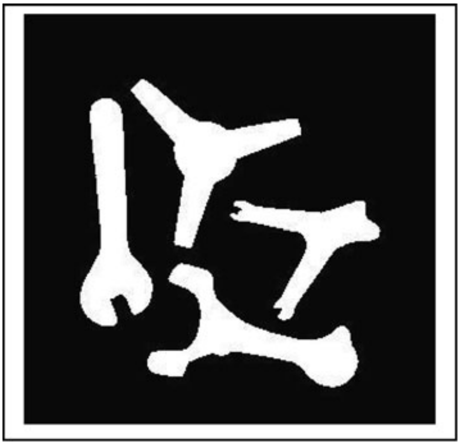
41.207 mch

Format	*mch*
Arguments	None
Output values	None
Image type	Binary
Type	Binary image analysis
Function	Detect the midpoint of each vertical chord crossing a white region
Switch images	Yes
Typical uses	Checking objects that are nominally symmetrical about a vertical axis
Remarks	*mch* provides a fast alternative to skeletonisation and is useful for certain applications where there is limited variability of form and orientation

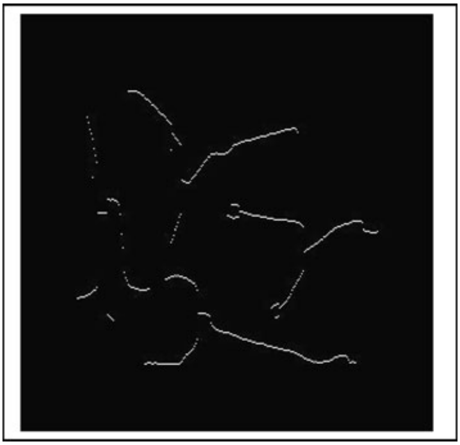
Illustration

Input image	4 piece parts
File number	46
Command	*thr*, *mch*

Top-left: original image



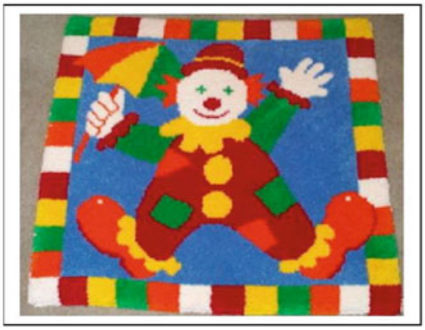
Right: after *mch*



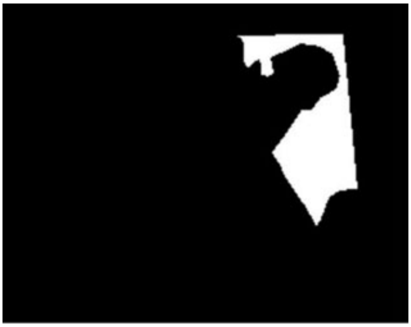
41.208 mci

Format	*mci*
Arguments	None
Defaults	Not applicable
Output values	None
Image types	Current: Colour image Alternate: Binary mask (same size as the current image)
Type	Colour image analysis
Function	Retain only those pixels in the current image that have the same address as a white pixel in the mask image. All other pixels are set to black
Switch images	Yes. The original colour image is retained
Typical uses	Selecting parts of an image for colour analysis, for example using the programmable colour filter, functions *shs and* pcf
Remarks	The binary mask image may conveniently be drawn using the function *pgn*
Illustration	
Input images	Current: image no. 354 Alternate: binary mask drawn using *pgn*
Command	*mci*

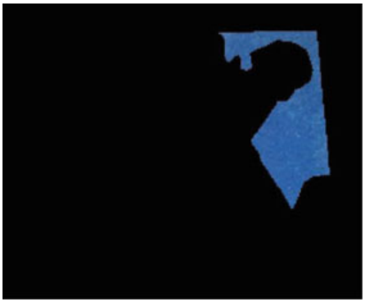
Left: colour image (current image)



Right: binary mask image (alternate image)



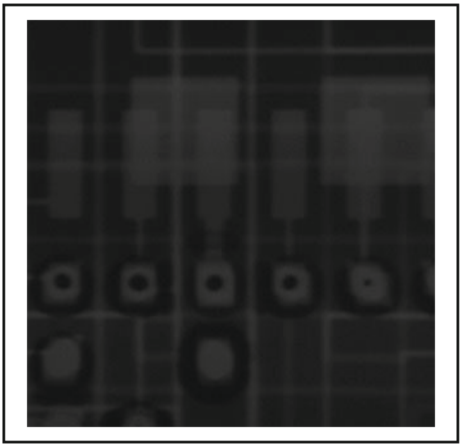
Bottom: after processing



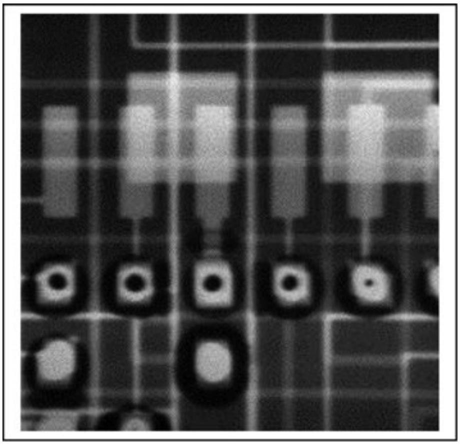
41.209 mcn

Format	*mcn*(*a*)
Arguments	a Intensity multiplier
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Monadic
Function	Multiply all intensities by a constant
Definition	C ⇐ a∗A with a hard limit placed at 255
Switch images	Yes
Typical uses	Increasing contrast
Remarks	Also see *air*, *blo*, *acn*, *din*, *hin*
Illustration	
Input image	PCB after shifting and rescaling intensities
File number	135
Command	*acn*(34), *hin*, *hin*, *hin*, *mcn*(*10*)

Left: original



Right: after processing



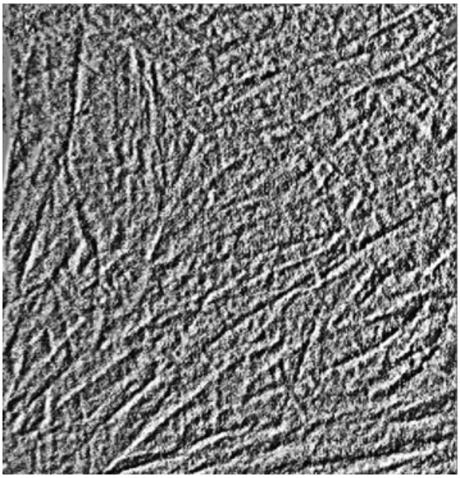
41.210 mdf

Format	*mdf*(*a*, *b*)	
Arguments	*a*	Processing window, width (normally an odd number)
	b	Processing window, height (normally an odd number)
Defaults	*a*	3
	b	3 (If only one argument is supplied, *b* = *a*)
Output values	None	
Image type	Grey scale	
Type	Non-linear filter	
Switch	Yes	
Function	Median filter	
Definition	Within that window of size a × b pixels centred on point (*i*, *j*), the median intensity value is calculated. The median value is the one placed [(ab + 1)/2] after sorting in rank order	
Switch images	Yes	
Typical uses	Pre-processing to reduce the effects of sensor noise	
Remarks	Very high spatial frequencies are attenuated. Edges are not blurred significantly	

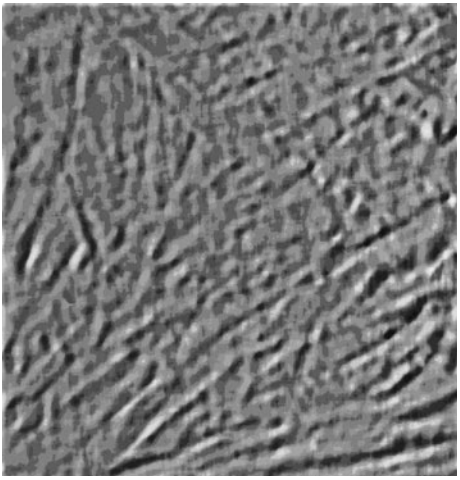
Illustration

Input images	Creased fabric, pre-processed by applying *lhq*(*15*, *15*)
File numbers	28
Command:	*mdf*(*9*, *9*)

Left: before *mdf*



Right: after *mdf*(*9*, *9*)

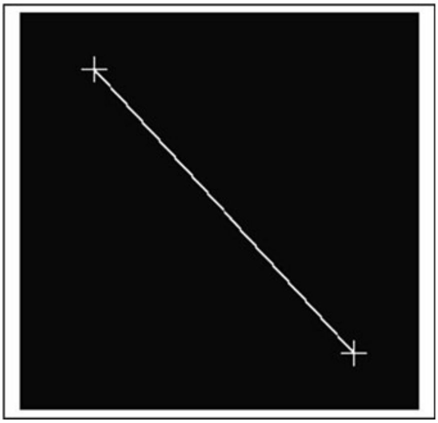


41.211 mdp

Format	[*xl*, *yl*, *x2*, *y2*] = *mdp*
Arguments	None
Output values	[*x1*, *y1*, *x2*, *y2*] Coordinates of the most distant points
Image type	Binary, single blob
Type	Object measurement
Function	Find those two points that are furthest from each other
Switch images	Yes
Typical uses	Measuring the length/diameter of a binary object
Remarks	Also see *fwp*
Illustration	
Input image	Scissors
File number	48
Command	mdp

Left: original image

Right: after *mdp*



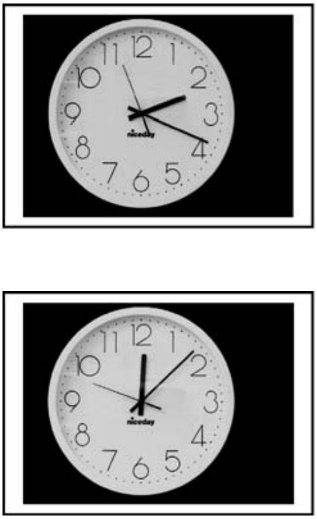
41.212 mic

See *lic*

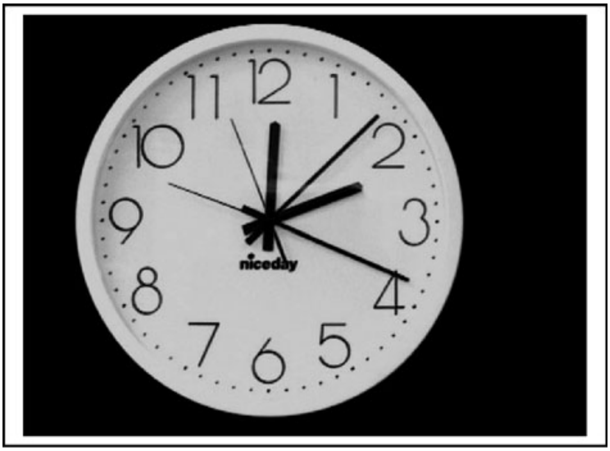
41.213 mni

Format	*mni*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Dyadic
Function	Find the minimum intensity between corresponding pixels in the current and alternate images
Definition	C ⇐ *MIN*(A, B)
Switch images	Yes
Typical uses	Using one image to mask another Segmenting an image by isolating an object from its background
Remarks	*mni* is a component of numerous algorithms
Illustration	
Input images	Clocks
File number	142 and 143
Command	*mni*

Left: two originals



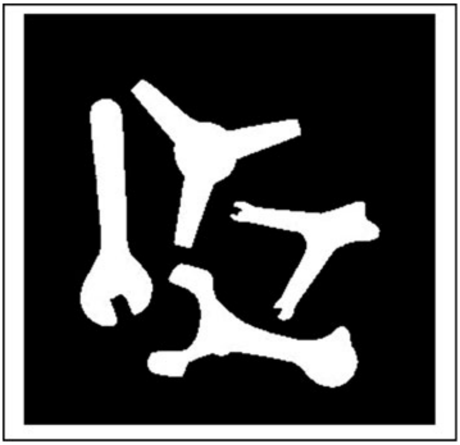
Right: after processing



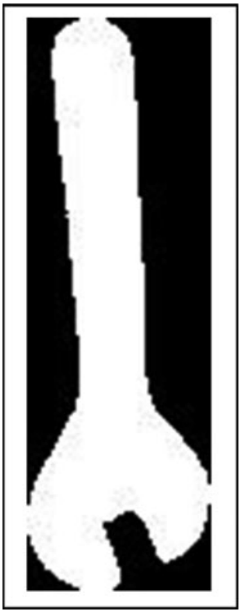
41.214 mrc

Format	*mrc*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob shape analysis
Switch	Yes
Function	Crop the image around the bounding box. (That is the minimum area rectangle aligned with the sides of the image that encloses all white pixels.) If there is more than one blob, the operation is performed on the first one found during a raster scan (searching top-to-bottom and left-to-right)
Switch images	Yes
Typical uses	Blob-shape analysis
Remarks	Notice that the image size changes as a result of applying *mrc*
Illustration	
Input image	Four metal parts
File number	46
Command	*mrc*

Left: original



Right: after processing



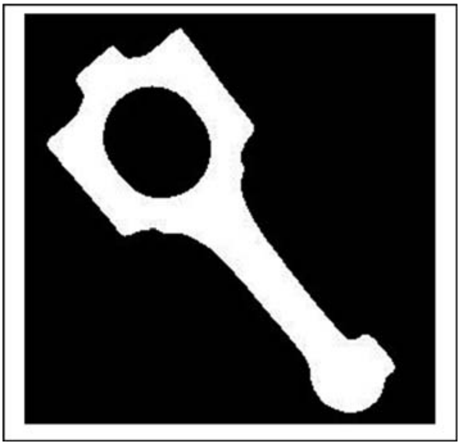
41.215 mst

Format	*mst*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob transform
Function	Thin each blob in a binary image; reduce it to a match-stick figure
Switch images	Yes
Typical uses	Blob shape analysis
Remarks	Thinning produces a connected figure that approximates the medial axis (skeleton, *ske*) but is subtly different from it

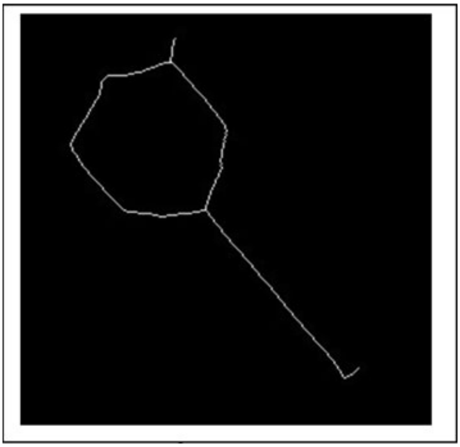
Illustration

Input image	Con-rod
File number	47
Command	*thr*, *mst* (N.B. *thr* avoids the artifacts created by storing a binary image in JPEG format)

Left: original



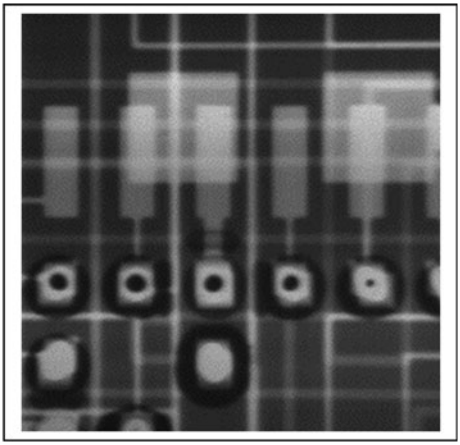
Right: after processing

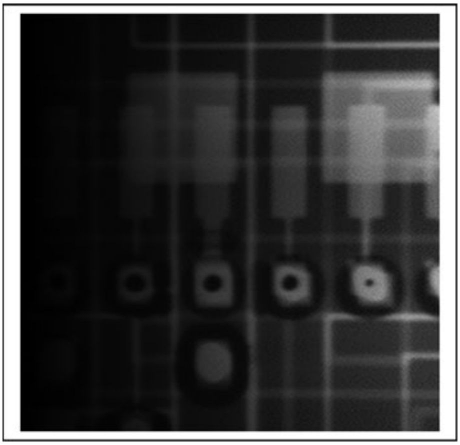


41.216 mul

Format	*mul*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Dyadic
Function	Multiply the intensities in the current and alternate images
Definition	C ⇐ (A × B)/255
Switch images	Yes
Typical uses	Calculating moments
Remarks	*mul* allows the contrast to be varied in different parts of an image
Illustration	
Input images	PCB and intensity wedge (*wgx*)
File number	135
Command	*adi*

Left: original (Other one is output of *wgx*) *Right*: after processing

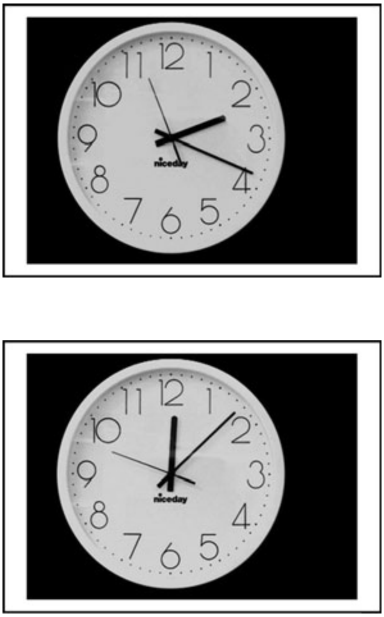




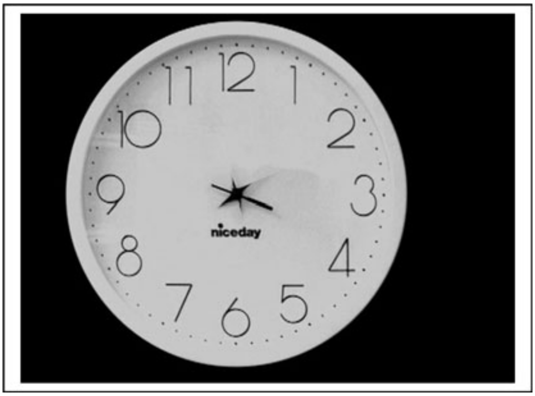
41.217 mxi

Format	*mxi*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Dyadic
Function	Find the maximum intensity between corresponding pixels in the current and alternate images
Definition	C ⇐ *MAX*(A, B)
Switch images	Yes
Typical uses	Superimposing (white) lettering onto an image Superimposing bright features of one image onto darker regions of the other
Remarks	*mxi* forms a component of numerous algorithms
Illustration	
Input image	Clocks
File number	142 and 143
Command	*mxi*

Left: two originals



Right: after processing



41.218 mxs

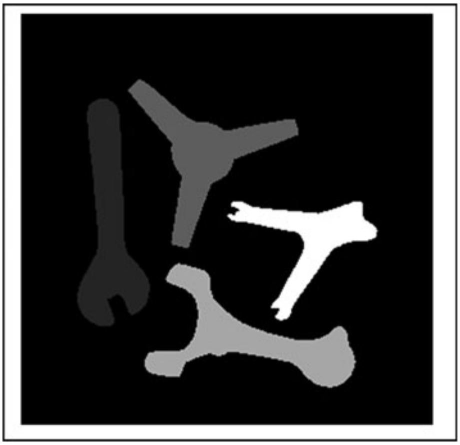
See *asi*

41.219 ndo

Format	*ndo*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob transform
Function	Numerate distinct objects in a binary image. All pixels within a given blob are set to the same intensity. Each blob is shaded with a different intensity value. Blobs are defined by 8-connectivity of white pixels
Switch images	Yes
Typical uses	Analysing binary images containing multiple blobs
Remarks	A 4-connected version could also be defined but this is not particularly useful as human beings often find 4-connectivity difficult to distinguish from 8-connectivity. Also see *big* and *kgr*
Illustration	
Input image	Metal objects
File number	46
Command	*thr*, *ndo*, *enc* (N.B. *thr* avoids the artifacts created by storing a binary image in JPEG format. enc was included to improve visibility. The result of applying *ndo* to this image is very dark, since there are only four blobs)

Left: original

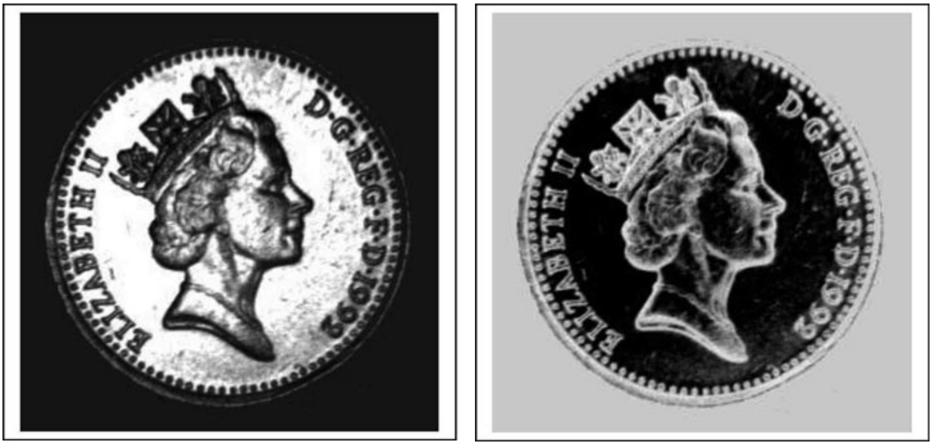
Right: after processing



41.220 neg

Format	*neg*
Arguments	none
Defaults	not applicable
Output values	none
Image type	grey
Type	monadic
Function	negate the input image
Definition	C ⇐ (W − A)
Switch images	yes
Typical uses	Preprocessing for commands that treat white and black differently (e.g., *rlc, blf, chu*)
Remarks	The function *neg* produces an image that resembles a photographic negative
Illustration	
Input image	Coin
File number	23
Command	*neg*

Left: original *Right*: after processing



41.221 news

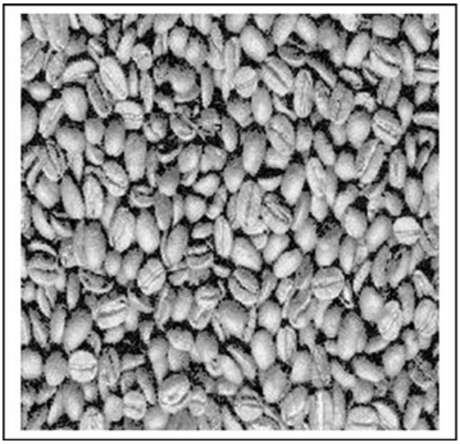
Format	*news*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Not relevant
Type	Information
Function	Visit the author’s web page providing the latest information about updates to QT
Switch images	No

41.222 nlg

Format	*nlg*
Arguments	None
Defaults	None
Output values	None
Image type	Grey
Type	Monadic
Function	Logarithm of intensities in the current image
Definition	$C \Leftarrow 255*log(A)/log(255)$ with hard limits placed at 0 and 255
Switch images	Yes
Typical uses	Used to improve contrast in dark regions of the image
Remarks	Also see *alg*, *sqr* and *sqt*
Illustration	
Input image	Coffee beans
File number	4
Command	*nlg*

Left: original

Right: after processing



41.223 nlp

See *dil*

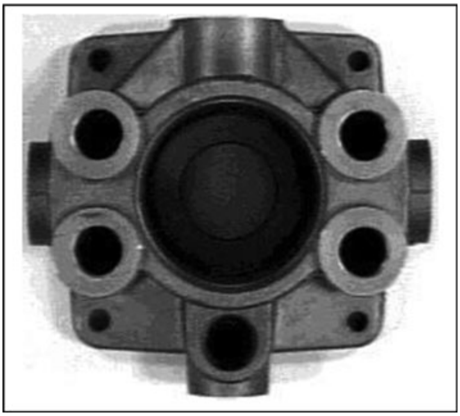
41.224 nop

Format	*nop*
Arguments	*None*
Defaults	Not applicable
Output values	None
Image type	Not relevant
Type	Dummy command
Function	Do nothing
Switch images	No
Typical uses	As a dummy command when calling functions such as *pci*

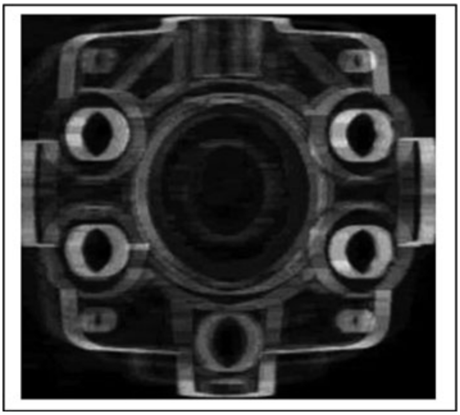
41.225 nrf

Format	*nrf*(*a*, *b*)	
Arguments	*a*	Width of the filter window
	b	Height of the filter window
Defaults	One argument supplied: b = a	
	No arguments supplied: a = b = 5	
Output values	None	
Image type	Grey	
Type	Monadic	
Function	The intensity values within each rectangular neighbourhood of size a × b are placed in rank order. The smallest value is then subtracted from the largest value	
Switch images	Yes	
Typical uses	Edge detection in images where there are no step-like intensity transitions	
Remarks	*nrf*(*a*, *b*) produces the same results as *rdf*(*a*, *b*, *1*)	
Illustration		
Input image	Hydraulics manifold	
File number	37	
Command	*nrf*(*11*, *3*)	

Left: original



Right: after processing

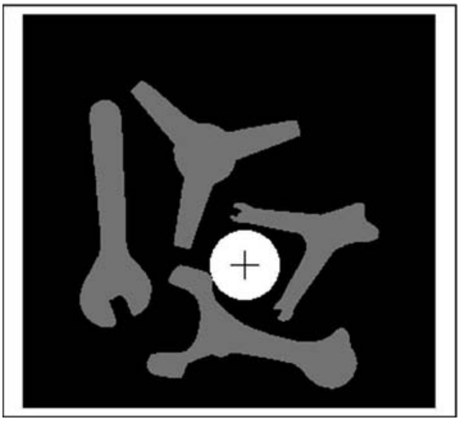


41.226 nwp

Format	[*x*, *y*, *r*] = *nwp*(*u*, *v*)	
Arguments	*u*	X-coordinate of a point
	v	Y-coordinate of a point
Defaults	None	
Output values	*x*	X-coordinate of the nearest white point
	y	Y-coordinate of the nearest white point
	r	Distance from the given point [*u*, *v*] to its closest white point, [*x*, *y*]
Image type	Binary	
Type	Binary image analysis	
Function	Draw the disc centred at [*x*, *y*] with radius *r*, where r is the distance from [*x*, *y*] to its closest white point	
Switch images	Yes	
Typical uses	Joining 'broken' contours (These are often produced when thresholding the output of an edge detector)	
Remarks	Also see *lic*, *mic*, *dic*, *fwp*	
Illustration		
Input images	Four piece parts	
File number	46	
Commands	*rni*(*46*), *thr*, [*x*, *y*] = *cur*; *nwp*(*x*, *y*), *adi*, *mxi*, *wri*, *zer*, *dcl*(*x*, *y*, *0*), *crs*(*10*), *neg*, *rei*, *mni*	

Left: original image

Right: after processing



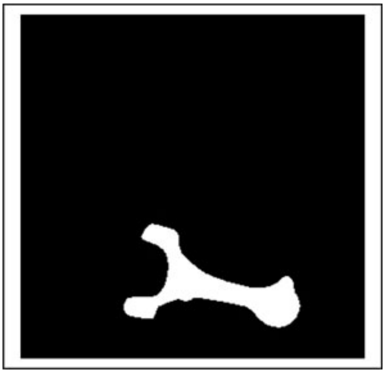
41.227 nxy/npo/nxy_all

Format	*nxy* *npo* *nxy_all*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Binary image transform
Function	*nxy*: Normalise position so that the centroid is relocated at the centre of the image *npo*: As *nxy*, plus normalisation of orientation (i.e., rotate the image so that the principal axis is horizontal) *nxy_all*: As *nxy* when there are multiple objects *npo_all*: As *npo* when there are multiple objects
Switch images	Yes
Typical uses	Analysing blob shape for a wide variety of applications in parts inspection and robot vision
Remarks	Imagine a visually guided robot whose task is to pick up flat objects lying on a table. The camera is directly overhead. Given three parameters (X- and Y-coordinates and orientation), the robot can pick up objects one a time. *npo* and *nxy* allow this process to be simulated. *nxy_all/npo_all* should be used when there is more than one binary object

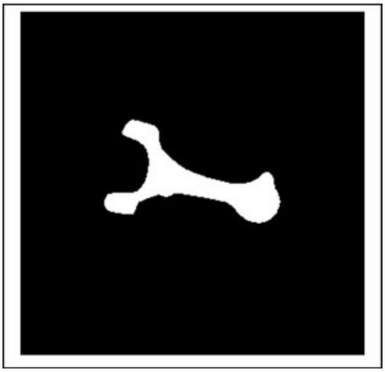
Illustration

Input image	Four metal objects with one item selected (con-rod)	
Image generation	*rni*(*46*), *thr*, *big*	
Command	npo	
	Removing blobs from only 2 sides:	*rbe*(*1*) or *rbe*(*1*, *8*)

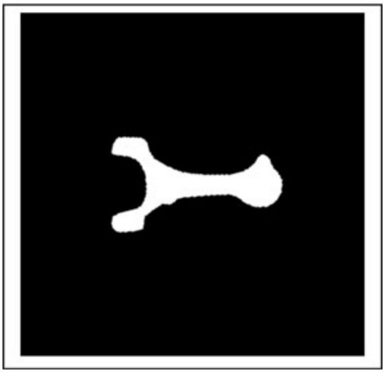
Top-left: original image



Top-right: after *nxy*



Bottom-left: after *npo*



41.228 oab

See *bpa*.

41.229 opn

See *dil*.

41.230 opr

Format	*opr*(*a*, *b*)	
Arguments	*a*	Operation to be performed
	b	Number of times operation is performed
Defaults	*b*	256
Output values	None	
Image type	Grey scale	
Type	Depends on the first argument (*a*)	
Function	Operation *a*(*i*) is performed for *i* = 1:*b*	
Switch images	Yes	
Typical uses	Training	
	Improving user understanding of image structure	
Remarks	The following examples shows how *opr* may be used	

Expanding spatial resolution

[p, q] = dgw; opr('rsz(i, i);', min(p, q))

Rotating the image

opr('tur(i)', 180)

Barrel shifting the image (Shift in X direction)

[p, q] = dgw; opr('psw(i, 0)', p)

Shifting the image (Shift in Y direction)

[p, q] = dgw; opr('psh(0, i)', q)

Variable gamma adjustment followed by pseudo-colour display

opr('gma(i/100), psc', 100)

Thresholding/Drawing intensity contours

caf, opr('thr(i-1), bed', 256)

Selecting single intensity level. Smoothing applied first

caf, opr('thr(i, i)', 256)

Intensity plot

[w, h] = dgw; opr('plt(i)', h)

Displaying blobs in order of size (biggest first)

n = cbl, opr('big(i), adi', n)

Display first N standard images

opr('rni(i)', N)

Bug tracing a binary edge

bed, n = cwp; seb, opr('thr(i, i)', n)

41.231 pad/sqi

Format *pad*(*a*, *b*, *c*)
sqi(*a*)

Arguments (*pad*) *a* Width of the left/right sides of the border
b Width of the top/bottom sides of the border
c Intensity of the padded border

Defaults *b* Equal to *a* when 1 or 2 arguments are supplied
c 0, when 1 argument is supplied

Output values None

Image type Grey or binary

Type Adjust image geometry

Function *pad* adds a border of intensity *c* and
width *a* to the left and right of the original image
width *b* to the top and bottom of the original image
sqi(*a*) automatically adjusts the aspect ratio so that the image is square. The padded area is assigned intensity value *a*

Switch images Yes

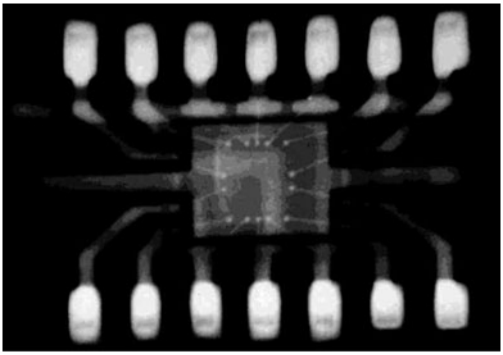
Typical uses *pad* Errors also occur if dyadic operators (*adi*, *sub*, *mxi*, *mni*, *exr*) are applied to two images of different sizes
sqi Some operation sequences, such as [*yxt*, *mxi*] *and* [*yxt*, *adi*], generate errors if the input image is not square

Remarks Also see *pex*, *psh* and *psw*

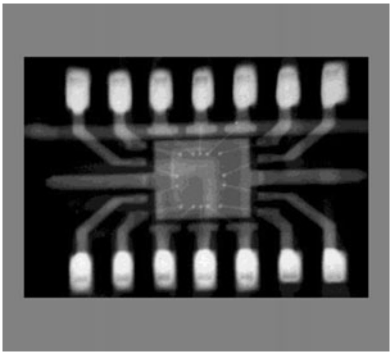
Illustration

Input image	X-ray of a DIL integrated circuit	
File number	113	
Command	*pad*(*20*, *50*, *128*)	(*Top-right*)
	sqi(*128*)	(*bottom-left*)

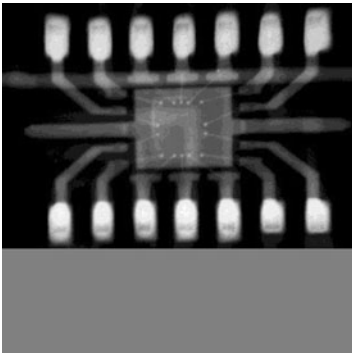
Top-left: original



Top-right: after *pad*(*20*, *50*, *128*)



Bottom-left: after *sqi*(*128*)



41.232 pbi

Format *[x, y, z1, z2] = pbi(x1, y1, x2, y2)*

Arguments *x1* X-coordinate of point 1

y1 Y-coordinate of point 1

x2 X-coordinate of point 2

y2 Y-coordinate of point 2

Defaults None

Output values *x* X-coordinate of the point mid-way between *[x1, y1]* and *[x2, y2]*

y Y-coordinate of the point mid-way between *[x1, y1]* and *[x2, y2]*

z1 Orientation of the line joining [x1, y1] and [x2, y2]

z2 Orientation of the perpendicular bisector of the line joining *[x1, y1]* and *[x2, y2]*

Image type Any

Function Calculate a point *[x, y]* on the perpendicular bisector of the line joining *[x1, y1]* and *[x2, y2]* and draw the perpendicular bisector

Switch images Yes

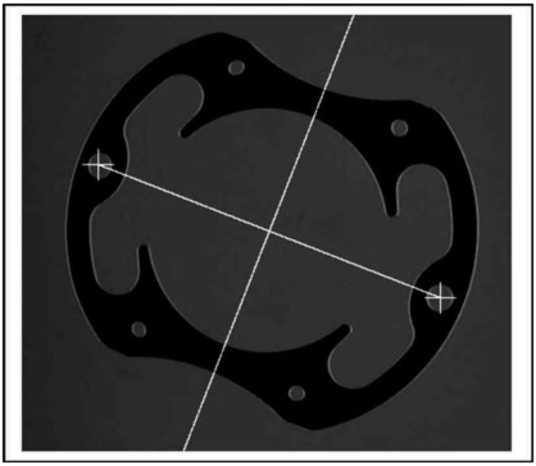
Typical uses Analysing binary images

Illustration

Input image Stator stamping for an electric motor

File number 174

Image The user defined the two points indicated by crosses with the cursor. Their coordinates were then passed to *pbi*.

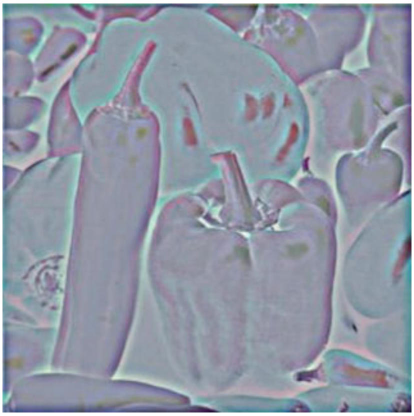


41.233 pcd/adc/mnc/mxc/suc

Format	*pcd*(*x*) *adc* *mnc* *mxc* *sue*	
Arguments	x	Dyadic operator (*pcd* only)
Defaults	None	
Output values	None	
Image type	Colour	
Type	Colour image processing	
Function	*pcd*(x)	Perform a given dyadic operator (x) to two colour images
	adc	Add two images (point-by-point, RGB channels are processed separately)
	mnc	Minimum of intensities in two images (point-by-point, RGB channels are processed separately)
mxc	mxc	Maximum of intensities in two images (point-by-point, RGB channels are processed separately)
	sue	Subtract two images (point-by-point, RGB channels are processed separately)
Switch images	Yes	
Typical uses	Filtering colour images	
Remarks	Also see *pci*, *adi*, *mxi*, *mni*, *sub*	
Illustration		
Input images	Peppers	
File number	73	
Command	*pci*(‘*caf*’), *pcd*(‘*mni*’) (High pass filter, applied to the RGB channels separately)	

Top-left: original

Top-right: after processing



41.234 pcf

Format	*pcf*
Arguments	None
Defaults	Not applicable
Output values	None
Image types	Current, colour; alternate, 255 × 255-pixel grey-scale image
Type	Colour image processing
Switch	Yes
Function	Programmable colour filter

Definition The alternate image is used as a look-up table, to transform the current image (colour) into a monochrome image in which colour sets are mapped into distinct intensity levels. Thus, a broad set of colours that a person might call 'yellow' might be mapped to level 137, while 'turquoise' is mapped to level 119 and 'mauve' to 93. The assignment can be selected arbitrarily. To understand how colour recognition can be achieved by *pcf*, consider the following. Let $A(p, q)$ denote the intensity at the pixel (p, q) in the alternate image. Also, let pixel (i, j) in the current image have hue $H(i, j)$ and saturation $S(i, j)$. These values are calculated from the RGB values generated by *pcf*. (The intensity component is not considered, since this does alter the identification of colours in the context of Machine Vision.) The pixel at address (i, j) within the output image generated by *pcf* is assigned an intensity value equal to

$$A(H(i, j), S(i, j))$$

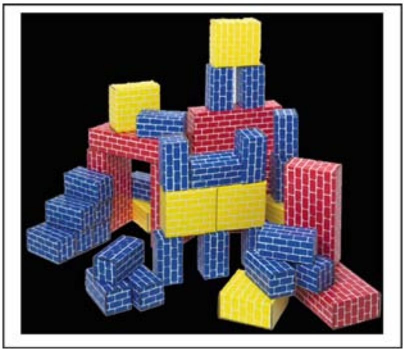
Typical uses Colour recognition; reduction of colour images to grey-scale for simplified processing

Remarks A programmable colour filter is best applied to images containing 'block colours'. These are discrete, well separated, colours that do not vary significantly over a large area and are very different from each other. (See illustration below.) Normally, the boundaries between block colour regions are typically quite sharp, with little or no blending from one block colour region to another. Block colour printing is unable to represent real scenes but is cheap to produce. As a result, many containers/labels, for food products, household cleaners, toiletries, etc. are printed using block colours. In the first illustration below, the alternate image, which defines the colour look-up table, is computed from the hue-saturation scattergram. (*shs*) Notice that low values of saturation (corresponding to the left-hand side of the image) are mapped to black. In the second example, the look-up table is a simple pattern based on an intensity wedge. Again, low values of saturation are mapped to black. Another example, showing how *shs* and *pcf* can be used together is given in the notes relating to *shs*

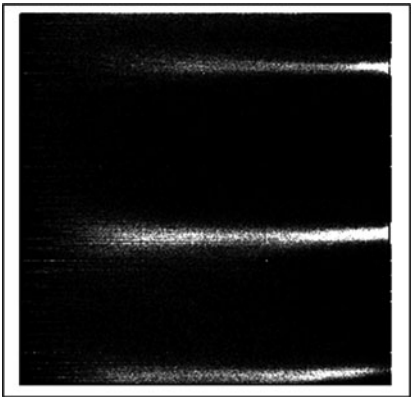
Illustration

Input image	Coloured blocks
File number	168
Command	See below

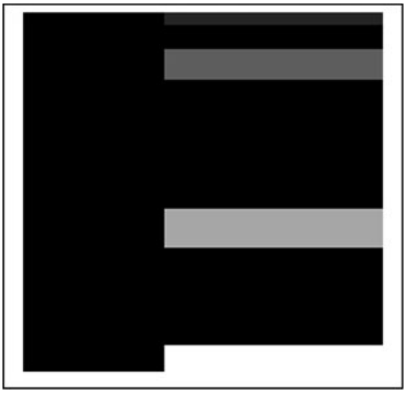
Top-left: original



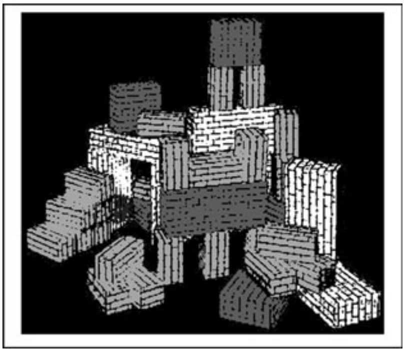
Top-right: after *shs*



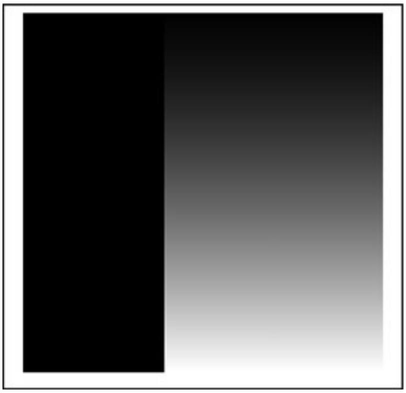
Centre-left: look-up table generated using shs, *rin*, *csh*, *thr*(*32*), *ndo enc*, *dil*(*9*). *psh*(*100*, *0*)



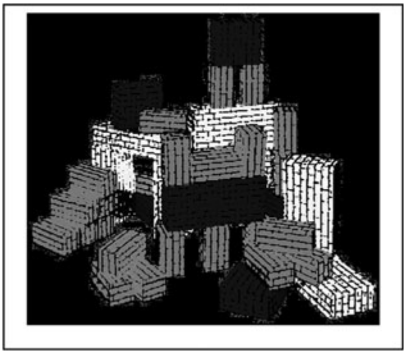
Centre-right: *pcf* using the look-up table defined by the 'bars' image to the left



Bottom-left: look-up table generated using *wgy*, *psh*(*100*)



Bottom-right: *pcf* using this look-up table (Each hue is assigned a to a different intensity value)



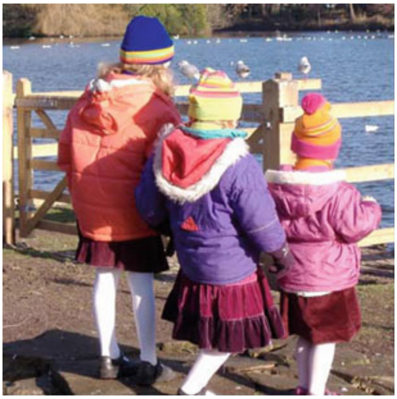
41.235 pci

Format	*pci*(*r*, *g*, *b*)
Arguments	*r* Image processing command/sequence applied to the R-channel *g* Image processing command/sequence applied to the G-channel *b* Image processing command/sequence applied to the B-channel
Defaults	*g* Same value as *r* *b* Same value as *r*
Output values	None
Image type	Colour
Type	Progam control
Function	Apply the following commands/sequences of commands *r* applied to the R-channel *g* applied to the G-channel *b* applied to the B-channel
Switch images	Yes
Typical uses	Image enhancement for visual display, or preprocessing prior to colour recognition
Remarks	The commands/sequences of commands can be chosen independently. Commands that return outputs cannot be used here (Use *pca* instead)

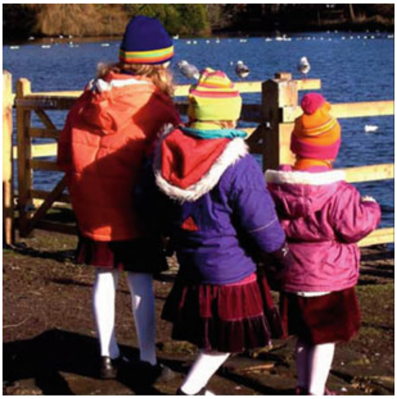
Illustration

Input image	Children
File number	264
Command	See below

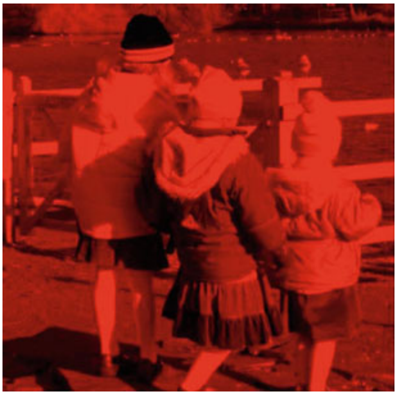
Top-left: Original



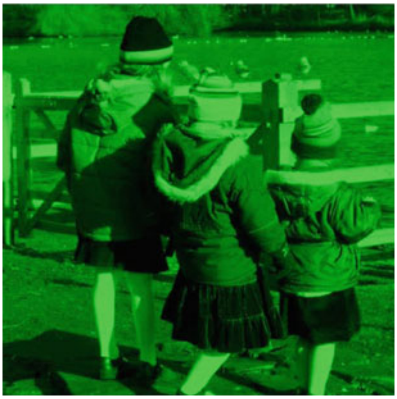
Top-right: *pci(‘sqr’)*



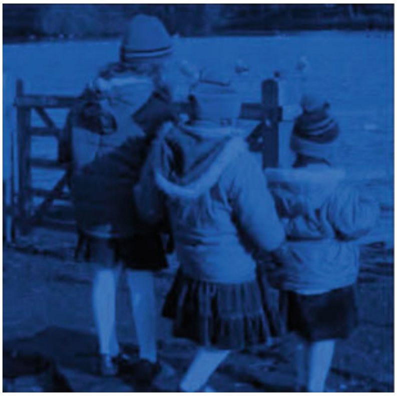
Centre-left: *pci(‘nop’,‘zer’,‘zer’) (This retains only the red channel)*



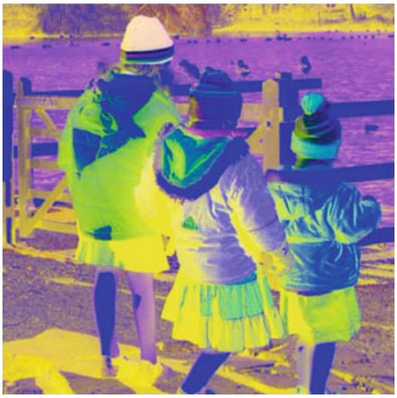
Centre-right: *pci(‘zer’,‘nop’,‘zer’) (This retains only the red channel)*



Bottom-left: *pci(‘zer’,‘zer’,‘nop’)*



Bottom-right: *pci(‘neg’,‘neg, sqr’,‘heq’)*



41.236 ped

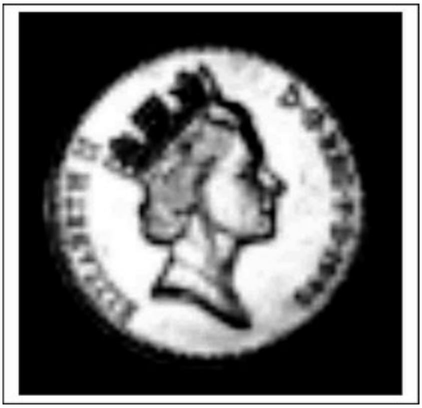
See *ced.*

41.237 pex

Format	*pex*(*x*)	
Arguments	*x*	
Defaults	*x*	2
Output values	None	
Image type	Grey	
Type	Image warping	
Function	Rescale the spatial resolution along both axes, by a given amount, *x*	
Switch images	Yes	
Typical uses	Demonstrates the effects of adjusting the spatial resolution	
Remarks	The input parameter, *x*, may be greater than 1, in which case the image resolution is reduced. If $x < 1$, the image resolution is increased, using bilinear interpolation. It is interesting to squint when viewing a reduced resolution image. This has the effect (similar to the operator *caf*) of blurring edges and may actually make it more readily identifiable visually. Also see *rsz*	
Illustration		
Input image	Coin	
File number	23	
Command	*pex*(4)	

Left: original

Right: after processing



41.238 pft

Format $p = pft(a, b, m, n)$

Arguments a Row number indicating start of scan

b Row number indicating end of scan

m Number of scan lines

n Polynomial order

Defaults None

Output values p Vector containing the coefficients of the fitted polynomial

Image type Binary

Type Image analysis

Switch No

Function Fit a polynomial curve of order n to a set of edge points, found by scanning from left to right along a series of m horizontal lines. The latter are equally spaced between rows a and b

Switch images No. A curve is drawn in Figure 6

Typical uses Defining circularly symmetric filters

Remarks Choosing an appropriate value for the polynomial order (parameter n) is crucial. In the example given below, n is too small and does not allow an accurate fit of the edge points. (See right-hand-side of the graph.) However, raising n can cause problems, due to over-fitting. As a general guide, n should be as low as possible, while ensuring a good fit. Normally, the number of data points, m, should be greater than $5n$. Furthermore, the edge should have a finite slope and be single-valued for every value of the X-coordinate. If necessary, polynomial curves should be fitted to small segments of a complicated curve

Illustration

Input image Bottle, with thresholding to create a binary image

Image generation *rni*(*31*), *thr*, *neg*

Command *pft*(*80*, *275*, *20*, *3*)

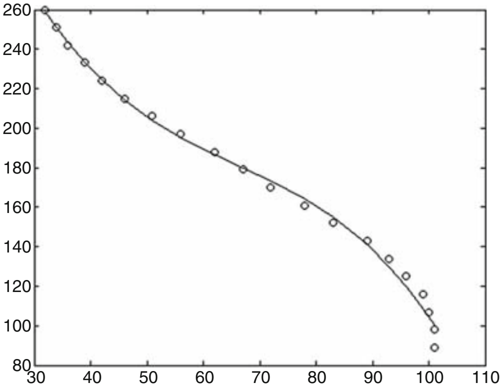
Output: $p = -0.0176\ 1.4281\ -30.8175\ 347.2487$

This defines the following polynomial:

$$y = -0.0176x^3 + 1.4281x^2 - 30.8175x + 347.2487$$

Left: original image. The vertical limits of the scanning process, performed by *hzs*, are indicated by crosses

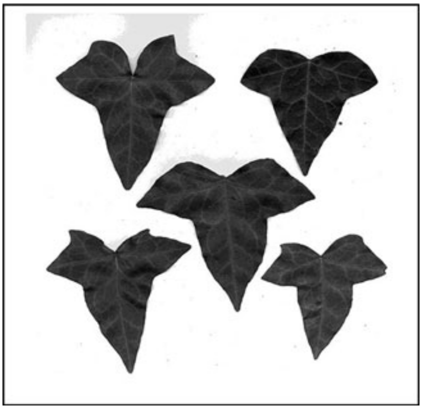
Right: pivotal points found by *hzs* (circles). The curve is a 3rd-order polynomial, which slightly under-fits this data. Notice that the Y axis is inverted



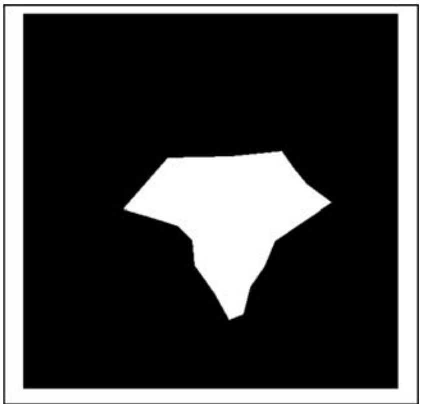
41.239 pgn

Format	*pgn*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey-scale or binary
Type	Drawing
Function	Draw a filled polygonal interactively, using the cursor
Switch images	Yes
Typical uses	Selecting specific bodies or regions of interest, to improve speed of processing. Selecting sub-images for use as morphology structuring elements of convolution kernels
Remarks	The user places the cursor at each of the nodes of the polygon in turn and then presses RETURN to finish. The result is a binary image, which could be used as a mask to isolate an object of special interest (e.g., the central leaf in the illustration below)
Illustration	
Input image	Ivy leaves
File number	92
Command	*pgn*

Left: original image



Right: polygonal region defined by the author using *pgn*. Use *mni* to use this as a mask to isolate the largest leaf

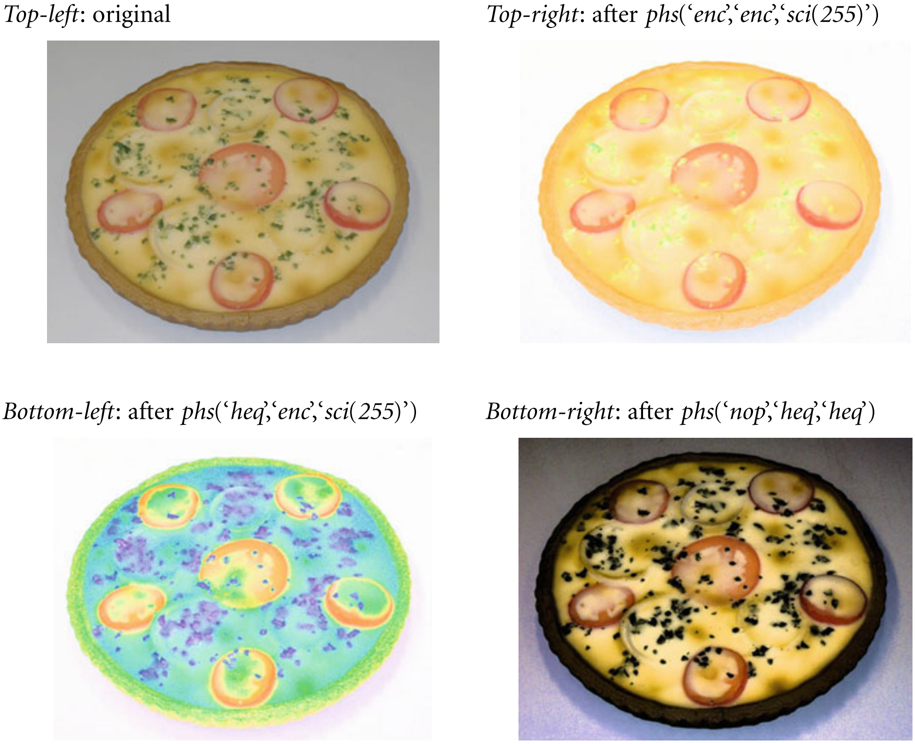


41.240 pgt

Format	$c = pgt(x, y)$	
Arguments	x	X-coordinate of the given point
	y	X-coordinate of the given point
Defaults	None	
Output values	c	Intensity at the given point
Image type	Grey or binary	
Type	Image measurement	
Switch	No	
Function	Find the intensity at a point given its XY-coordinates	
Typical uses	Understanding image structure	
Remarks	Also see *cur*, *fap*, *vpl*	

41.241 phs

Format	*phs*(*a*, *b*, *c*)	
Arguments	*a*	Processing sequence for the hue (H) channel
	b	Processing sequence for the saturation (H) channel
	c	Processing sequence for the intensity (I) channel
Defaults	*a*	‘*nop*’ (no operation)
	b	‘*sci*(*255*)’ (*set to maximum value*)
	c	‘*sci*(*255*)’ (*set to maximum value*)
Output values	None	
Image type	Colour	
Type	Colour image processing	
Function	Operation *a* is applied to the H channel Operation *b* is applied to the S channel Operation *c* is applied to the I channel	
Switch images	Yes	
Typical uses	Enhancing colour contrast to improve visibility or processing	
Remarks	Also see *pci*	
Illustration		
Input image	plastic replica of a pizza	
File number	219	
Command	*phs*(‘*enc*’,‘*enc*’,‘*sci*(*255*)’) *phs*(‘*heq*’,‘*enc*’,‘*sci*(*255*)’) *phs*(‘*nop*’,‘*heq*’,‘*heq*’)	



Top-left: original

Top-right: after *phs*(‘*enc*’,‘*enc*’,‘*sci*(*255*)’)

Bottom-left: after *phs*(‘*heq*’,‘*enc*’,‘*sci*(*255*)’)

Bottom-right: after *phs*(‘*nop*’,‘*heq*’,‘*heq*’)

41.242 pis

Format	*pis*(*a*, *b*, *c*)	
Arguments	*a*	Folder for input images
	b	Operation to be performed on those images
	c	Folder for output images
Defaults	*c*	QT_work_files
Output values	None	
Image type	Any	
Type	Processing image sequences	
Function	Perform operation *b* on all images in folder *a*, leaving the results as numbered images in folder *c*	
Switch images	Depends on the command sequence *b*. Normally, both images are overwritten	
Typical uses	Processing image sequences	
Remarks	Example: pis(qt_images_path, ‘enc,thr,bed’,qt_work_path)	

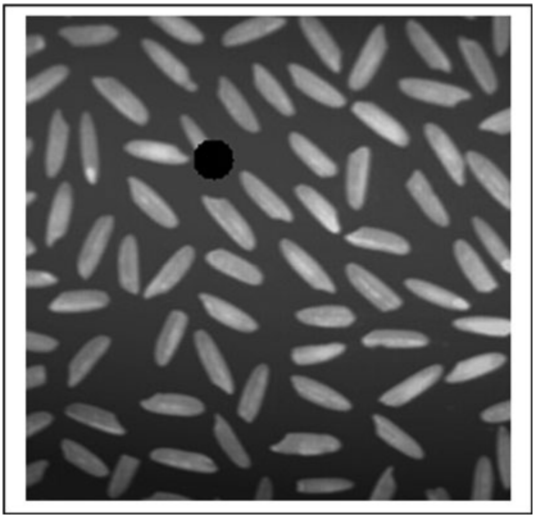
41.243 pki

Format	[*p*, *q*, *s*] = *pki*
Arguments	None
Defaults	Not applicable
Output values	*p* X-coordinate of the first point found with the peak intensity
	q Y-coordinate of the first point found with the peak intensity
	s Value of the peak intensity
Image type	Grey
Type	Image analysis
Switch	Yes
Function	Locate the point (*p*, *q*) with the peak intensity (*s*). If there is more than one such point, the coordinates and value of the left-top-most one are returned. The point (*p*, *q*) is plotted as a single white pixel in the current image, which is otherwise black
Typical uses	Analysis of Radon transform image. (See *rdn.*) Suppressing highlights due to glinting
Remarks	The point of maximum intensity and the area immediately around it (i.e., pixels, within a distance *r*) can be masked out by using the following command sequence: *wri*, [*a*, *b*, *c*] = *pki*; *zer*, *cir*(*a*, *b*, *r*), *blf*, *neg*, *rei*, *mni*

Illustration

Input image	Rice grains
File number	3
Commands	*wri*, [*a*, *b*, *c*] = *pki*; *zer*, *cir*(*a*, *b*, *10*), *blf*, *neg*, *rei*, *mni*

Below: point of peak intensity and the pixels surrounding pixels suppressed



41.244 plot3d

Format *plot3d*(*a*, *b*)

Arguments *a* Adjust the resolution of the plot along the X-axis (Large values reduce the resolution)

b Adjust the resolution of the plot along the Y-axis (Large values reduce the resolution)

Defaults *a* 1

b 1

Output values None

Image type Grey

Switch images No

Type Understanding image structure

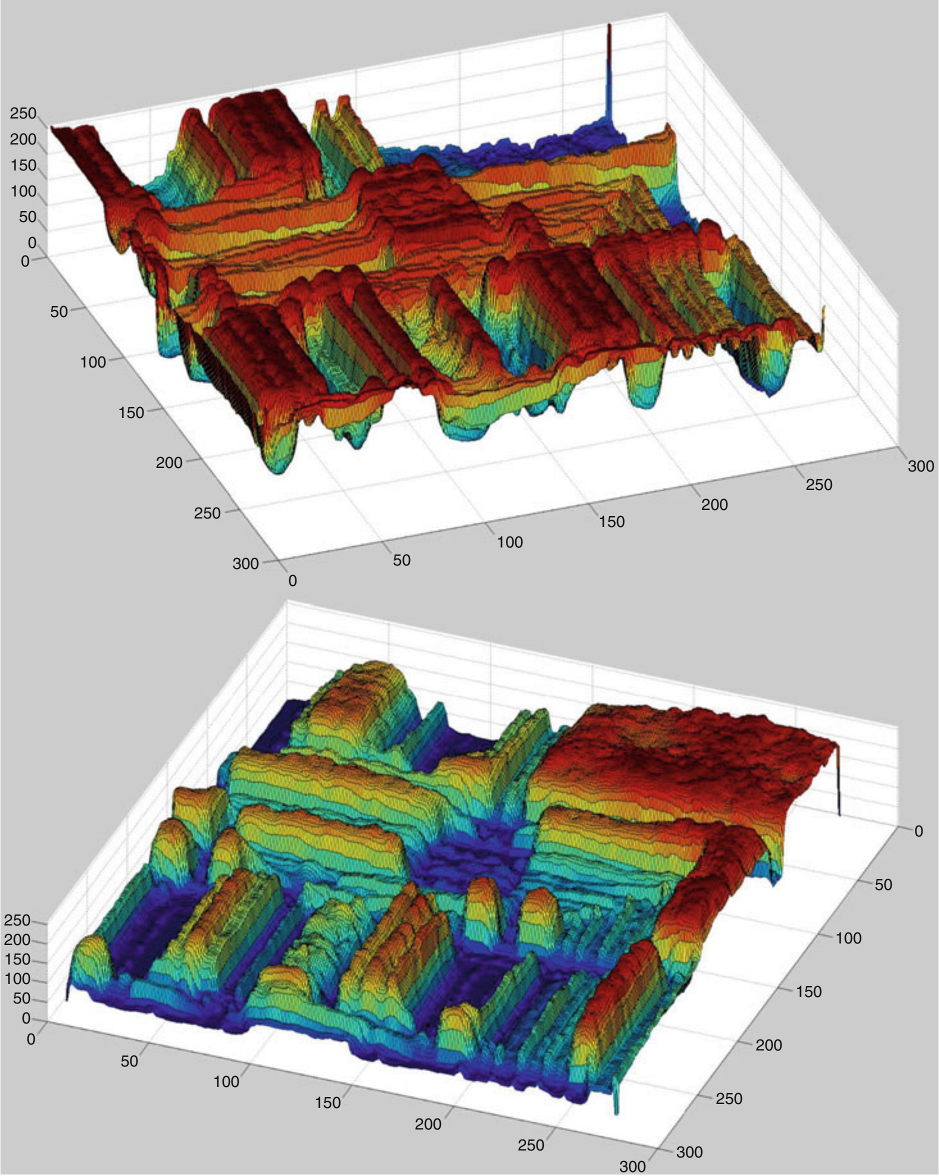
Function Display the current image as a 3-dimensional plot with a wire-mesh overlay and in pseudo-colour. The user can adjust the viewing point interactively with the mouse. (Click on the '*Rotate 3D*' icon first.) This creates the effect of flying over a wrinkled and coloured carpet, whose height is proportional to the image intensity. To exit, the user selects MATLAB's '*Rotate 3D*' icon and clicks on the image pane

Remarks It is often useful to apply image smoothing (*caf*, *raf*, or *mdf*) and/or enhancement (*enc*, *heq*, *sqr*, *sqt*, *nlg*, *alg*) first. Negation *(neg)* is also useful

Illustration

Input image	Circuit
File number	136

Top:	*rni(136), mdf(5), enc, plot3d(1, 1)*	The author selected the viewing point interactively
Bottom:	*rni(136), mdf(5), enc, neg, plot3d(1, 1)*	The author selected the viewing point interactively



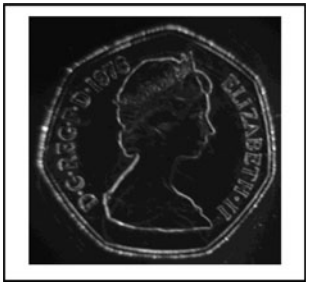
41.245 plt

Format	*plt*(*a*)	
Arguments	*a*	Height of sampling line for profile
Defaults	*a*	Half image height
Output values	None	
Image type	Grey	
Type	Measurement and anlysis	
Function	Plot the intensity profile along a defined row in the current image. The plot is drawn in a new figure	
Switch images	No	
Typical uses	*plt* is used principally as an aid to understanding the structure of an image and in guiding algorithm design	

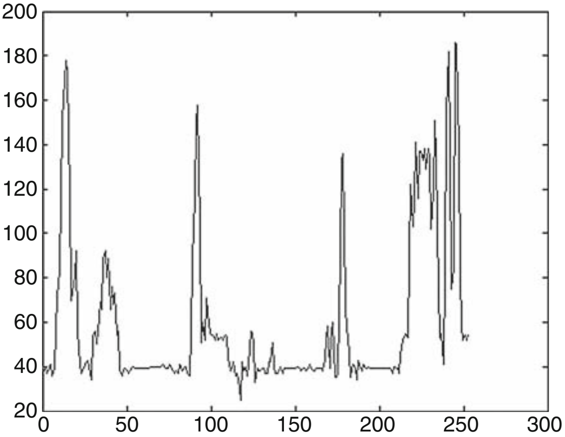
Illustration

Input image	Coin
File number	24
Command	*plt*

Left: input image



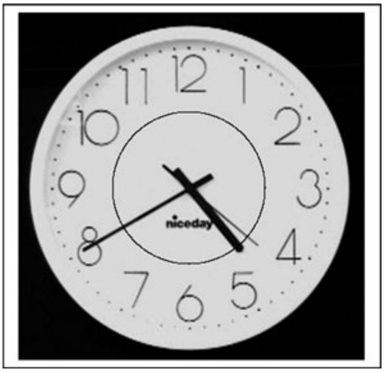
Right: intensity profile
Abscissa, horizontal position
Ordinate, intensity



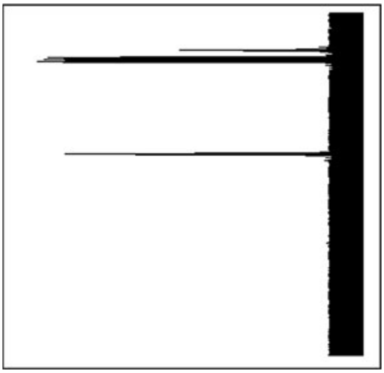
41.246 plv

Format	*plv*(*a*)
Arguments	a Vector to be plotted
Defaults	None
Output values	None
Image type	Grey
Type	Drawing
Function	Plot the given vector in the current image
Switch images	Yes
Typical uses	Used in conjunction with *lsc* or *scc*, *plv* is able to plot intensity profiles along a line or around a circular path
Remarks	*plt* and *ipl* plot intensity profiles in a new figure, whereas *plv* draws a graph in the current image. As a result, the output of *plv* can be processed like any other binary image. Also see *plt*, *lsc* and *scc*
Illustration	
Input image	Clock
File number	148
Command	*a* = *scc*(*288*, *247*, *85*); *plv*(*a*)

Left: input image For the purposes of illustration, the sampling circle used by *scc* has been superimposed and the image has been cropped. Notice that this circle intersects the second-, hour- and minute-hands and that it traverses the image in a clockwise direction, beginning at the 3 o'clock position



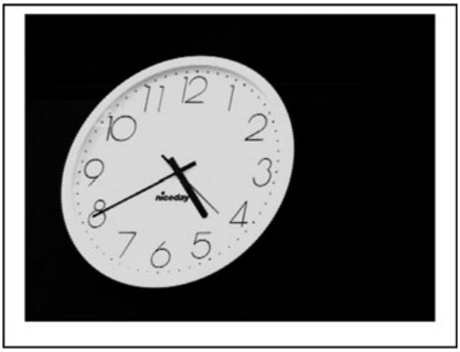
Right: Intensity plot. *Horizontal axis*: intensity. *Vertical axis*: angular position. Going from top to bottom, the three intensity 'peaks' correspond to the second-, hour- and minute-hands, respectively. Notice that this is an image and hence can be processed as normal using QT operators



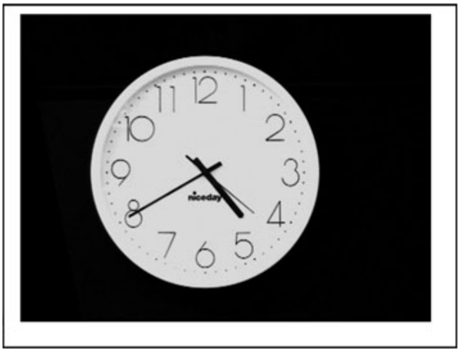
41.247 plw

Format	*plw*(*a*)
Arguments	*a* Warping parameter
Defaults	None
Output values	None
Image type	Grey scale or binary
Type	Image warping
Switch	Yes
Function	Parallelogram warping; the rows of the image are shifted sideways by an amount that is proportional to the vertical position
Switch images	Yes
Typical use	Correcting image distortion caused by viewing from an oblique angle
Illustration	
Input image	Clock, with parallelogram warping deliberately introduced (*plw*(*0.2*))
File number	148
Command	*plw*(−*0.2*)

Left: original



Right: after processing



41.248 pmf

Format *pmf(a, b, c, d)*

Arguments *a* *Start frame number*
b *End frame number*
c *Function to be applied*
d *URL of the AVI movie file (This must be RGB, non-compressed format)*

Default values *a* *1*
b *10*
c *neg*
d *QT_images/movie sample3.Avi*

Output values None

Output images The output images are saved in files
movie_frame1, movie_frame2, movie_frame2, etc.
in folder
/QT_image_sequences/

Image type Defined by the movie

Function Process a non-compressed AVI movie file with a given function

Switch images Not relevant as both the Current and Alternate images are destroyed

Typical uses Analysing AVI movies, or separating such movies into individual frames

Remarks The operator *pmf* is likely to be very slow as it processes many images. Examples of use

1. Convert to grey and negate each frame
   c = ‘*gry, neg*’
2. Apply edge detector (*sed*) to each colour channel
   x = ‘*sed, enc, neg*’, *c* = [‘*pci*(‘*char*(*39*), *x*, *char*(*39*),’)’]
3. Split the movie into separate frames for processing later
   c = ” (*empty string*)

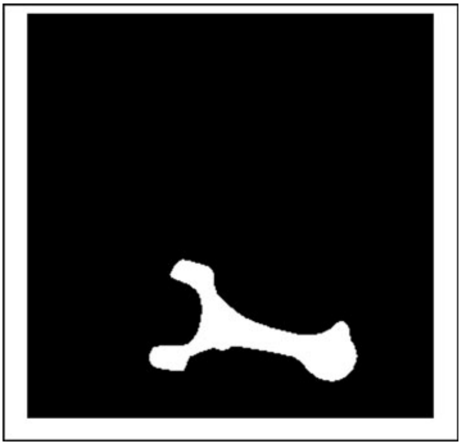
Use pis to process a set of images.

Use psf to process a series of images and then construct a movie file.

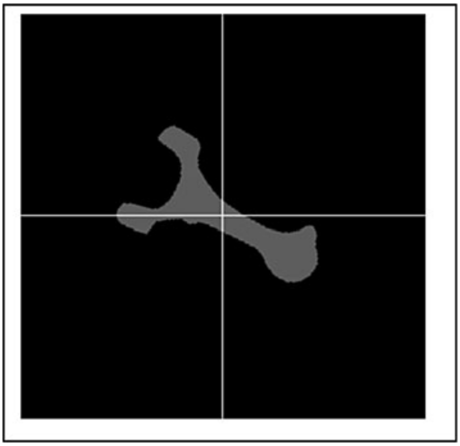
41.249 pon

Format	[*x*, *y*, *t*] = *pon*	
Arguments	None	
Output values	*x*	X-coordinate of the centroid
	y	Y-coordinate of the centroid
	t	Orientation
Image type	Binary, multiple blobs	
Type	Binary image transform	
Function	Find the position of the centroid of all white pixels (call this point A) and B, the white pixel furthest from A. Normalise the position, so that the centroid is at the centre of the image and then rotate it, so that the line AB is horizontal	
Switch images	Yes	
Typical uses	Analysing blob shape for a wide variety of applications in parts inspection and robot vision	
Remarks	*pon* is an alternative to *npo* and *npo_all*. Also see *rdc*	
Illustration		
Image	Four metal objects with one item selected (con-rod)	
Image generation	*rni*(46), *thr*, *big*	
Command	*pon*, *hin*, *dci* (The horizontal line through the centre of the image intersects the edge of the blob at the point that is furthest from the centre)	

Left: original



Right: after processing



41.250 pop

Format	*pop*
Arguments	None
Output values	None
Image type	Any
Type	Stack operator
Function	Pop the image stack. The top image on the stack is copied into the current image and is then deleted from the stack
Switch images	Yes
Remarks	Also see *push*, *ssk*, *rsk*, *csk*

41.251 pre

Format *[a, b] = pre*

Arguments None

Output values *a* X-coordinate of the point selected by the user

b Y-coordinate of the point selected by the user

Image type Any

Function The user delects a point, [*a*, *b*], with the cursor. This is then used to define the parameters for plotting the row and column intensity profiles

Switch images No

Typical uses The intensity plots are displayed as overlays on the current image, which is not altered. Any further processing deletes them. The alternate image also remains unaltered

Remarks Initial analysis of grey-scale images

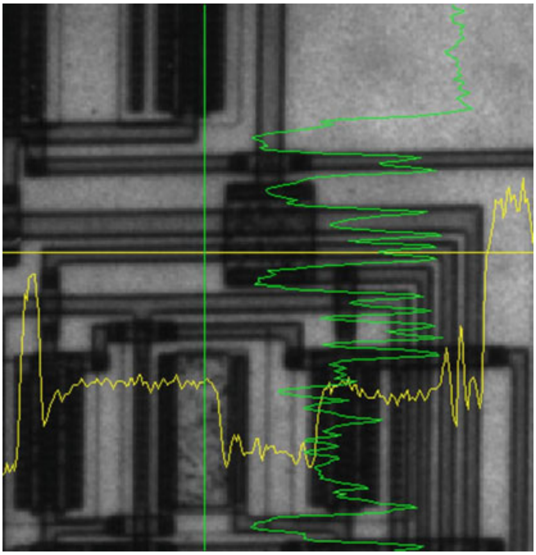
Also see *cur*, *plt*

Illustration

Input image Electronic circuit on ceramic substrate

File number 136

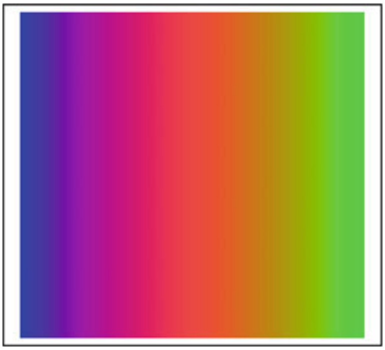
Command *prc* (The user selects a point to define which row and column profiles are to be plotted)



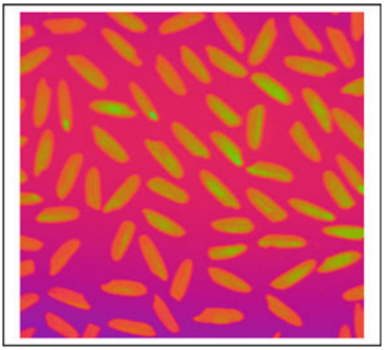
41.252 psc/rpc

Format	*psc* *rpc*	
Arguments	None	
Defaults	Not applicable	
Output values	None	
Image type	Grey scale	
Type	Image display	
Function	*psc*	Display the image using a pseudo-colour mapping based on gamma adjustment. The original grey-scale image is also displayed
	rpc	Display the image using a pseudo-colour mapping that is reversible (using the function *ity*). The original grey-scale image is also displayed
Switch images	Yes	
Typical uses	Improving user understanding of image structure	
Remarks	The human eye is more sensitive to changes in colour than intensity. Low-contrast intensity variations can often be seen more easily in a pseudo-colour image. The results of the pseudo-colour mapping can be processed in as any colour image. The function *rpc* produces a mapping that approximately preserves intensity. This means that the sequence [*rpc*, *ity*] produces a grey-scale image that is the same as the original	
Illustration		
Input image	Wedge generated by *wgx* Grains of rice (file number 3)	
Commands	*psc* *rpc*	

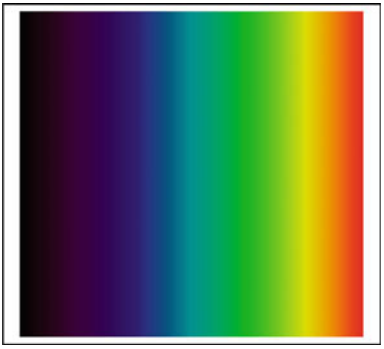
Top-left: *psc* applied to an intensity wedge (*wgx*)



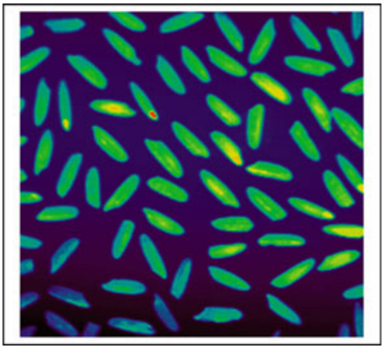
Top-right: *psc applied to the rice image*



Bottom-left: *rpc* applied to an intensity wedge (wgx)



Bottom-right: *rpc* applied to a the rice image



41.253 psf

Format	*psf* (*a*, *b*)
Arguments	*a*
Default	*a* = '*/QT_image_sequences/*'
Output values	None
Output movie	An AVI movie called '*movie_output.avi*' is stored in the folder '*QT_work_files*'
Image type	Defined by the input movie
Function	Process all of the images in folder *a* using function *b*
Switch images	Both Current and Alternate images are retained unchanged
Typical uses	Enhancing or annotating individual frames of a movie. For example, *psf* might be used to demonstrate how a moving object can be tracked
Remarks	This operators can be very slow as many images may have to be processed. For software to split MPEG files into individual frames visit *http://www.flydragonsoft.com/avi-decompressor.htm* Alternatively, use *pmf* with its second argument equal to the empty string (")

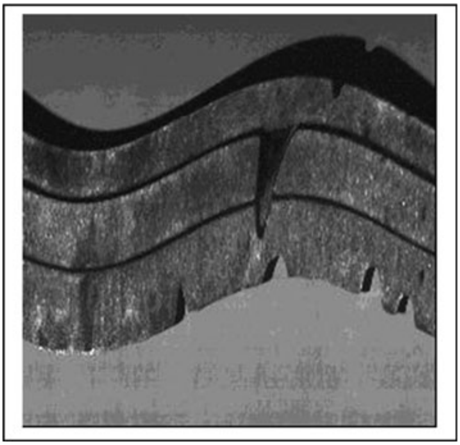
41.254 psh

Format	*psh*(*a*, *b*)
Arguments	*a* Horizontal shift *b* Vertical shift
Defaults	*b* 0
Output values	None
Image type	Grey scale or binary
Type	Affine
Switch	Yes
Function	Picture shift, without wrap-around
Definition	$C(i + a, j + b) \Leftarrow A(i, j)$
Switch images	Yes
Remarks	Those areas of the output image that are not covered by the shifted input picture are set to black. Also see *psw*

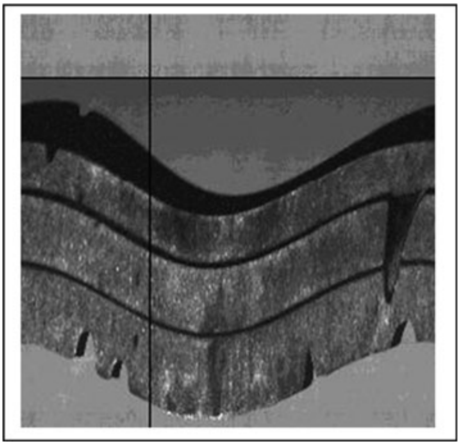
41.255 psw

Format	*psw*(*a*, *b*)
Arguments	*a* Horizontal shift
	b Vertical shift
Defaults	*b* 0
Output values	None
Image type	Grey scale or binary
Type	Affine
Switch	Yes
Function	Picture shift, with wrap-around
Switch images	Yes
Remarks	Also see *psh*
Illustration	
Input image	Line scan image of a circular component on a turn-table
File number	99
Command	*psh*(*80*, *40*)

Left: original



Right: after processing



41.256 ptc

Format *ptc*(*a*, *b*)

Arguments *a* X-coordinate for the pivotal point for the axis transformation

b Y-coordinate for the pivotal point for the axis transformation

Defaults *a* X-coordinate of the centre of the image

b Y-coordinate of the centre of the image

Output values None

Type Image warping

Switch Yes

Function Polar-to-Cartesian coordinate axis transformation

Switch images Yes

Typical uses Inspecting objects that have circular, spiral, or radial ('wheel spoke') features

Remarks Concentric circular features centred on the pivotal point (*a*, *b*) are mapped to parallel vertical lines, while radial lines are mapped to parallel horizontal lines. An Archimedean spiral (equation r = k θ) is mapped to an inclined straight line. Also see *ctp*

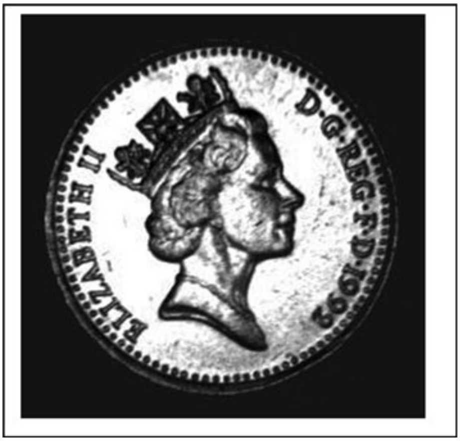
Illustration

Input image Coin

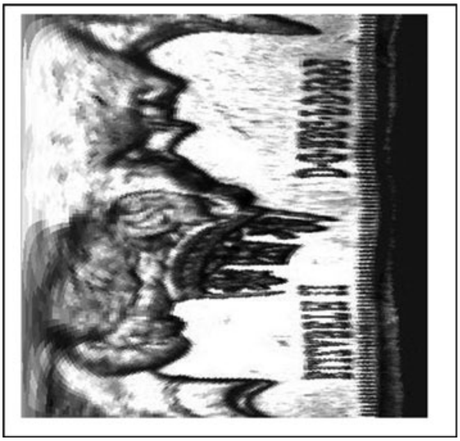
File number 23

Commands *ptc*

Below: original image



Right: processed image



41.257 pth

Format	*pth*(a)
Arguments	a Proportion of picture area ($0 \leq a \leq 1$)
Defaults	None
Output values	None
Image type	Grey-scale
Type	Threshold
Function	Threshold at that intensity level so that a proportion a of the pixels are set to white
Switch images	Yes
Typical uses	Analysing textured surfaces and other applications where the proportion of bright and dark pixels is known beforehand
Remarks	Function *ipp* provides automatic compensation for lighting variations. Also see *ipp*, *tmd*

41.258 push

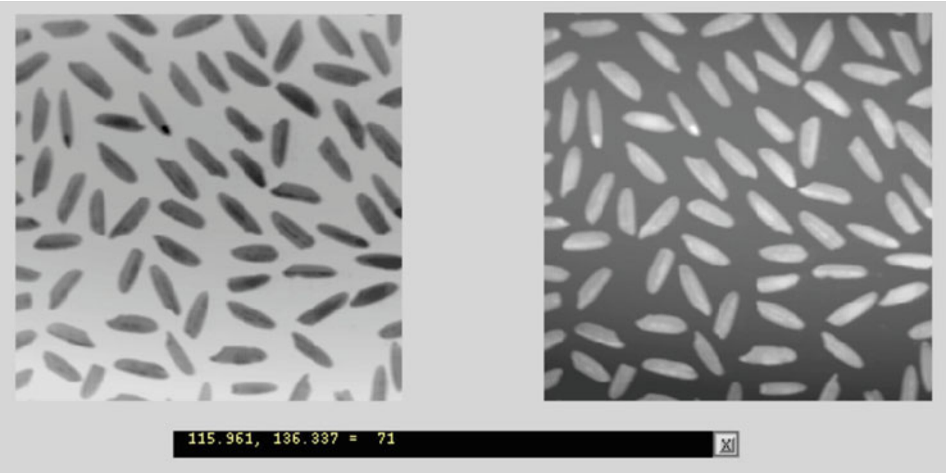
Format	*push*
Arguments	None
Output values	None
Image type	Any
Type	Stack operator
Function	Push the current image onto the image stack. The current and alternate images do not change
Switch images	No
Remarks	Also see *pop*, *ssk*, *rsk*, *csk*

41.259 pxv

Format	*pxv*(*a*)	
Arguments	a	Interactive display ON/OFF
Defaults	a	1 (display ON)
Output values	None	
Image type	Any	
Type	Miscellaneous	
Function	Interactive investigation of position and intensity	
Switch images	No	
Remarks	*pxv* (or *pxv*(*1*)) invokes the MATLAB function *impixelinfo*, which interactively displays information about image pixels in an image on the screen. This can be any image in Figure 1 or 2. Neither the Current nor Alternate image image is modified. Also see *cur*	

Illustration

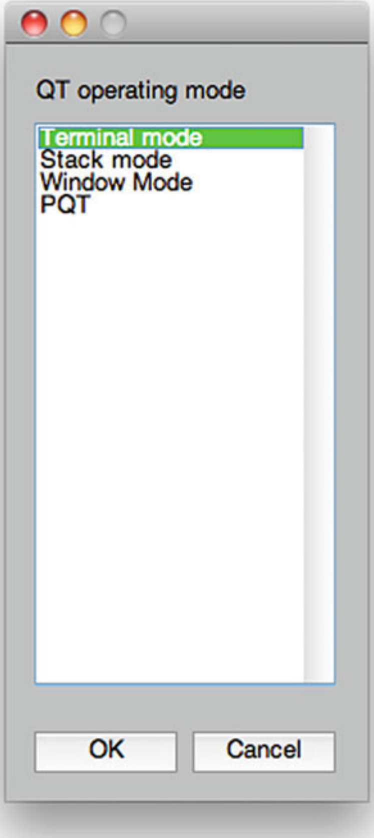
Screen dump of *Figure No. 1*. The current image is always displayed on the left and the alternate image on the right.



```
115.961, 136.337 =  71
```

41.260 qt

Format	*qt*
Arguments	None
Output values	None
Image type	Not relevant
Type	Not relevant
Function	The user is invited to select the operating mode via a pop-up menu
Switch images	No



QT operating mode

- Terminal mode
- Stack mode
- Window Mode
- PQT

41.261 qtp

Format	*qtp*(*a*)
Arguments	*a* Enable (a = 1) or disable (a = 0) functions that are
Default	*a* Flip the *enabled/disabled* state
Output values	None
Image type	Not relevant
Type	Information
Function	*qtp*(*0*) Disable new functions
	qtp(*1*) Enable new functions, previously downloaded using *qtu*
	qtp Flip the *enabled/disabled* state
Switch images	No

41.262 qts

Format	*qts*
Arguments	None
Output values	None
Image type	Any
Function	Operate in *Stack Mode*

Remarks

In *Stack Mode*, the user prompt is changed to

QT:

All standard QT commands (i.e., those available in *Terminal Mode* and listed by the *hlp* command) are executed as normal and the current image is then pushed onto the image stack

To exit *Stack Mode* and re-enter *Terminal Mode*, type any one of the following:

stop, *end*, *quit*, *exit*, *halt*

The image stack is implemented using a series of TIFF image files called

stack_image(*0*).*tif*, *stack_image*(*1*).*tif*, . . ., *stack_image*(*stack_size*).*tif*

(The stack size (variable *stack_size*) is initialised in *qtstart*.) These images are stored in folder *QT_work_file*s and together form what is called the *stack array*. The top of the stack is indicated by the global variable *stack_pointer* The file

stack_image(*stack_pointer*).*tif*

is therefore the top of the stack. When the current image is pushed onto the stack (command *push*), the following actions happen:

(i) The variable *stack_pointer* is incremented by 1, modulo *stack_size*

(ii) The current image is copied to file *stack_image*(*stack_pointer*).*tif*

No other files in the stack array are changed. When the stack is popped, the following actions result:

(i) The variable *stack_pointer* is decremented by 1, modulo *stack_size*

(ii) File *stack_image*(*stack_pointer*).*tif* is copied into the current image. Hence, it is usually possible to recover an image that was recently held at the top of the stack, if we simply decrement *stack_pointer* by, modulo *stack_size*. This is the basis of the command *uds*. (This facility is useful, if the user has been too enthusiastic in the use of the command *b*. See table below.) The best way to understand the stack operators is to enter Stack Mode (*qts*), then use *rsk*. This fills the stack with a random selection of standard QT images and displays part of the stack. Then, experiment with *pop*, *push*, *uds*, *rsi*, *f* and *b*.

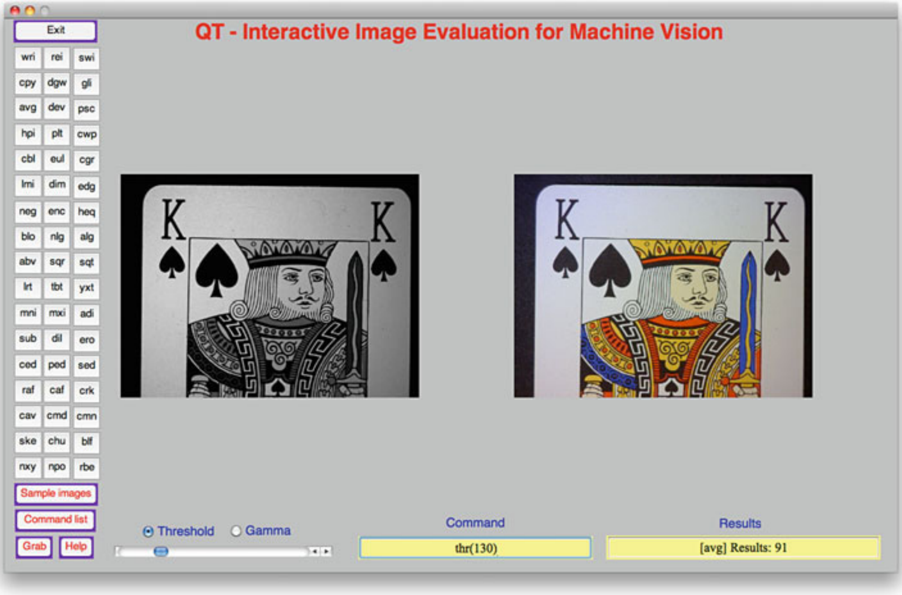
Secondary Commands, implemented by qts

These commands are valid only in *Stack Mode* (i.e., after typing *qts.*) The behaviour of some standard QT commands is slightly different in *Stack* and *Terminal Modes.* (Indicated by '∗')

Command	Function
end∗	Exit *Stack Mode*; return to *Terminal Mode*
exit	
stop	
quit∗	
halt	
qt(*l*)∗	Enter Window Mode
spon	Stack enabled
spof	Stack disabled
sdon	Stack display on
sdof	Stack display off
Null (*CR only*)	Interchange current and alternate images
pop	Pop the image stack
b	Rotate the stack array back one step
c	Grab an image
f	Rotate the stack array forward one step
h	Help: display command list
r	Read image from temporary file (See command 'w')
s	Read same standard QT image as the last time *rni* was used
t	Get image from the top of the stack but do not alter the stack
w	Write image to a temporary file (See command 'r')
z	Interchange current and alternate images. Stack is not changed
csk∗	Clear the stack. Fill it with images generated by a Gaussian noise source
dtx	Display text ON/OFF
ssk	Show part of the stack array
Other QT commands∗	Execute command as normal, then push current image onto the stack
Failed command	'WARNING: That command was not recognized – try again'

41.263 qtw

Format	*qtw*
Arguments	None
Output values	None
Image type	Not relevant
Type	System control
Function	Operate in Stack Mode
Switch images	No
Remarks	Provides a convenient frame-work facilitating rapid interactive image processing and is particularly helpful for training inexperienced users. Many popular functions can be run, simply by clicking on labelled buttons. A command-line interface and slider control of threshold and gamma correction are included


QT - Interactive Image Evaluation for Machine Vision

Command: thr(130)

Results: [avg] Results: 91

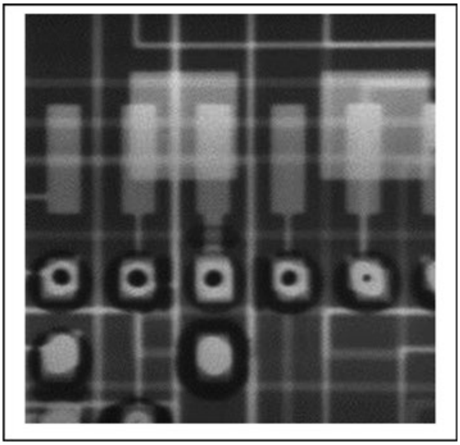
41.264 r

Format	$r(a)$
Function	Read an image from the image stack
Arguments	a
Defaults	0
Output values	None
Image type	Any
Type	User interface
Switch images	No
Function	Synonym for *rsi*
Remarks	This function is included simply to improve the effectiveness of the interactive dialogue. Alternatively, click on the appropriate thumbnail in Figure 2

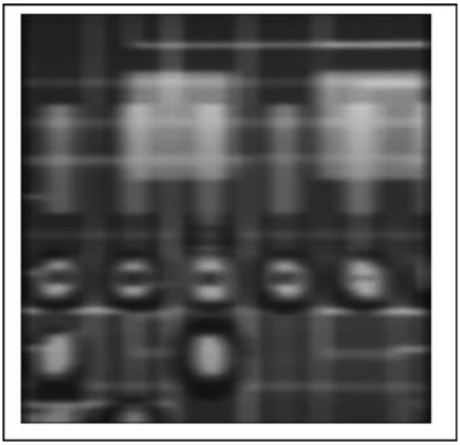
41.265 raf

Format	*raf(a, b)*
Arguments	*a* Filter kernel size, horizontal *b* Filter kernel size, vertical
Defaults	No arguments $a = 3$ $b = 3$ Single argument $b = a$
Output values	None
Image type	Grey
Type	Linear filter
Function	Local averaging within a rectangular window of size a × b pixels
Definition	For $a = b = 3$: $C(i, j) \Leftarrow [A(i-1, j-1) + A(i-1, j) + A(i-1, j+1) + A(i, j-1),+A(i, j) + A(a, j\ \ +1)+ A(i+1, j-1) + A(i+1, j) + A(i+1, j+1)]/9$
Switch images	Yes
Typical uses	Reducing the effects of noise Low-pass filtering an image, prior to subtracting the original, as part of a high-pass filter
Remarks	Both horizontal and vertical edges are blurred. The effect is most pronounced if *a* and/or *b* are large
Illustration	
Input image	Printed circuit board
File number	135
Command	*raf(15, 3)*
Remark	Vertical edges are blurred more than horizontal edges by this filter

Left: original



Right: after processing



41.266 rbe

Format *rbe*(*a*, *b*)

Arguments *a* If *a* = 0, remove blobs touching all 4 sides, otherwise remove blobs on 2 sides

b Connectivity (*b* = 4 or 8)

Defaults *a* 0

b 8

Output values None

Image type Binary

Type Binary image transform

Function Remove *b*-connected blobs (*b* = 4 or 8) that touch the edge of the current image

Definition If *a* = 0, remove all blobs that touch any of the 4 sides of the image

Otherwise, remove all blobs that touch either the left- or top-side of the image

In both cases, a blob can be defined in terms of 4- or 8-connectivity

Switch images Yes

Typical uses Analysing population density, size and shape of granular objects, such as seeds

Remarks To understand the importance of this function, imagine that the camera is viewing a small part of a larger field of uniformly scattered particles that do not touch or overlap (The image of rice grains below is one such example.) Some of the particles are only visible in part. Removing those blobs that touch the edge of the image, improves the accuracy of blob statistics as estimates of the characteristics of the overall population of physical objects. This remark applies to measurements such as area, size, eccentricity, solidity: functions *aeb*, *bpa*, *cab*, *cxa*, *eab*, *eqd*, *ext*, *oab* and *sly*.

In order to measure particle density accurately, *rbe*(*1*) is useful, since it eliminates those blobs that touch only two sides of the image (left or top)

Illustration

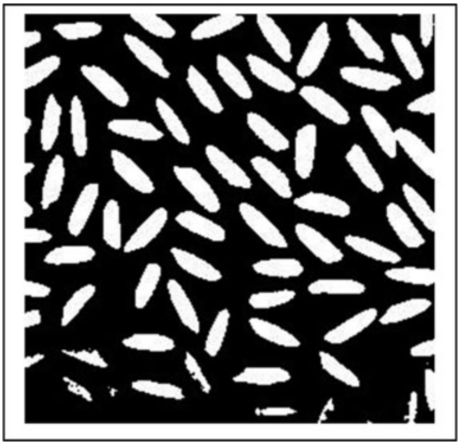
Input image Grains of rice, after thresholding

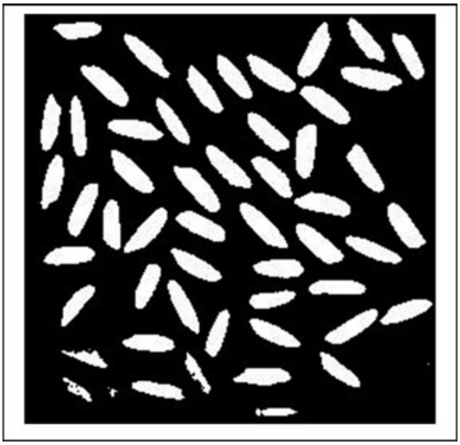
Image generation *rni*(*3*), *enc*, *thr*

Command Removing blobs from all four sides: *rbe* or *rbe*(*0*, *8*)

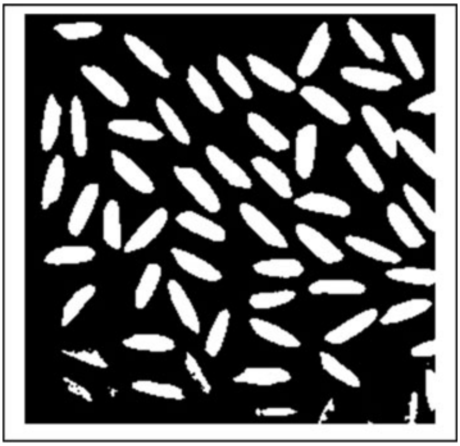
Removing blobs from only 2 sides: *rbe*(*1*) or *rbe*(*1*, *8*)

Top-left: original image *Top-right*: after *rbe*





Bottom: after *rbe*(*1*)



41.267 rbi

Format	*rbi*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Any
Type	Input/output
Function	Restore the current and alternate images from files placed on the disc by *sbi*
Switch images	No
Typical uses	Switching to a new application study temporarily without losing image data from the previous one
Remarks	The current and alternate image files can conveniently be stored on disc using *wbi*

41.268 rcf

Format *rcf*(*a*)

Arguments *a* Weight vector, 9 elements

Defaults None

Output values None

Image type Grey

Type Non-linear filter

Function Rank convolution filter, operating on a 3∗3 window

Description Similar to *con*, except that the intensities are sorted in rank order before weighting and adding

Definition Consider the point (i, j). The nine intensity values within its 3 × 3 neighbourhood are found and then sorted in rank order. These are then multiplied by the weights defined by the rescaled input vector *a* and are then summed. (Rescaling and the addition of the constant *k2* are performed so that the computed intensity values are normalised to lie within the range [0, 255].) Edge effects are removed by masking around the border of the image

Switch images Yes

Typical uses Edge detection. Noise reduction

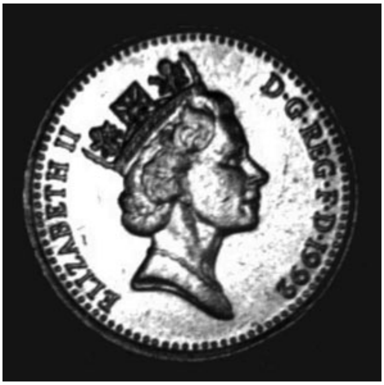
Remarks A range of operations similar to those performed by other QT functions can be achieved by *rcf*:

Weight matrix	Operation	Approximate QT equivalent
[1, 0, 0, 0, 0, 0, 0, 0, −1]	Edge detector	*lnb, wri, swi, snb, rei, sub*
[3, 2, 1, 0, 0, 0, −1, −2, −3]	Edge detector	No simple command sequence
[0,−1,−2,−1, 0, 1, 2, 1, 0]	Edge detector	No simple command sequence
[0, 0, 0, 0, 1, 0, 0, 0, 0]	Median filter	*mdf*(3)
[0, 0, 1, 2, 3, 2, 1, 0, 0]	Attenuate noise	*mdf*
[−1, 0, 0, 0, 3, 0, 0, 0, −1]	Sharpen	*shp*
[1, 2, 3, 4, 5, 4, 3, 2, 1]	Hybrid: low pass filter and median filter	No simple command sequence
[−1, 0, 0, 0, 2, 0, 0, 0, −1]	Approx. to second derivative	*lpc*

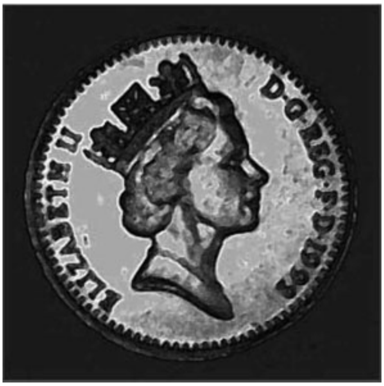
Illustration

Input image	Coin
File number	23

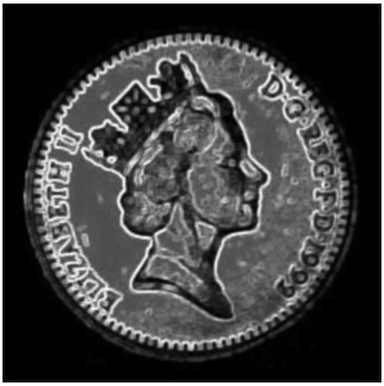
Top-left: original



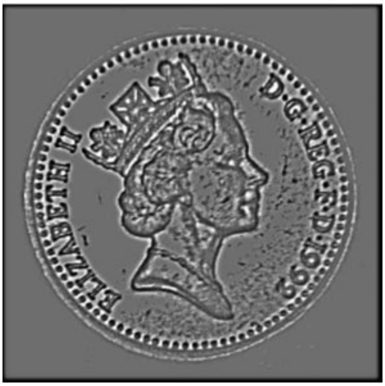
Top-right: after *rcf*([*1, 0, 0, 0, 3, 0, 0, 0, −1*]), *enc*



Bottom-left: after *rcf*([*0,−1,−2,−1, 0, 1, 2, 1, 0*]), *enc*



Bottom-right: high-pass filter; *rcf*([*1, 2, 3, 4, 5, 4, 3, 2, 1*]) applied 5 times then subtract original and enhance contrast (*enc*)



41.269 rci

Format	*rci*(*a*)
Arguments	*a* File number
Defaults	None
Output values	None
Image type	Colour image file
Type	Input/output
Function	Read a colour image file, defined by its number; do not convert it to grey-scale format
Remarks	This command is used to read one of the standard colour numbered images, held in a folder called QT_images. This folder must be located on MATLAB's search path. Also see *rni*

41.270 rdc

Format $[p, q] = rdc$

Arguments None

Output values p Vector of radial distance values, measured from the centroid
q Vector of circumferential distances, estimated from the chain code

Image type Binary

Type Image coding

Function Select the biggest blob and fill any lakes. Then, perform a raster scan, to find the top-left white pixel. Starting at this point, trace the outer edge. For each pixel on the edge, measure:
(a) The distance from the centroid of the blob (array p)
(b) The distance around the circumference. This distance is estimated from the chain code by counting 1 for each even-numbered chain-code element and √2 for each odd-numbered element.

Switch images No

Typical uses Determining the orientation of objects that do not have strong limb-like features and are not obviously elongated.

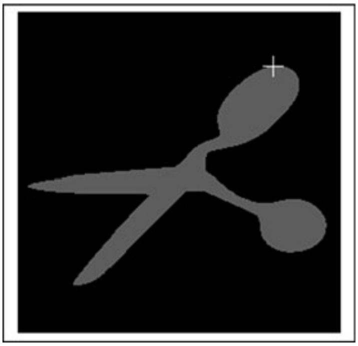
Remarks This function provides a periodic, single-valued representation of an edge contour. The output vectors p and q describe one cycle. Vector p measures distances from the centroid at a series of non-uniformly spaced points. Their positions around the edge of the blob is described by the output vector q. Hence, direct comparison of two p vectors is not possible and interpolation must be used first, to obtain homogenous sampling. To match two binary objects and compare the orientation of one to the other, we can use cross-correlation, after the sampling intervals have been homogenised. This is useful for shapes that have no obvious 'long axis', or possess distinctive features such as sharp corners, elongated limbs, well-defined and easily identifiable concavities, etc.

Illustration

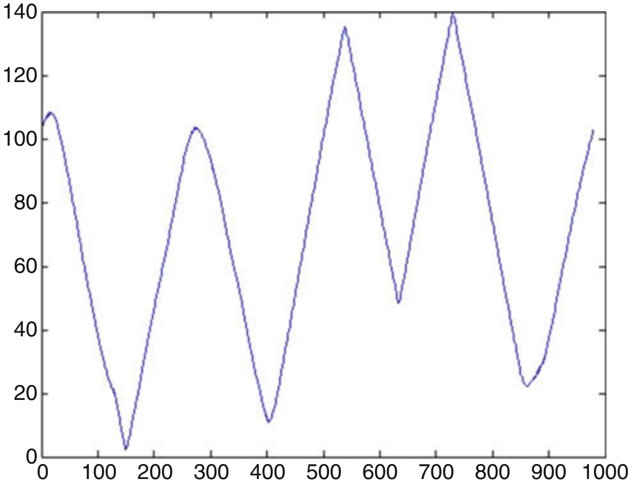
Image generation *rni(49), thr, blf*

Plotting: Linear: $[a, b] = rdc$; $plot(b, a)$
Polar: $[a, b] = rdc$; $b = 2*pi*b/max(b)$, $polar(b, a)$

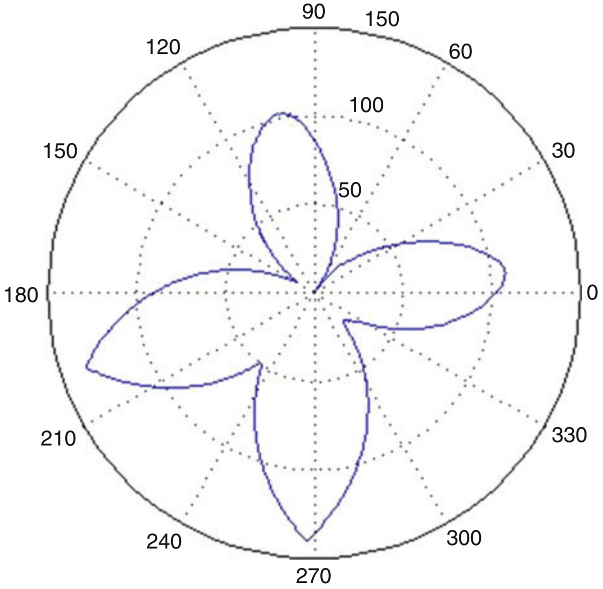
Top: original (The white cross indicates the starting position for the edge follower)



Centre: linear plot (Abscissa: circumferential distance. Ordinate: radial distance)



Bottom: polar plot (Angle: circumferential distance. Radius: radial distance)



41.271 rdf

Format *rdf*(*a*, *b*, *c*)

Arguments *a* Width of the filter window

b Height of the filter window

c Rank order of the two intensity values to be subtracted

Defaults Two arguments *(a, c)* supplied: The filter window is a square of size *a* × *a* pixels; *rdf*(*a*, *c*) has the same effect as *rdf*(*a*, *a*, *c*)

Output values None

Image type Grey

Type Monadic

Function The intensity values within each rectangular neighbourhood of size a × b are placed in rank order. The *c*th largest and *c*th smallest values are then subtracted and the resulting intensity scale expanded to improve contrast (using *enc*)

Switch images Yes

Typical uses Edge detection in images where there are no step-like intensity transitions

Remarks When an edge detector is needed for an image with no sharp edges, *rdf* produces better results than operators that always a 3 × 3 window. Notice that *rdf*(*3*, *1*) and *edd*(*2*) produce similar results. Also see *rkf*

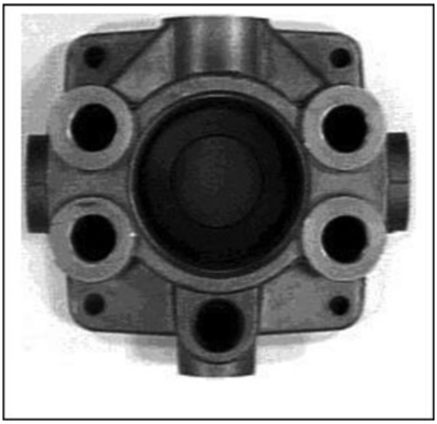
Illustration

Input image Hydraulics manifold

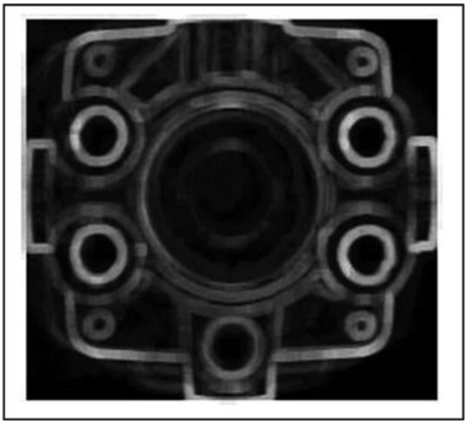
File number 37

Command *rdf*(*7*, *7*, *3*)

Left: original

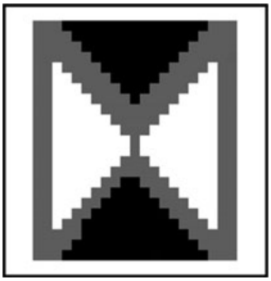


Right: after processing



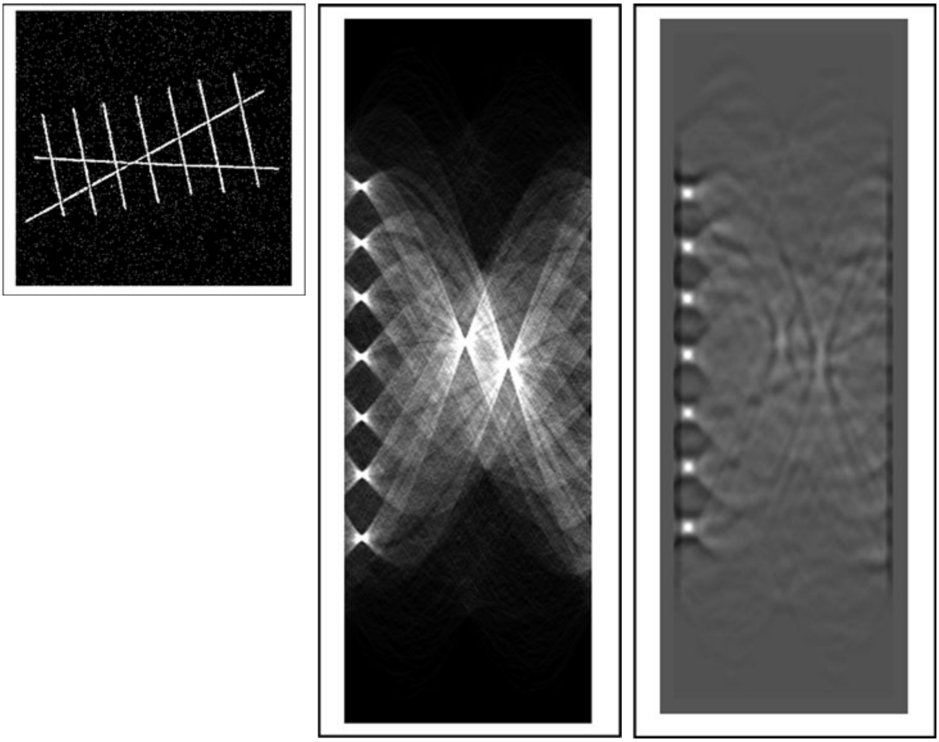
41.272 rdn

Format	*rdn*
Arguments	None
Defaults	None
Output values	None
Image type	Grey scale or binary
Type	Global transform
Switch	Yes
Function	Radon transform
Switch images	Yes
Typical use	Detecting linear streaks in noisy images
Remarks	The Radon transform produces an image (RT) that bears no obvious resemblance to the original; the [X, Y]-coordinates of bright spots in the RT image indicate the orientation and position of lines in the original, respectively. Also see *irt* and *slf*. The X-shaped spots can be detected by applying an appropriate convolution mask to the RT image: *Key*: Black: −1 Grey 0 White +1



Illustration

Input image	Lines, with preprocessing: *crk*(*3*), *enc*, *thr*, *spn*(*0.05*)
File number	158
Command	*rdn*, *wri*, *rni*(*161*), *y* = *round*((*double*(*x*)−*127*)/*127*); *rei*, *con*(*y*)



Left: original *Centre*: after *rdn* *Right*: after *con*(*y*)

41.273 rea

Format	*rea*(*a*)
Arguments	*a* File path name
Defaults	None
Output values	None
Image type	Defined by file that is being read
Type	Input/output
Function	Read an image. If *a* specifies a file containing a colour image, it is converted to grey scale. The image may be stored in a variety of formats, including BMP, JPEG, GIF, PNG and TIFF (Consult the MatLab documentation for *imread*)
Remarks	This function is not so easy to use as *rei* but it is much more versatile. Also see *rni*

41.274 red

See *sed*. (*red* is equivalent to *sed*(*2*))

41.275 rei

Format	$d = rei(a, b)$	
Arguments	*a*	File name/number
	b	0, read image previously saved in TIFF file, or 1, read image previously saved in JPEG file
Defaults	*a*	0
	b	0, read image previously saved in TIFF file
Output values	*d*	File path
Image type	Defined by file that is being read	
Type	Input/output	
Function	Read an image saved previously, in either TIFF (if $b = 0$) or JPEG format ($b = 1$), using *wri*	
Remarks	This function simply build the path name (See *wri*) and then calls *rea*. Also see *rni*	

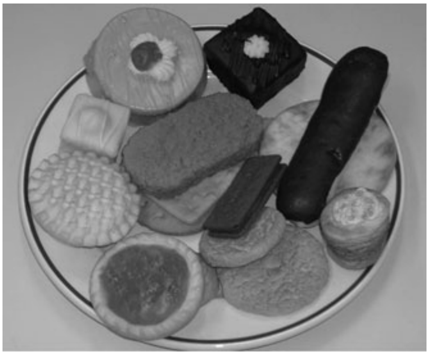
41.276 rev

Format	*function rev*
Arguments	None
Output values	None
Image type	Grey (Colour images can also be processed using pci)
Function	Bit reversal. For each intensity value, the bit order is reversed, so that the Most Significant BIT (MSB) becomes the Least Significant Bit (LSB) and the LSB becomes the MSB
Switch images	Yes
Typical uses	Obscuring the content of an image
Remarks	The function is exactly reversible; *rev*, *rev* reproduces the input image with no loss of information. An observer may be able to discern some gross structure in the output image but small features are not visible. To operate on a colour image, use *pci*(‘*rev*’)

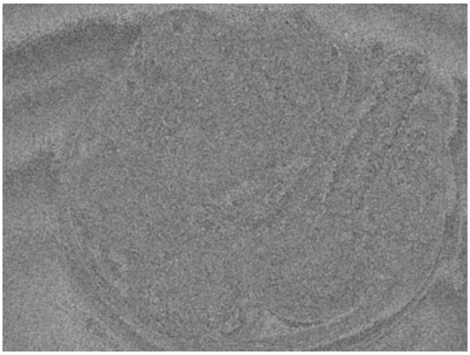
Illustration

Input image	Cakes and biscuits (Actually, these are plastic replica food products)
File number	225
Commands	*rni*(*225*), *rev* (Grey-scale image) *rci*(*225*), *pci*('*rev*') (Colour image)

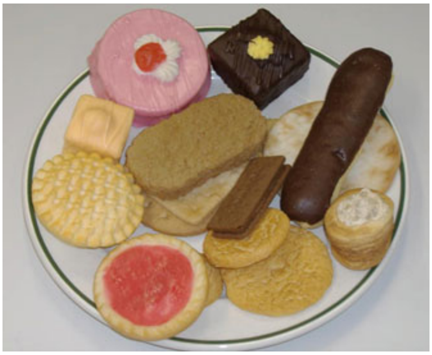
Top-left: original image (grey scale)



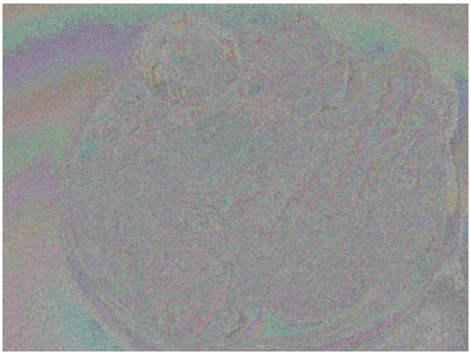
Top-right: after applying *rev*



Bottom-left: original image (colour)



Bottom-right: after applying *pci*('*rev*')



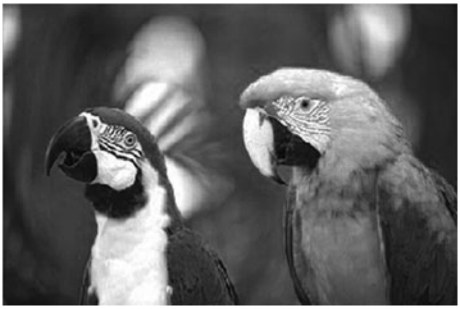
41.277 rgb

Format	*rgb*(*a*)	
Arguments	*a*	Select one of the *R*, *G*, *B* channels
Defaults	*a*	0 (Display *R*, *G* and *B* components separately. Do not alter the current image)
Output values	None	
Image type	Colour	
Type	Image display and channel separation	
Function	$a = 0$:	Separate and display the *R*, *G and B* channels. Do not alter the current image
	$a = 1$:	Set the current image to be equal to the *R*-channel of the colour image
	$a = 2$:	Set the current image to be equal to the *G*-channel
	$a = 3$:	Set the current image to be equal to the *B*-channel
Switch images	$a = 0$,	No
	$a \neq 0$,	Yes
Typical uses	Simplifying colour processing, reduce the problem one involving only grey-scale image processing	
Remarks	*rgb*(*0*) displays the 4 images in Figure 4 which is automatically scaled to fill the screen. Figure 4 closes when the user clicks on it. Also see *hsi*	

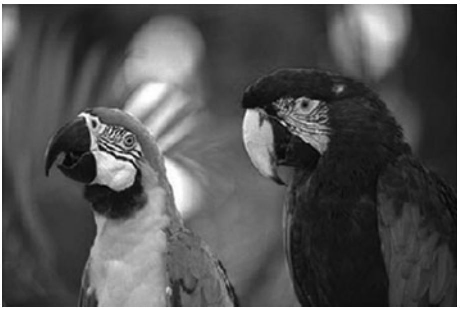
Illustration

Input image	Birds
File number	144
Command	*rgb*

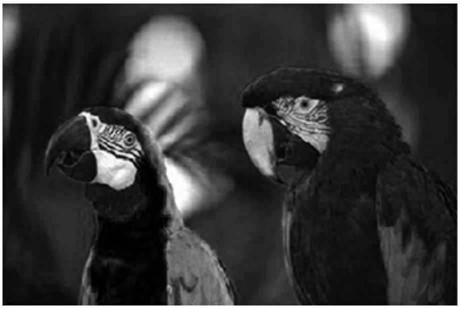
Top-left: *R*-channel



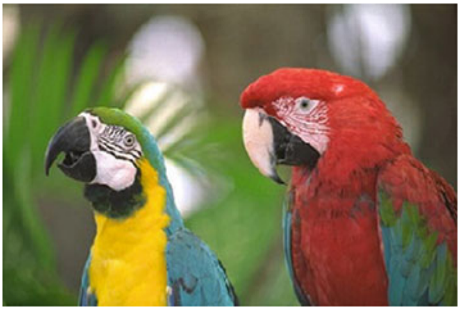
Top-right: *G*-channel



Bottom-left: *B*-channel



Bottom-right: original colour image



41.278 rgb_to_xyz/xyz_to_rgb

Format	$[X, Z] = rgb\_to\_xyz(R, G, B)$	
	$[R, G, B] = xyz\_to\_rgb(X, Z)$	
Arguments	R, G, B	R, G, B colour coordinates
	X, Y, Z	X, Y, Z colour coordinates
Defaults	None	
Output values	*X, Y, Z*	XYZ colour coordinates
	R, G, B	RGB colour coordinates
Image type	Not applicable	
Type	Colour processing	
Function	Convert colour coordinates RGB to/from XYZ	
Switch images	No	
Illustration		
Input image	plug	
File number	112	
Command	*thr*(*115*)	

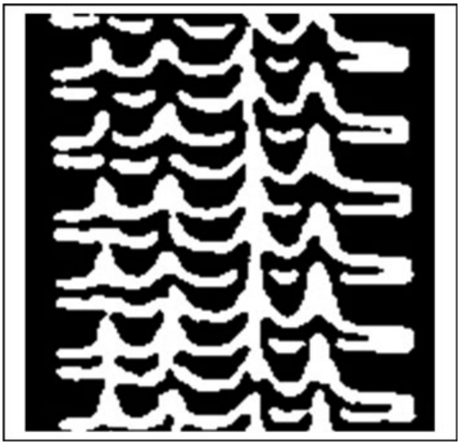
41.279 rim

Format	*rim*(*a*)	
Arguments	*a*	File name/number
Defaults	*a*	0 (zero)
Output values	None	
Type	Input/output	
Switch images	Yes	
Function	Read an image from the QT_images folder. The image may be nominated by a string, or using a numeric argument. The image may be coded in JPEG (file extension *jpg*) or TIFF (file extension *tif* or *tiff*) format	
Remarks	Also see *rea*, *rni*, *rci*, *gwi* and *wri*	

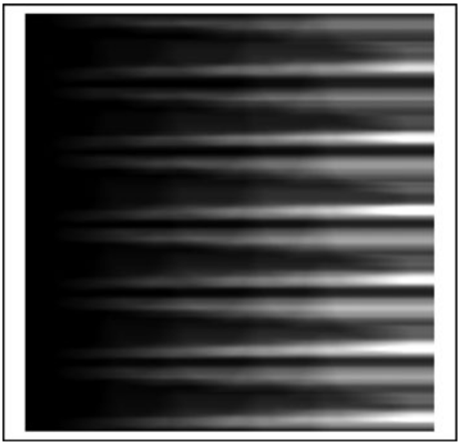
41.280 rin/cin

Formats	*rin* *cin*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Global
Function	Row/column integrate
Definition	rox: $C(i, j) \Leftarrow (C(i-1, j) + A(i, j))/N$, where $C(1, j) = A(1, j)$ (N is the image width) cox: $C(i, j) \Leftarrow (C(i, j-1) + A(i, j))/N$, where $C(i, 1) = A(i, 1)$ (N is the image width)
Switch images	Yes
Typical uses	Detecting linear horizontal/vertical features. The illustration below shows how *rin* and *cin* may be used to locate the 'feathering' on a certain type of cake decoration
Remarks	Also see *rdn*
Illustration	
Input image	Cake decoration pattern
File number	54 (cropped)
Command	*rin/cin*

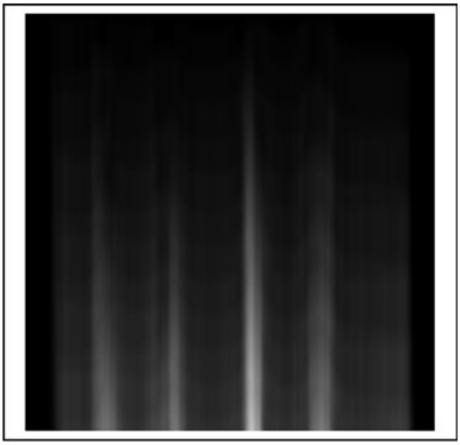
Top-left: original



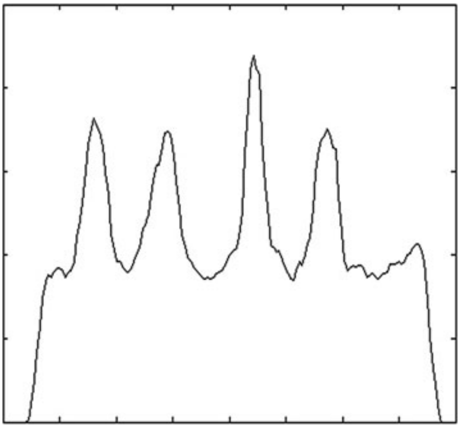
Top-right: *rin*



Bottom-left: *cin*



Bottom-right: intensity profile: *cin*, *yxt*, *csh*, *yxt*, *plt*



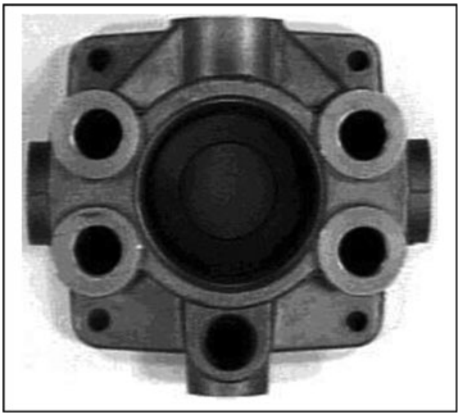
41.281 rkf

Format	*rkf*(*a*, *b*, *c*)	
Arguments	*a*	Width of the filter window
	b	Height of the filter window
	c	Rank order of the intensity value to be selected
Defaults	Two arguments *(a, c)* supplied: The filter window is a square of size *a* × *a* pixels	
Output values	None	
Image type	Grey	
Type	Monadic	
Function	The intensity values within each rectangular neighbourhood of size a × b are placed in rank order. The *c* th largest is then selected as the value for the pixel at the centre of that window	
Switch images	Yes	
Typical uses	Noise reduction	
Remarks	*rkf* is able to emulate *mdf*, *lnb*, *snb*, *dil and ero* and is less sensitive to noise than these other operators. Also see *nrf*, *mdf* and *rdf*	

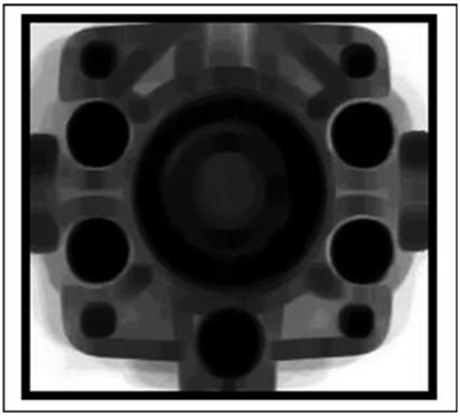
Illustration

Input image	Hydraulics manifold
File number	37
Command	*rkf*(*11*, *11*, *7*)

Left: original



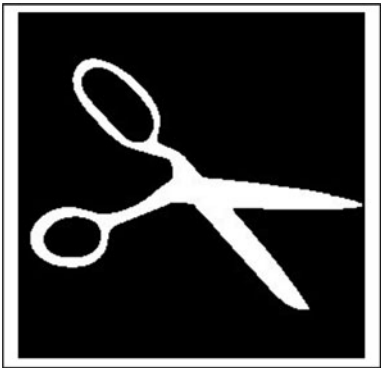
Right: after processing



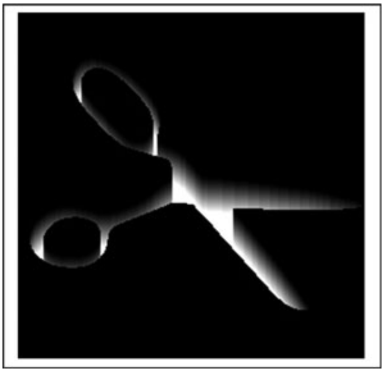
41.282 rlc/rlr

Format	*rlc* *rlr*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob transform
Switch	Yes
Function	Run-length code along the columns/rows
Switch images	Yes
Typical uses	Blob-shape analysis
Illustration	
Input image	Scissors
File number	48
Command	*rlc/rlr*

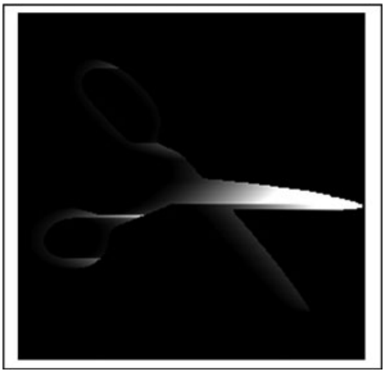
Top-left: original



Top-right: *rlc*



Bottom: *rlr*



41.283 rlr

See rlc.

41.284 rmn

See *rmx*.

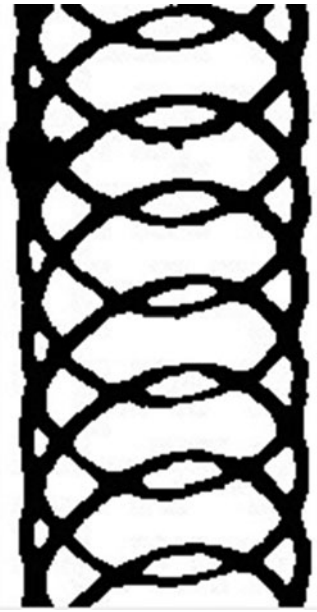
41.285 rmx/rmn

Format	*rmx* *rmn*
Arguments	None
Output values	None
Image type	Grey-scale
Type	Grey-scale image analysis
Function	Find the regional maxima/minima of the intensity function
Switch images	Yes
Typical uses	Analysing textured surfaces and images of granular material
Remarks	*rmx* can be used to analyse binary images, by applying *rmx* to the output of a suitable operator, such as the grass-fire transform (*gft*). *gft*, *rmx* provides an approximation to reversing the watershed function (*wsh*)

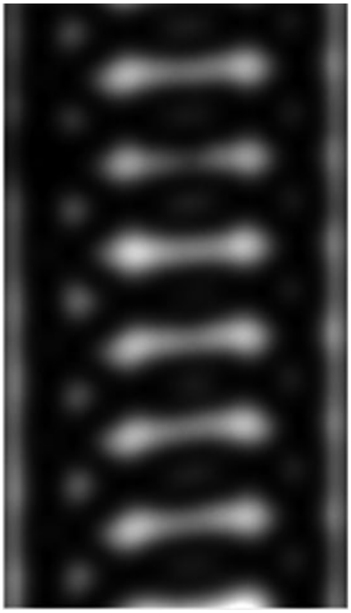
Illustration

Input image	Cake decoration pattern
File number	55
Commands	*rni*(*55*), *thr*, *neg*, *thn*, *neg*, *gft*, *enc*, *caf*(*3*), *rmx*, *shk*, *crs*(*3*), *rni*(*55*), *mxi*

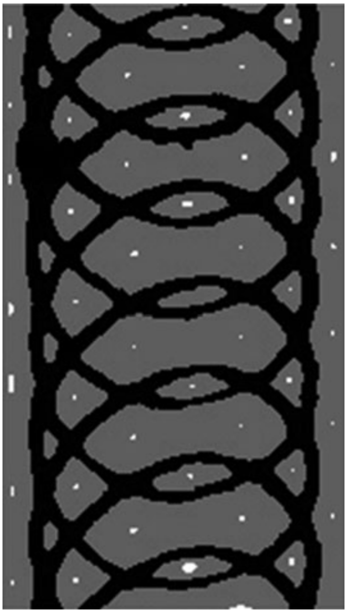
Top-left: original image



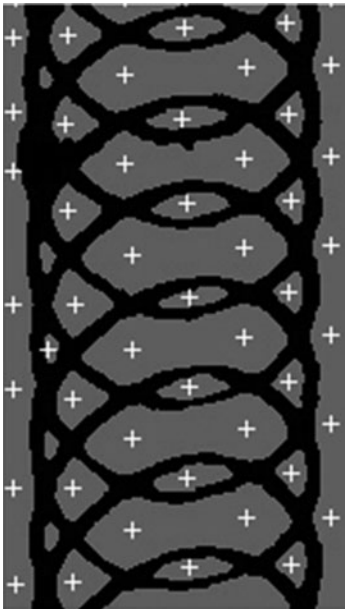
Top-right: after *gft*



Bottom-left: after *rmx*



Bottom-right: after mxi



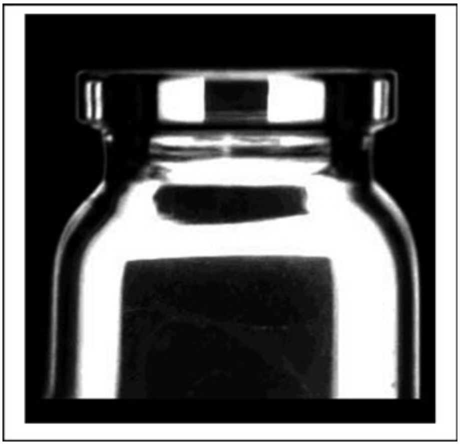
41.286 rni

Format	*rni*(*a*)
Arguments	*a* File number
Defaults	None
Output values	*None*
Image type	Colour or grey scale image file can be read. Colour images are converted to grey
Type	Input/output
Function	Read one of QT's standard images, defined by its number and stored locally in a folder called QT_images (This must be on MATLAB's search path)
Remarks	This function reads JPEG and TIFF images, which may be colour or monochrome. Also see *rci*, *rea*, *rei*, *img*, *gwi*

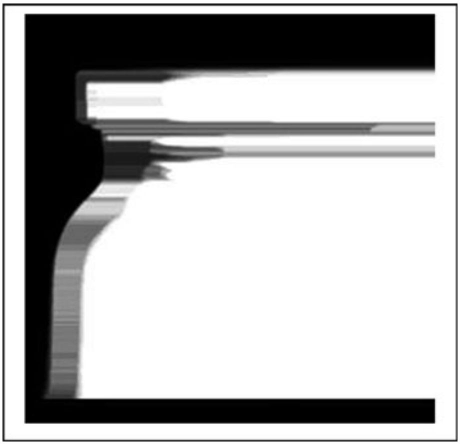
41.287 rox/cox

Formats	*rox* *cox*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Global
Function	Row/column maximum
Definition	rox: $C(i, j) \Leftarrow MAX(C(i-1, j), A(i, j))$, where $C(1, j) = A(1, j)$ cox: $C(i, j) \Leftarrow MAX(C(i, j-1), A(i, j))$, where $C(i, 1) = A(i, 1)$
Switch images	Yes
Typical uses	Analysing silhouettes of convex and nearly convex objects, such as bottles and coins; use *rox*, *wri*, *swi*, *lrt*, *rox*, *lrt*, *rei*, *mni* to 'fill in' the dark centre of the vial. (See illustration below.) This can lead to a significant reduction in the amount of processing required later
Remarks	The function *cox* is defined here by inter-changing the image axes (*yxt*) and applying *rox*. It is, of course, possible to define *cox* directly, in a similar manner to *rox*. These functions may be regarded as creating shadows of binary objects, with the "illumination" originating at the left (*rox*), or top (*cox*). Also see *rin*/*cin*
Illustration	
Input image	Glass vial
File number	32
Command	*rox/cox*

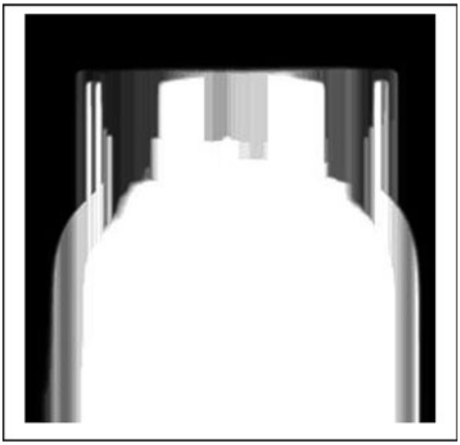
Top-left: original



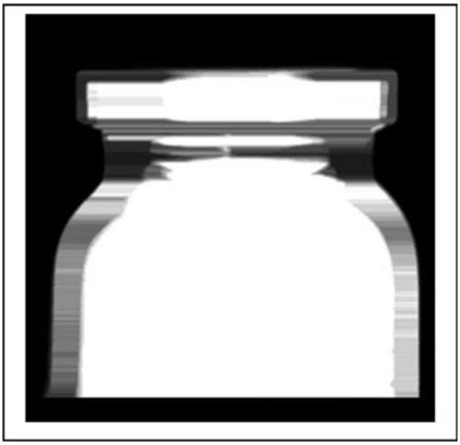
Top-right: *rox*



Bottom-left: *cox*



Bottom-right: after *rox*, *wri*, *swi*, *lrt*, *rox*, *lrt*, *rei*, *mni*



41.288 rsb

See *csh*

41.289 rsf

Format	*rsf*
Arguments	None
Output values	None
Image type	Grey
Type	Image warping
Function	Automatically resize and position the image display windows (Figure 1 and Figure 2) to fit onto the computer screen
Switch images	No
Remarks	The position and size of the Command Window should be adjusted manually

41.290 rsi/wis/interactive reading and writing of images

Format *rsi*(*a*)
wis(*a*)

Arguments *a* Integer in the range [−1, 7]

Defaults *a* 0

Output values None

Image type Any

Type Read/write images

Function *rsi* Read an image from the image stack
wis Write the current image into the image stack

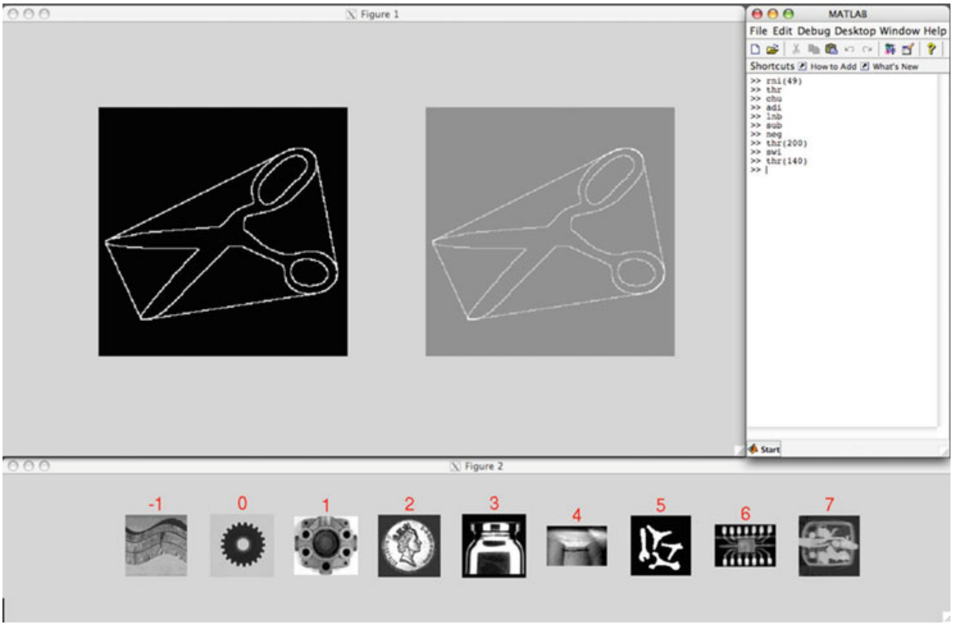
Switch images *rsi* Yes
wis No

Typical uses Storing and retrieving intermediate processing results during an interactive session

Remarks *rsi* and *wis* are used in the same way as *rei* and *wri*, providing random access to nine images stored on disc. The operators *pop* and *push* are disabled. The image stack is used only to provide the visual display, thereby eliminating the need for the user to remember where images are stored. Operators *r* and *w* have the same function as *rsi* and *wis*, respectively. The user can also read/write images by clicking on the appropriate thumbnail while holding the CONTROL key down. The following table summarises these functions. Also see *rni*, *rti*, *rsk* and *ssk*

Writing	Reading	Comments
wri(a)	rei(a)	*Argument a* can be numeric or alpha-numeric string.
wis(a) or w(a)	rsi(a) or r(a)	*a* is numeric Image stack display is updated (*wis* only)
CNTRL/CLICK	CLICK	Interactive writing/reading images Convenient and fast for the user

Screen shot, showing the image stack display (Figure 2) current and alternate images (Figure 1, *left* and *right* respectively) and the command window (right-hand-side).



```
>> rni(49)
>> thr
>> chu
>> adi
>> lnb
>> sub
>> neg
>> thr(200)
>> swi
>> thr(140)
>> |
```

41.291 rsk

Format	*rsk*
Arguments	None
Output values	None
Image type	Any
Type	Stack operator
Function	Fill the image stack with randomly chosen images, selected from those held in the folder QT_images
Switch images	No
Remarks	The main function of this operator is help the user demonstrate how the stack works. To do this, switch to Stack Mode (*qts*), or use *ssk* (stay in Terminal Mode), then observe the stack display (Figure 2), as images are popped (*pop*) and pushed (*push*) onto the stack. Also see *csk*

41.292 rsz

Format	*rsz*(*a*, *b*)	
Arguments	*a*	Width of the output image
	b	Height of the output image
Defaults	No arguments:	$a = b = 512$
	One argument:	$b = a$
Output values	None	
Image type	Any	
Type	Image geometry	
Switch	Yes	
Function	Adjust the size/aspect ratio of the image	
Switch images	Yes	
Typical uses	Adjusting the size or aspect ratio of an image to make subsequent processing faster, or more convenient	
Remarks	The output image can be larger or smaller than the original, along either axis	

Illustration

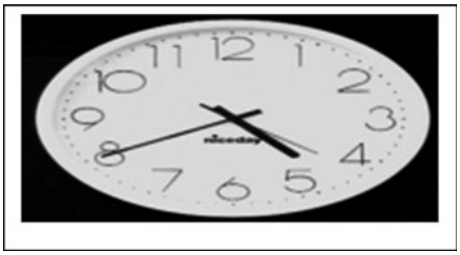
Input image Clock, cropped manually

File number 148

Command *rsz*(*200*, *100*)

Top-left: original

Top-right: after processing



41.293 rti

Format *rti*
Arguments None
Output values None
Image type Any
Type Image display
Function Display the (9) most popular QT test images in the image stack
Switch images No

Typical screen display. *rti* loads images into the stack (9 thumbnails at the bottom). An image can be loaded into the current image, simply by clicking on the appropriate thumbnail.

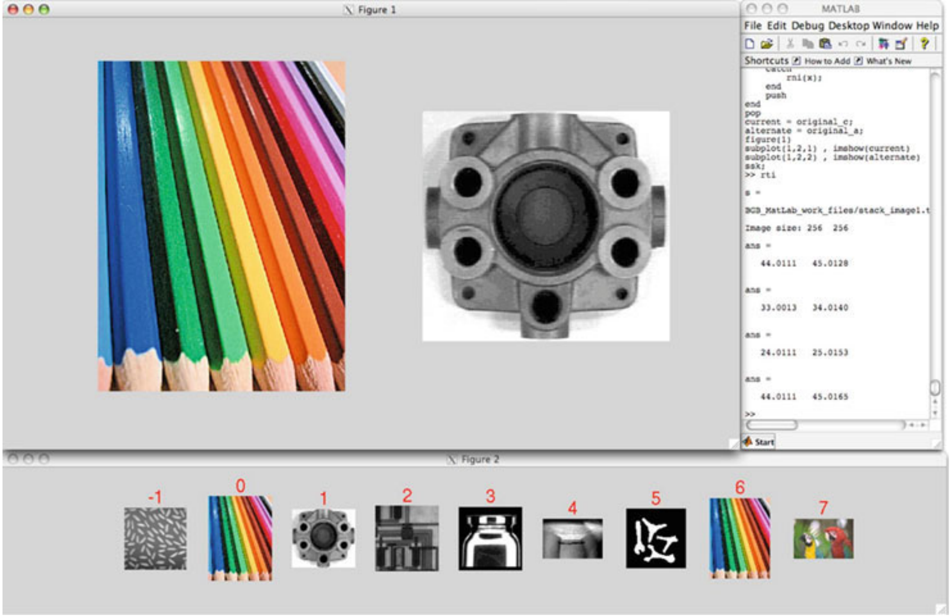


Figure 1
MATLAB
File Edit Debug Desktop Window Help
Shortcuts How to Add What's New
Start
Figure 2
-1
0
1
2
3
4
5
6
7

```
        rni(x);
    end
    push
end
pop
current = original_c;
alternate = original_a;
figure(1)
subplot(1,2,1) , imshow(current)
subplot(1,2,2) , imshow(alternate)
ask;
>> rti

s =

DGB_MatLab_work_files/stack_image1.t

Image size: 256  256

ans =

   44.0111   45.0128


ans =

   33.0013   34.0140


ans =

   24.0111   25.0153


ans =

   44.0111   45.0165

>>
```

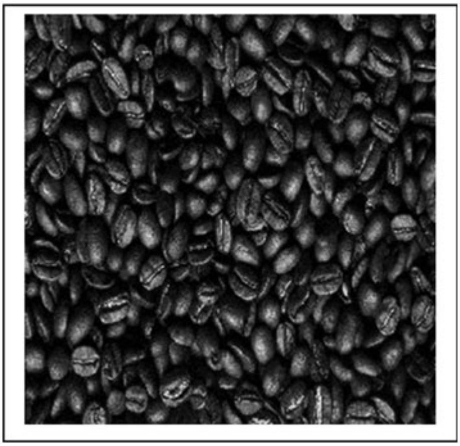
41.294 rxm

Format	*rxm*(*l*, *u*)	
Arguments	*l*	Lower limit
	u	Upper limit
Defaults	*l*	3
	u	5
Output values	None	
Image type	Grey scale	
Type	Image filter	
Function	Relaxed median filter	
Switch images	Yes	
Typical uses	Reducing the effects of camera noise and eliminating fine texture	
Reference	A. Ben Hamza et al, '*Removing Noise and Preserving Details with Relaxed Median Filters*', Journal of Mathematical Imaging and Vision, 11, 161–177, 1999	
Remarks	*rxm* provides a better filter for reducing the effects of impulse noise than does the standard median filter (*mdf*)	

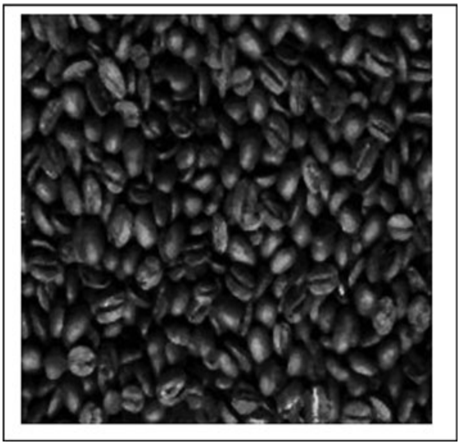
Illustration

Input image	Coffee beans
File number	4
Command	*rxm*(*5*, *11*)

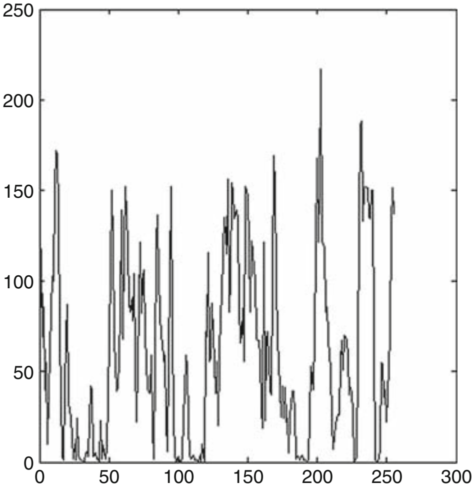
Top-left: original image



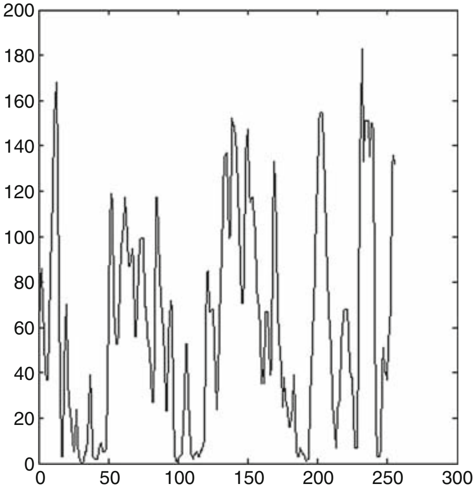
Top-right: after processing



Bottom-left: intensity profile for the original image



Bottom-right: intensity profile after *rmx*(*5*, *11*)



41.295 sai

Format	*a* = *sai*
Arguments	None
Defaults	Not applicable
Output values	*a* Sum of all intensities
Image type	grey scale
Type	Image measurement
Switch	No
Function	Sum the intensities of all pixels
Typical uses	Camera set-up in variable lighting conditions
Remarks	The result from *sai* is the product of that from *avg* (average intensity) times the number of pixels

41.296 sat

See *hsi*

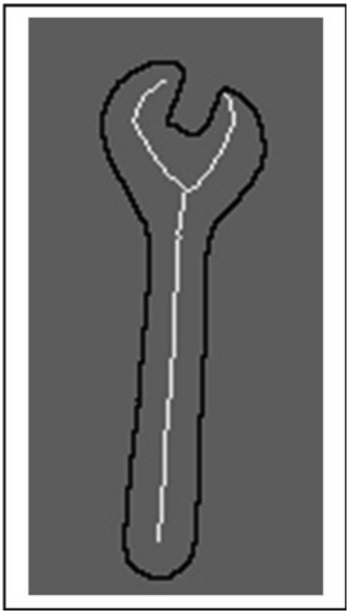
41.297 say

Format	*say*(*a*)
Arguments	*See below*
Defaults	*See below* ‘*You did not tell me what to say*’
Output values	None
Image type	Not relevant
Type	User interface support
Function	Converts a given string (argument *a*) to speech. A verbal warning is given if no string is specified
Switch images	No. Neither image is altered
Typical uses	Verbal feedback provides a good way to follow the execution of a cyclic QT program

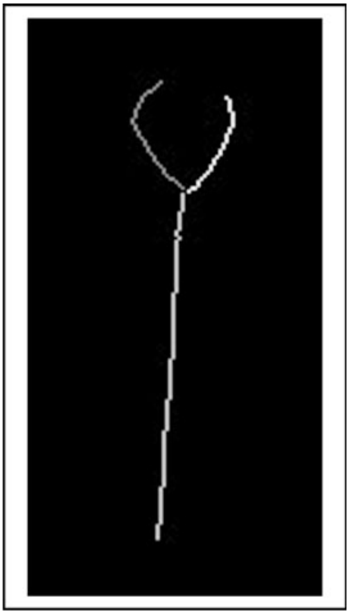
41.298 sbc

Format	*sbc*(*a*)
Arguments	*a* Pixel count
Defaults	*a* 1
Output values	None
Image type	Binary
Type	Blob transform
Switch	Yes
Function	Retain white pixels that have exactly *a* 8-neighbours
Typical uses	Shape analysis
Remarks	*sbc* (or *sbc*(*0*)) finds limb ends in ‘match-stick’ figures (equivalent to *jnt*); *sbc*(*2*) isolates individual limbs (illustrated below); *sbc*(*3*) finds joints (equivalent to *jnt*)
Illustration	
Input images	Four metal parts (file number 46), processed to isolate the spanner (wrench)
Image generation	*crp* (interactive selection of cropping rectangle), *tbt*, *thn*
Commands	*sbc*(*2*), *ndo*, *enc* The last two commands show the results more clearly

Left: binary: original image



Top-right: after processing



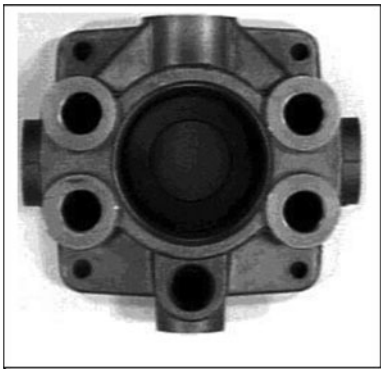
41.299 sbi

Format	*sbi*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Any
Type	Input/output
Function	Save both current and alternate images on disc
Switch images	No
Typical uses	Switching to a new application study temporarily without losing image data from the previous one
Remarks	The current and alternate image files can be restored using *rbi*

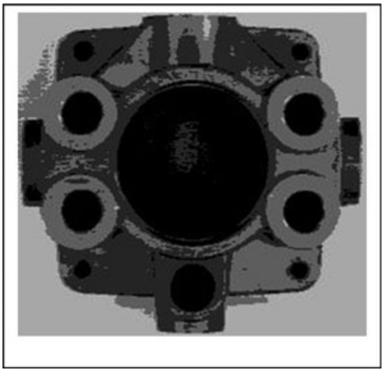
41.300 sca

Format	*sca*(*a*)
Arguments	*a* Number of bits to be retained ($1 \leq a \leq 8$)
Defaults	None
Output values	None
Image type	Grey
Type	Monadic
Function	Posterising; reducing the number of bits in the representation of the intensity for each pixel
Definition	Set the (8-*a*) least-significant bits to zero
Switch images	Yes
Typical uses	Plotting intensity contours Reducing the amount of information to be stored
Remarks	Also see *bcl*, *bcp*
Illustration	
Input image	Hydraulics component
File number	37
Command	*sca*(*2*)

Left: original



Right: after processing



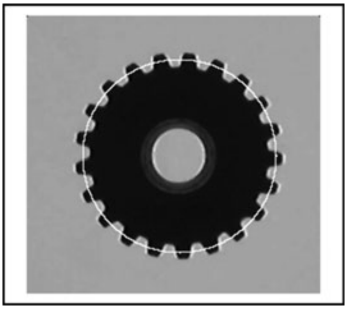
41.301 scc

Format	z = scc(x, y, r)	
Arguments	*x*	X-coordinate of the centre of the scanning circle
	y	Y-coordinate of the centre of the scanning circle
	r	Radius of the scanning circle
Defaults	*None*	
Output values	*z*	Vector containing 360 intensity values
Image type	Grey	
Type	Image analysis	
Switch	No	
Function	Circle-scan; sample the intensities uniformly around the circle centred on (*x*, *y*) and of radius *r*	
Typical uses	Reducing processing time, by converting a 2D image into 1D vector. Determining periodicity in known directions	
Remarks	The intensity is sampled at 1° intervals. Hence the output vector (*z*) has 360 elements. These elements are equal to −1, if the sample point is outside the image	

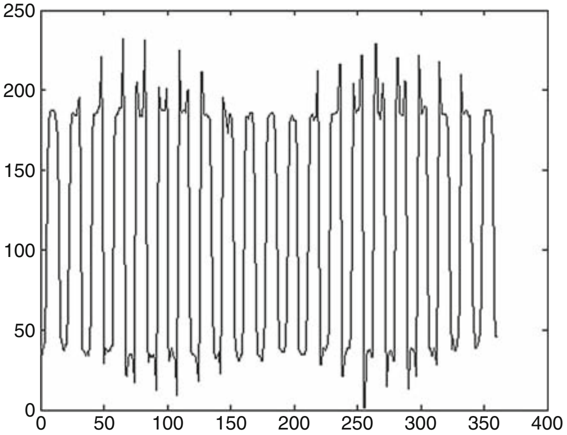
Illustration

Input image	Gear with median filtering to reduce noise effects
File number	171
Command	rni(171), *mdf*, *x* = *scc*(*129*, *119*, *81*), *figure*(*2*), *plot*(*x*), *figure*(*1*), *cir*(*129*, *119*, *81*, *255*)

Left: original with scan circle superimposed by *cir* (The circle parameters were defined manually)



Right: after *plot*(*z*) Horizontal axis, scan angle. Vertical axis, intensity



41.302 sci

Format	*sci*(*a*)
Arguments	*a* New intensity value
Defaults	*a* 0 (black)
Output values	None
Image type	Binary
Type	Drawing
Switch	Yes
Function	Set all pixels to a given intensity value (*a*). C(i, j) ⇐ a
Switch images	Yes
Typical uses	*sci* and *zer* are both used extensively in a wide variety of situations
Remarks	*zer* is equivalent to *sci*(*0*)

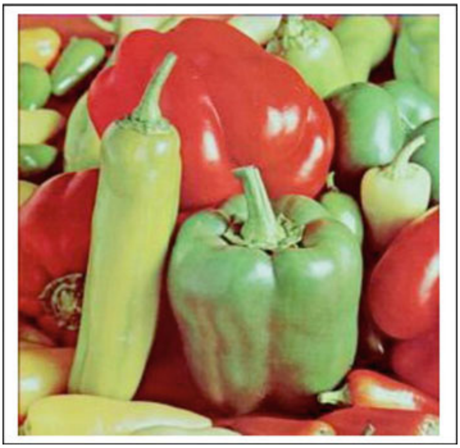
41.303 scp

Format	*scp*
Arguments	None
Output values	None
Image type	Colour
Type	Analysing colour images
Function	The user is invited to define a polygonal mask region within a colour image. Then, QT extracts the RGB values for all pixels within that region and plots them in 3D space
Switch images	Yes
Typical uses	Analysing colour images, particularly when building colour recognition systems
Remarks	See *shs* and *pcf*. The user can ‘fly around’ the cloud of points in the RGB scattergram, to obtain a variety of views and thereby gain an improved understanding of the structure of the cluster(s)

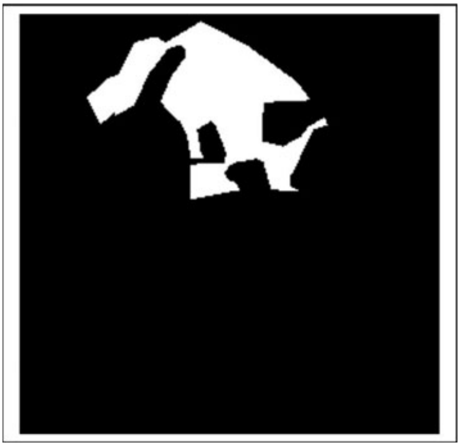
Illustration

Input image	peppers
File number	73
Command	*scp*

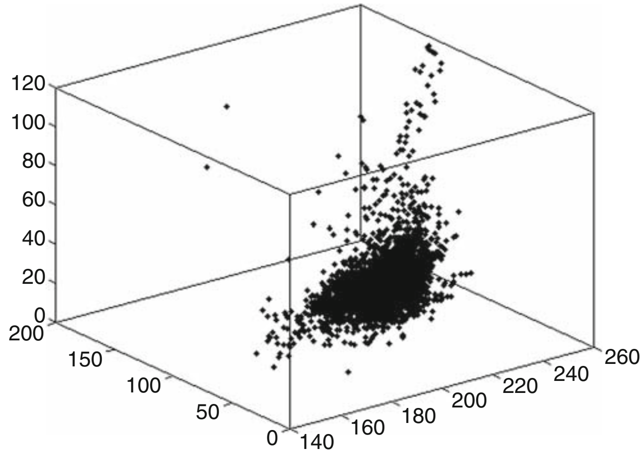
Left: *original*



Right: *polygonal mask drawn by the author*



Bottom: scattergram of RGB values found within the polygonal mask region



41.304 scr/usc

Format *scr*(*p*)
usc(*p*)

Arguments *p* New image size(after normalising to produce a square image)

Defaults 512

Output values None

Image type Grey scale

Type Image scrambling

Function *scr*: Scramble the input image by following the Hilbert curve
usc: Reverse the scrambling performed by *scr*

Switch images Yes but the picture displayed in the Alternate Image is reformatted to be square and of size $2^{\mathrm{m}} \times 2^{\mathrm{m}}$, where m is an integer and is equal to $\lceil log_2(p) \rceil$

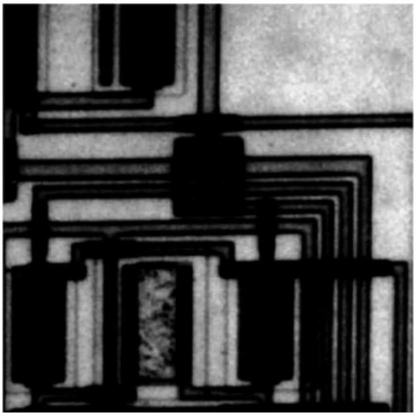
Remarks Also see *rev* and *hbt*. It is also possible to scramble the image using *usc* and then reverse it with *scr*. Notice that [*scr*, *rev*] is reversible, using [*rev*, *usc*]. Further scrambling options can be obtained using *lrt*, *tbt* and *yxt*. For example, the scrambling produced by [*scr*, *scr*, *yxt*, *lrt*, *scr*] can be reversed using [*usc*, *lrt*, *yxt*, *usc*, *usc*] and vice versa. Although not very secure, these coding methods will confuse casual observers. To encode a colour image, using a grey-scale operator, *scramble*, apply *pci(‘scramble’)*

Illustration

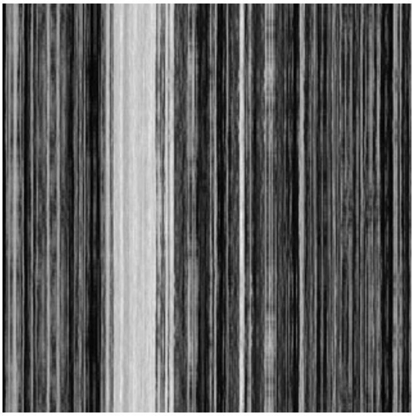
Input image Thick film circuit

File 136

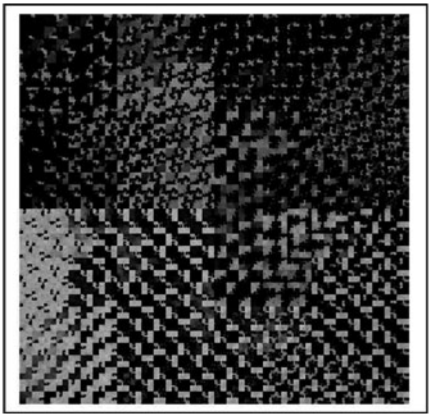
Top-left: original image



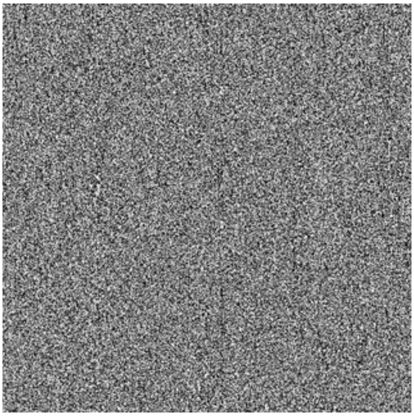
Top-right: after applying *scr*



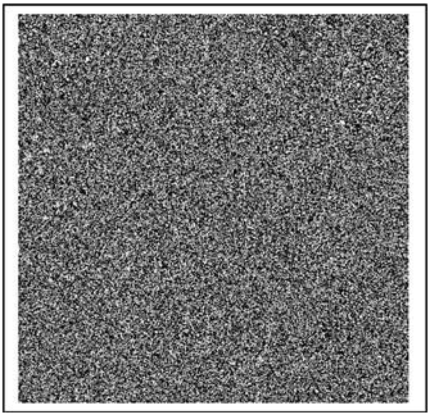
Centre-left: after applying *usc*



Centre-right: after applying *scr*, *rev*



Bottom-left: after *usc*, *rev*



41.305 scv

Format *scv*(*x*, *y*, *r*, *g*, *b*)

Arguments *x* *X*-address of the pixel to be accessed

y *Y*-address of the pixel to be accessed

r *R*-channel value

g *G*-channel value

b *B*-channel value

Defaults None

Output values None

Image type Any

Type Accessing individual pixels

Function Set the RGB values at point [*r*, *g*, *b*] in the current image to [*r*, *g*, *b*].

Switch images Yes

Typical uses Annotating images; placing points at critical features for display purposes. Testing image processing operators

Remarks Use *scv*, rather than accessing individual pixels of the current image directly, since it follows the QT convention that the x-address precedes the y-address. It also avoids the need to define the current image as *global* via the Command Window. Also see *ccv* and *siv*

41.306 sdm

Format $d = sdm(a, b, c)$

Arguments a Maximum X-shift

b Maximum Y-shift

c Incremental shift in the x and y directions

Defaults None

Output values d Grey-level spatial dependency matrix (SDM)

Image type Grey-scale image analysis

Type Global transform

Function Calculate the grey-level spatial dependency matrix (SDM). Steps:

(i) Let $S = \{[mc, nc] \mid 0 \leq mc < \mathrm{a}, 0 \leq nc < b\}$ (N.B. m and n are integers)

(ii) Choose an element $[X, Y]$ from S

(iii) Shift the image by an amount $[X, Y]$

(iv) Cross-correlate the shifted image with the original. The correlation score is held in $d(m, n)$

(v) Repeat steps (ii) – (iv) for all members of set S

Switch images Yes

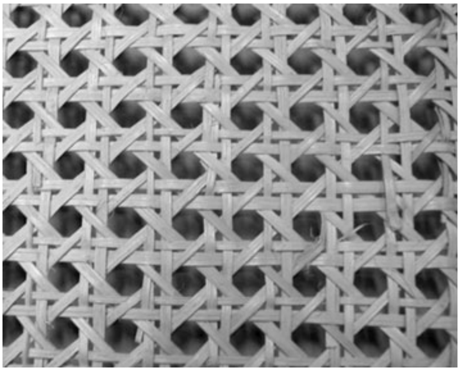
Typical use Describing texture and other images containing nearly periodic structures

Remarks In the illustration below, the periodic structure of the lattice is evident from the extended SDM. However, it is normal to calculate the SDM with image shifts that are smaller than the period. The SDM may be presented to a pattern classifier to obtain a decision about the nature/identity of the texture

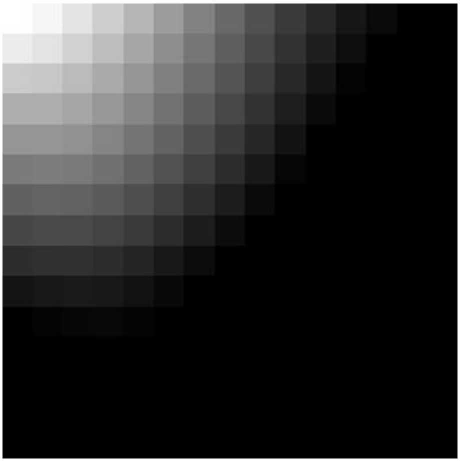
Illustration

Input image	Woven lattice of split cane
File number	239
Commands	*x* = *sdm*(*50*, *50*, *2*); *y* = *uint8*(*255∗x*) (This represents the SDM as an image)

Top: original image



Bottom-left: SDM represented as an image array



Bottom-right: Extended SDM (sampled) shows the periodic nature of the image

41.307 seb

Format [*x*, *y*, *z*] = *seb*(*s*)

Arguments *s* Scaling factor

Defaults *s* Chosen automatically, to maximise use of intensity scale

Output values *x* Vector containing the X-coordinates of edge pixels

y Vector containing the Y-coordinates of edge pixels

z Vector containing the chain code value for each edge pixel (See *fcc*)

Image type Binary

Type Blob transform

Function Shade the outer edge of a blob according to the distance travelled around its circumference. The starting point is the top-left-most, non-black pixel. The edge follower travels in a clock-wise direction. The edges of any lakes are ignored

Switch images Yes

Typical uses Analysing blob shape, by relating points of high curvature (corners detected using *cnr* or *hcd*) to circumferential position

Remarks A single blob can be processed using *seb*. If any other blobs are present, the result is an error. Other blobs and any lakes must analysed separately. Also see *fcc*

Illustration

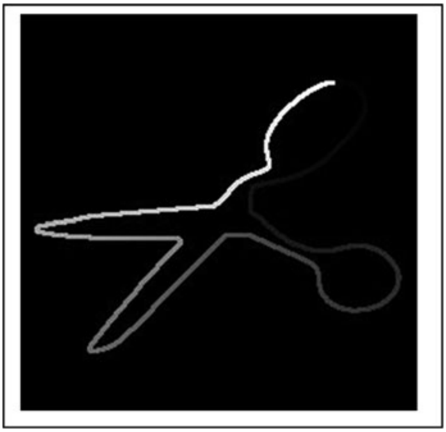
Input image Scissors

File number 49

Command *seb*, *lnb* (*ln*b broadens the edge contour, making it easier to see)

Left: original

Right: after processing



41.308 sed

Format	*sed*(*a*)	
Arguments	*a*	Function selector ($0 \leq a \leq 4$)
Defaults	*a*	0
Output values	None	
Image type	Grey	
Type	Edge detector	
Function	Sobel and similar edge detectors	

Definitions

sed(*0*)

$$C(i, j) = [\ |A(i-1, j-1) + 2A(a, j-1) + A(i+1, j-1) - A(i-1, j+1) + 2A(i, j+1) + A(i+1, j+1)| + |A(i-1, j-1) + 2A(a-1, j) + A(i-1, j+1) - A(i+1, j-1) + 2A(i+1, j) + A(i+1, j+1)|\ ]/6$$

sed(*1*)

$$C(i, j) = [\ |\ A(i-1, j-1) + A(a, j-1) + A(i+1, j-1) - A(i-1, j+1) + A(i, j+1) + A(i+1, j+1)\ | + |A(i-1, j-1) + A(a-1, j) + A(i-1, j+1) - A(i+1, j-1) + A(i+1, j) + A(i+1, j+1)\ |\ ]/4$$

sed(*2*)

$$C(i, j) = [\ |\ A(i-1, j-1) - A(i+1, j+1)\ | + |\ A(i-1, j+1) - A(i+1, j-1)\ |\ ]/2$$

sed(*3*)

$$C(i, j) = [\ |\ A(a, j-1) - A(i, j+1)\ | + |\ A(a-1, j) - A(i+1, j)\ |\ ]/2$$

sed(*4*)

$$C(i, j) = [\ |\ A(a, j-1) - A(i, j+1)| + |\ A(a-1, j) - A(i+1, j)| + |A(i-1, j-1) - A(i+1, j+1)\ | + |\ A(i-1, j+1) - A(i+1, j-1)\ |\ ]/4$$

Switch images Yes

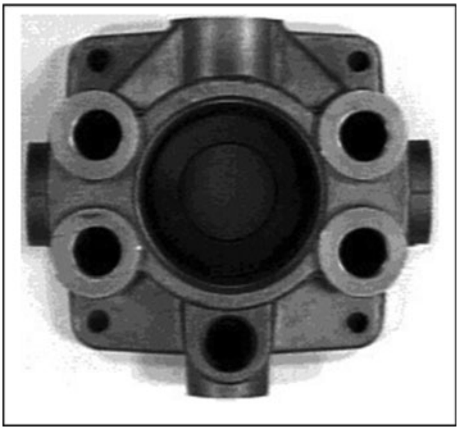
Typical uses Highlighting edges (i.e., regions of large intensity gradient)

Remarks There are many similar operators that differ slightly in their definition and performance. Others can be constructed using different weights for the *con* operator. Notice that line *47* (*adi*) could replaced by *mxi*, to define another set of edge detectors

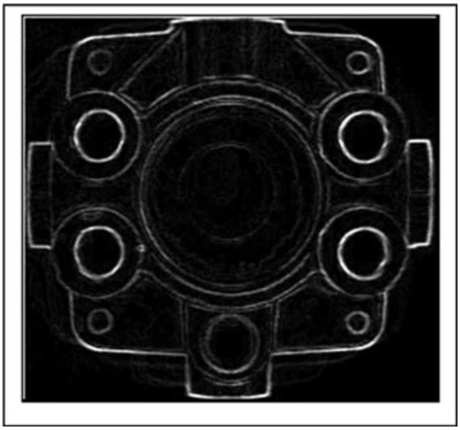
Illustration

Input image	Hydraulics component
File number	37
Command	*sed*(*0*), *enc*

Left: original



Right: after processing



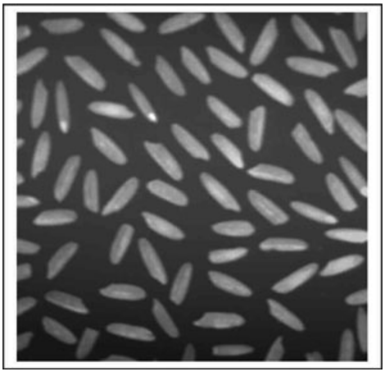
41.309 seg/rbk

Format	c = *seg*(*a*, *b*) *rbk*(*a*)	
Arguments	*a*	Size of structuring element for opening operation
	b	Increment/decrement for thresholding ($-1 \leq b \leq 1$) (*seg* only)
Defaults	*a*	15
	b	0 (seg only)
Output values	c	Threshold parameter (seg only)
Image type	Grey	
Type	Monadic	
Function	(i) Apply opening with a disc-like structuring element of size a (ii) Subtract the original image (iii) Enhance contrast (iv) Threshold using Otsu's method for calculating the threshold parameter (*seg* only)	
Switch images	Yes	
Typical uses	*seg* is used for image segmentation (i.e., isolating an objects of known size from the background	
Remarks	*seg* is far more reliable than *thr* for objects of known size. *rbk* is similar to *seg* but does not include thresholding	

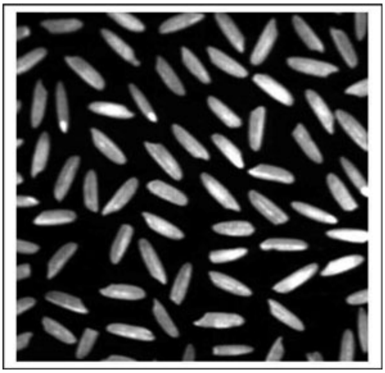
Illustration

Input image	Rice grains
File number	3
Command	*seg*(*15*, *0*) *and rbk*(*15*)

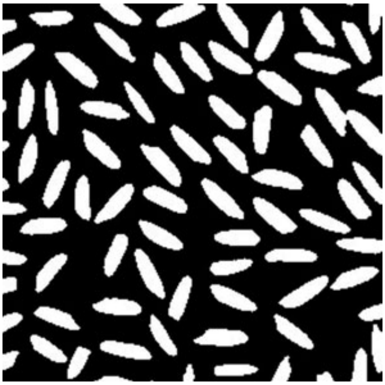
Top-left: original



Top-right: after *rbk*(*15*)



Bottom-left: after *seg*(*15*, *0*)



41.310 shk

Format *shk*

Arguments None

Defaults Not applicable

Output values None

Image type Binary

Type Blob transform

Switch Yes

Function Shrink each blob to a single white pixel. If it has *N* lakes ('holes'), the blob is reduced to a match-stick figure with *N* loops

Typical uses Inspecting granular materials. Inspecting sheet materials with textured surfaces or randomly distributed spots (e.g., certain floor coverings)

Remarks *shk* preserves the Euler number. We can reduce all blobs to a single point using *blf*, *shk*. If the blob contains no lakes, the point produced by *shk* lies within that blob. However, this is not guaranteed if we fill the lakes. Hence, applying *shk* to a circular ring (O), generates a point within the lake, rather than the white area.

Illustration

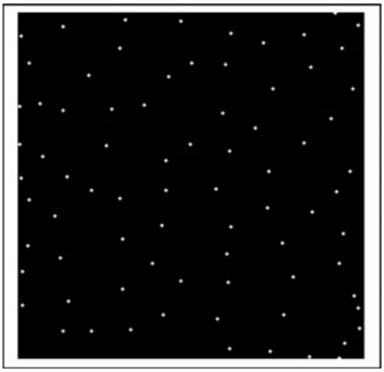
Input images Binary image derived from rice grains (image file 4)
Scissors (image file 164)

Image generation Rice grains: *rni*(*4*), *enc*, *thr*, *dil*, *ero*
Scissors: *thr* (needed to avoid artifacts due to JPEG coding)

Commands Rice grains: *shk*, *dil* (*dil* was included simply to make the spots visible)
Scissors: *shk*

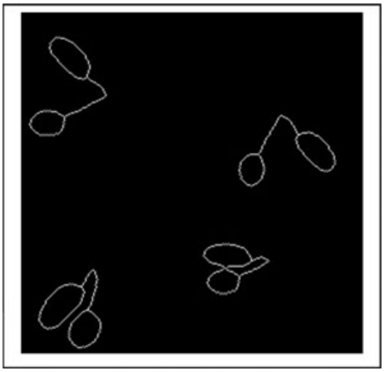
Left: binary rice grains image

Top-right: binary rice grains after *shk*, *dil*



Bottom-left: scissors image

Bottom-right: scissors image after *shk*



41.311 shp

See *con*

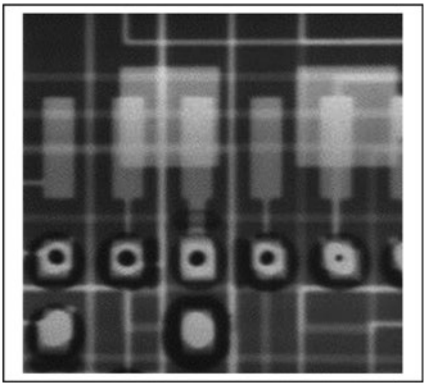
41.312 shr

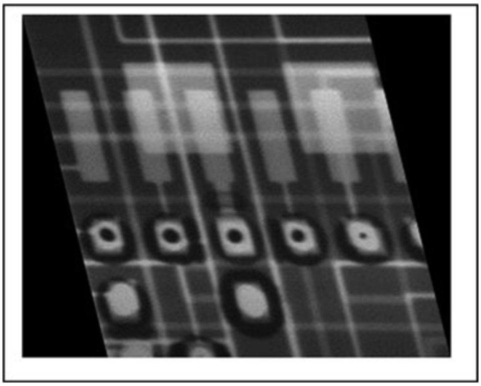
Format	*shr*(*a*)
Arguments	*a* Warping factor ($0 \leq a \leq 1.0$)
Defaults	None
Output values	None
Image type	Grey
Type	Image warping
Function	A shear transformation is applied to the image. An upright rectangle is mapped into a parallelogram
Switch images	Yes
Typical uses	Correcting image distortion due to motion during scanning
Remarks	Notice that the image becomes larger (wider) as a result of applying this operator. A command sequence of the form *shr*(*a*); $b = -a$; *filter-2-dimensional*; *shr*(*b*) performs filtering along two non-orthogonal axes

Illustration

Input image	Printed circuit board
File number	135
Command	*shr*(*0.25*)

Left: original (size: 256∗256) *Right*: after processing (size: 256∗320)

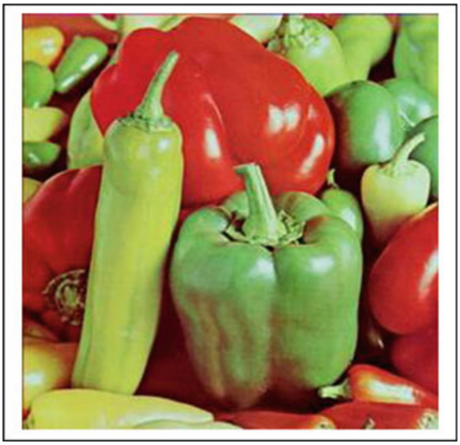




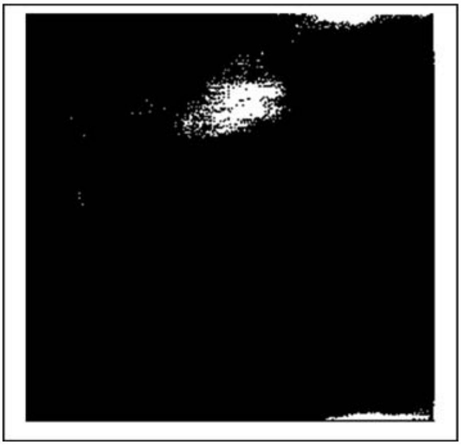
41.313 shs

Format	*shs*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Colour
Type	Colour image processing
Switch	Yes
Function	Plot the scattergram of hue (vertical axis) versus saturation (horizontal axis)
Typical uses	Analysis of clustering of colour measurements. Data reduction of images containing blocks of nearly constant, highly saturated colours, prior to applying a programmable colour filter (*pcf*)
Remarks	Also see *pcf*
Illustration	
Input image	Peppers
File number	73

Top-left: original colour image.

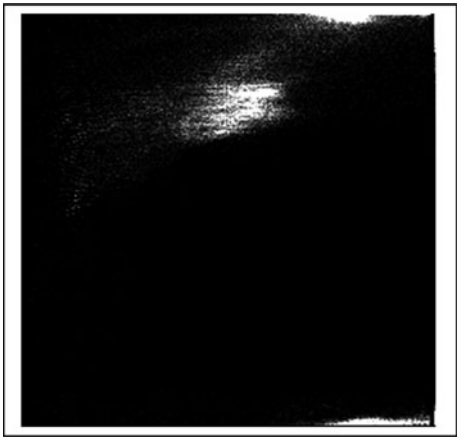


Top-right: after *shs*

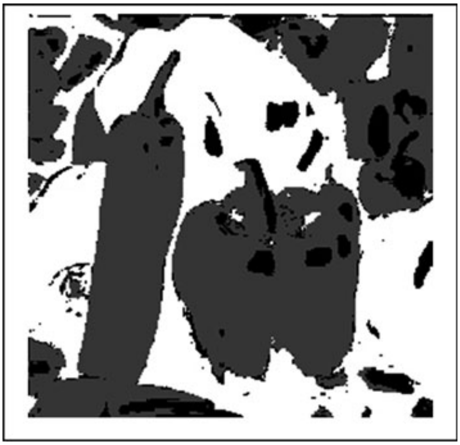


Centre-left: after *shs, enc, thr*

Centre-right: after *shs, enc, thr, kgr(25), dil(15), ndo, enc, hil(170,255,255), dil(8))*.
This is the alternate image forming the colour look-up table used by *pcf* in the example below.



Bottom-left: output of *pcf*

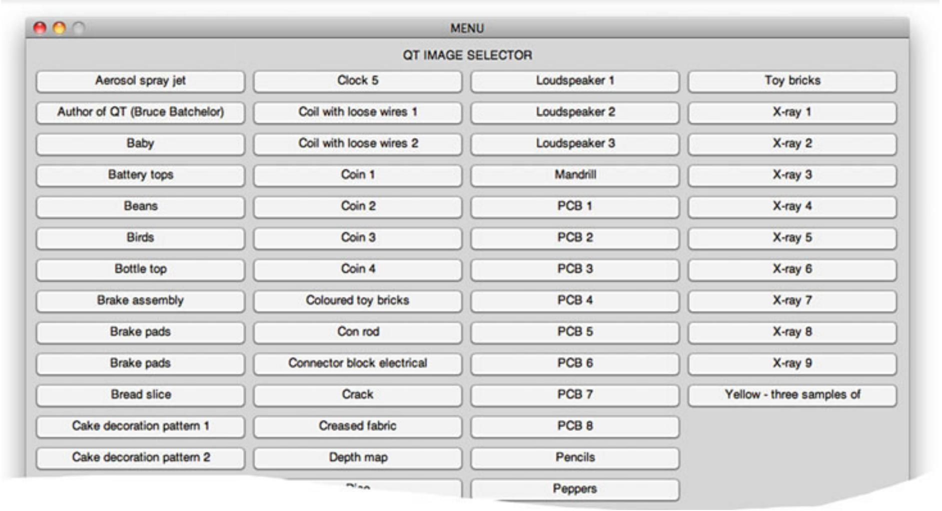


41.314 sia

Format	*sia*(*a*)	
Arguments	*a*	Name of the folder where the new images are to be found
Default value	*a*	The string QT_images
Output values	None	
Image type	Binary, grey or colour	
Type	QT maintenance/extension	
Function	Save all image files in the named folder (*a*) to the Image Archive: QT_images The new image files are in TIFF format	
Switch images	No	
Typical uses	Extending the Image Archive quickly by adding a new batch of image files	
Remarks	Also see *aia*	

41.315 sim

Format	*sim*
Arguments	None
Output values	None
Type	User aid
Switch images	No
Function	Display a menu showing the names of popular images. The user clicks on one of these, which is then read into the Current Image and displayed in Figure 1. The most recent image captured from the camera using *frz*
Remarks	Also see *did*



41.316 sir

Format	*sir*(*x*, *y*, *w*, *h*, *z*)	
Arguments	*x*	X-coordinate of top-left corner of the rectangle
	y	Y-coordinate of top-left corner of the rectangle
	w	Width of the rectangle
	h	Height of the rectangle
	z	Intensity value for pixels within the rectangle
Defaults	*z*	0
Output values	None	
Type	Drawing	
Switch images	Yes	
Function	Set all pixels within the specified rectangle to intensity *z*	
Remarks	Also see *rea*, *rni*, *rci*, *gwi* and *wri*	

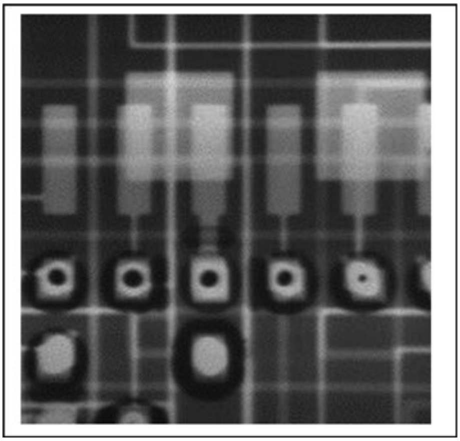
41.317 sit

Format	*sit*
Arguments	None
Defaults	None
Output values	None
Image type	Grey scale
Type	Monadic
Function	Apply a sigmoid intensity mapping function
Switch images	Yes
Typical uses	Enhance image contrast for display of further processing
Remarks	The central part of the intensity scale is stretched; the ends are compressed. *sit* can be used in conjunction with other monadic functions (e.g., *enc*, *heq*, *sqr*, *sqt*, *din*, *neg*, *alg*, *nlg*), to produce other useful mapping operators

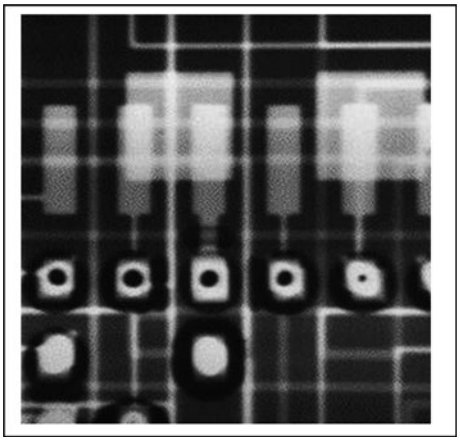
Illustration

Input image	X-ray, PCB
File number	135
Command	*sit*

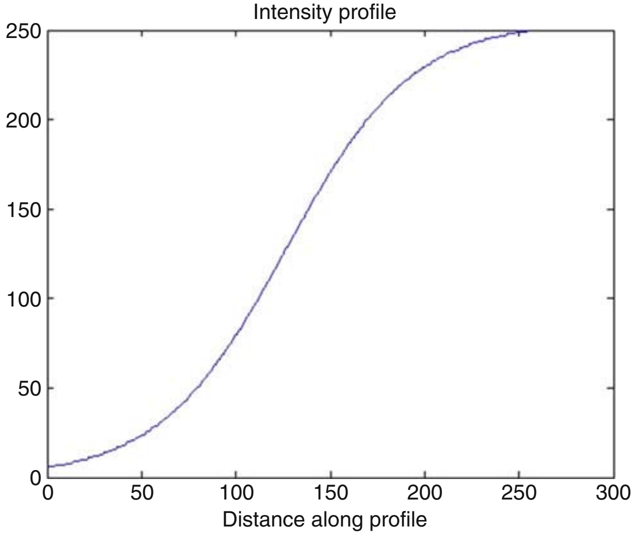
Left: Original image



Top-right: *after processing*



Bottom: Intensity mapping function for *sit*



41.318 siv

Format	d = *siv*(*a*, *b*, *c*)		
Arguments	*a*	X-address of the pixel to be accessed	
	b	Y-address of the pixel to be accessed	
	c	New intensity value	
Defaults	None		
Output values	*d*	0	Inputs *[a, b]* within range
	d	1	Inputs *[a, b]* out of range
Image type	Any		
Type	Accessing individual pixels		
Function	Set the intensity at point [*a*, *b*] in the current image to level *c*		
Switch images	Yes		
Typical uses	Annotating images; placing points at critical features for display purposes. Testing image processing operators		
Remarks	Use *siv*, rather than accessing individual pixels of the current image directly, since it follows the QT convention that the x-address precedes the y-address. It also avoids the need to define the current image as *global* via the Command Window		

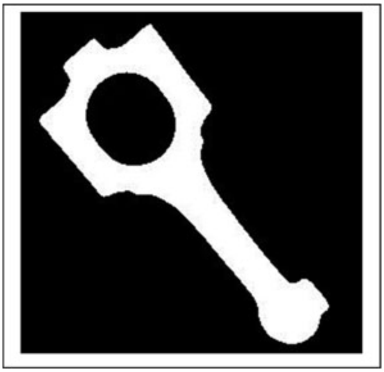
41.319 ske

Format	*ske*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Binary
Type	Blob transform
Function	Skeleton of a single blob in a binary image
Switch images	Yes
Typical uses	Blob shape analysis
Remarks	The skeleton is a connected figure that approximates the medial axis; each point in the skeleton is equidistant from at least two edge points

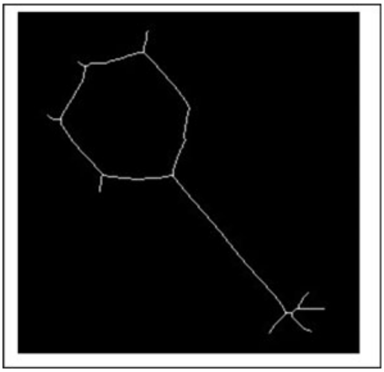
Illustration

Input image	Con-rod
File number	47
Command	*thr*, *ske* (N.B. *thr* avoids the artifacts created by storing a binary image in JPEG format)

Left: original



Right: after processing



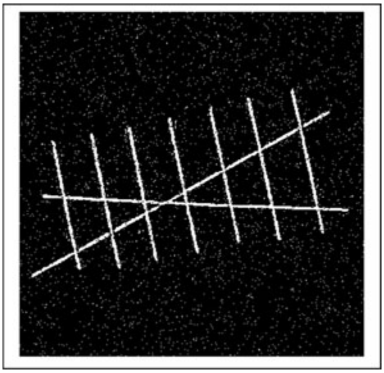
41.320 slf

Format	a = *slf*
Arguments	None
Defaults	None
Output values	*a* Orientation of strongest linear feature
Image type	Binary
Type	Global transform
Switch	Yes
Function	Strongest linear feature
Definition	The Radon transform (*rdn*) is computed first. The resulting image (RT) is then processed to find the brightest spot. Its X-coordinate indicates the orientation of the strongest linear feature in the input image. The input image is then rotated so that the strongest linear feature is vertical
Switch images	Yes
Typical use	Detecting linear streaks in noisy images
Remarks	It is a simple matter to identify other lines that are nearly parallel to the strongest linear features; apply *cin*, *rsb*, If there are other lines parallel to the strongest linear feature, these will appear as bright streaks in the output image (Also see *rdn* and *irt*)

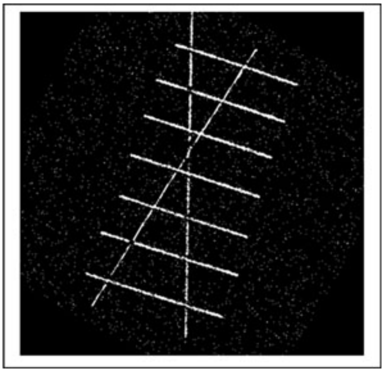
Illustration

Input image	Lines, with pre-processing to create a noisy binary image: *crk*(*3*), *enc*, *thr*, *spn*(*0.05*)
File number	160
Commands	*a* = *slf*; *cin*, *rsb*, *tur*(−*a*), *enc*, *thr*(*255*), *big*, *thn*

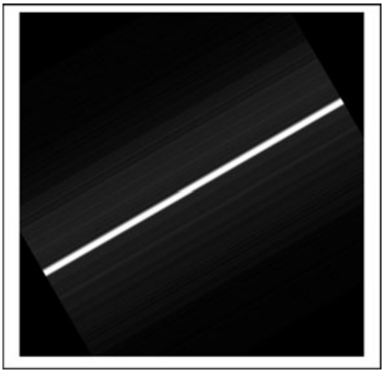
Left: original



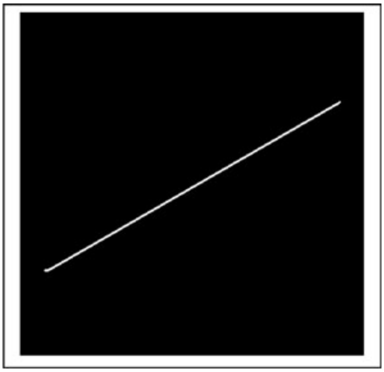
Right: immediately after *slf*



Bottom-left: just after *tur*(−*a*)



Bottom-right: final result, after *thn*



41.321 sly

See *bpa*.

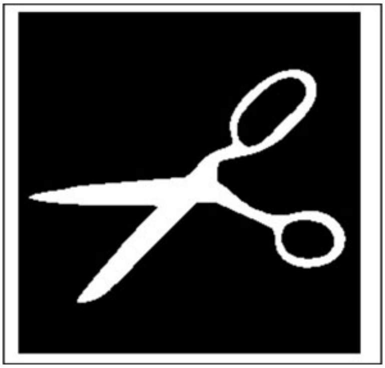
41.322 sme

Format	*sme*(*u*)
Arguments	*u* Separation parameter for edge-follower 'bugs'
Defaults	*u* 10
Output values	None
Image type	Binary image, single blob, no lakes
Type	Binary image analysis
Switch	Yes
Function	Smooth the edge of the blob
Switch images	Yes
Typical uses	Noise reduction, prior to analysing blob geometry
Remarks	Convex and concave corners can be detected separately, by subtracting the output from the original image

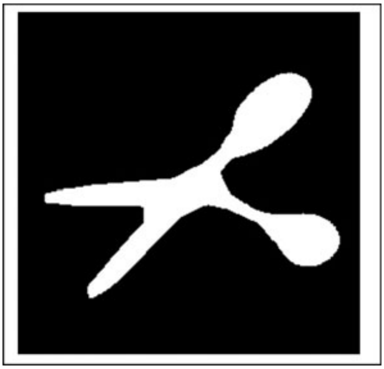
Illustration

Input image	*rni*(*49*), *thr*, *blf*
Commands	(i) *sme* (ii) *sme*, *rei*('*original*'), *sub*, *neg*

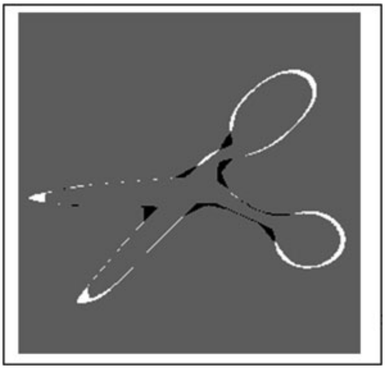
Left: original image



Top-right: *sme*



Bottom-left: *sme*, *rei*('*original*'), *sub*, *neg*



41.323 snb

Format	*snb*(*a*)	
Arguments	*a*	Size of processing window
Defaults	*a*	3
Output values	None	
Image type	Grey scale or binary	
Type	Non-linear filter	
Switch	Yes	
Function	Smallest (i.e., darkest) neighbour in an *a* × *a* window	
Typical use	Noise reduction. Component of a number of other operators (e.g., *snb*, *sub* is a highly effective edge detector.)	
Remarks	*snb*(*a*) is equivalent to *ero*(*a*, *1*). Also see *lnb*	

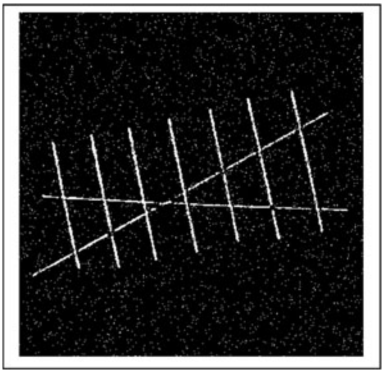
41.324 sob

Format	*[x, y] = sob*	
Arguments	None	
Output values	*x*	X-coordinate of the point defined by the user with the cursor
	y	Y-coordinate of the point defined by the user with the cursor
Image type	Binary	
Type	Miscellaneous	
Function	Select one 8-connected blob with the cursor	
Switch images	Yes	
Typical uses	Interactive selection of blobs in a complex image, prior to more detailed analysis	
Remarks	*cob* allows more than one blob to be selected; *sob* allows only one but does return the point defined by the user	

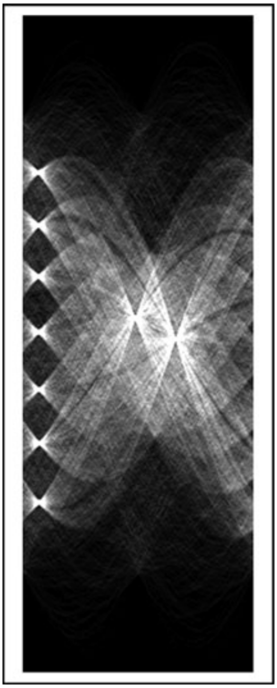
41.325 irt

Format	*irt*
Arguments	None
Defaults	None
Output values	None
Image type	Grey scale or binary
Type	Global transform
Switch	Yes
Function	Inverse Radon transform
Switch images	Yes
Typical use	Isolating individual linear streaks in noisy images
Remarks	Normally, *irt* is used in conjunction with *rdn* (Radon transform). The example shown below illustrates how a group of nearly vertical, parallel lines can be identified. The Radon transform image (RT) is first weighted by multiplying it by the intensity wedge (*wgx*, *neg*). The contrast is then adjusted by applying *sqr*. (This reduces the number of 'noise' points in the final image.) Finally, *irt* is applied. Also see *rdn*
Illustration	
Input image	Lines
File number	158
Command	*rdn*, *wri*, *wgx*, *neg*, *rei*, *mul*, *sqr*, *irt*

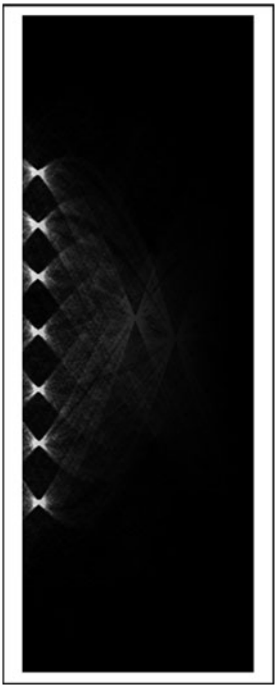
Top-left: *original*



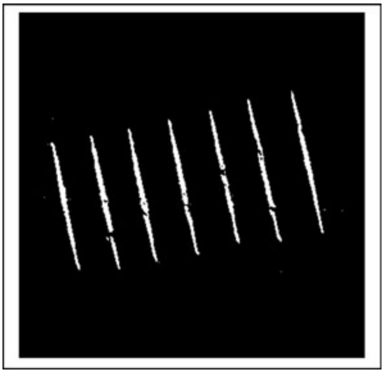
Centre: just after *rdn*



Bottom-left:just before *irt*



Bottom-right: after *irt*



41.326 spi

Format *spi*(*A*, *B*, *T*)

Arguments *A* Background process (QT command sequence)

B Interjected process (QT command sequence)

T Timer interval (seconds)

Defaults None

Output values None

Type Process control

Function Imagine process *A* is running. This might contain an infinite repeating loop

Every *T* seconds, *A* is suspended and process *B* is executed

When *B* terminates, *A* is restored and continues running, as if nothing had happened

Typical uses Occasional inspection by stealing time from an on-going process that can tolerate an occasional loss of a processing cycle. For example, *A* may be a nearly continuous product-inspection process and *B* a safety check that need only be performed once every few seconds, or even minutes

Stopping To stop both background and interjected processes, perform the following operations:

(i) Click in MATLAB's Command Window

(ii) Press CNTRL/C to terminate the background process (*A*)

(iii) Enter *q* (lower case *Q*) and press RETURN, to terminate the interjected process (B)

If step (iii) is not performed, the timer will still fire periodically and cause the interjected process to operate during keyboard interaction: very confusing!

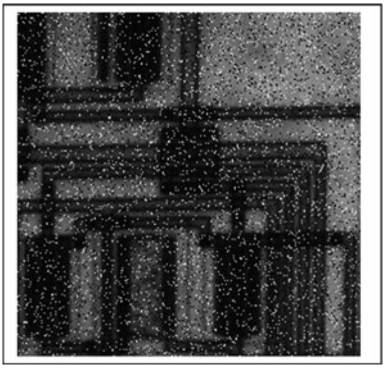
Remarks Normally, the execution time for process *B* is less than *T* seconds. It is important that processes *A* and *B* use separate disc image files, otherwise they will interfere with one another

Example of use *spi*([‘*n* = *round*(*100∗rand*(*1*, *1*));*rni*(*n*); *avg*; *psc*’], [‘*grb*(*6*); *gli*;, *psc*’], *10*)

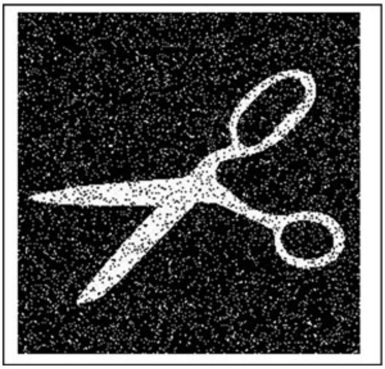
41.327 spn

Format	*spn*(*a*)
Arguments	*a* Proportion of the pixels that are changed ($0 \leq a \leq 1$)
Defaults	*a* 0.02. (1% of the pixels are set to black and 1% are set to white)
Output values	None
Image type	Grey scale or binary
Type	Drawing and image generation
Switch	Yes
Function	Add 'salt-and-pepper' noise to an image
Definition	A proportion *a*/2 of the pixels in the current image are selected at random and are set to black. A similar proportion are randomly selected and set to white
Switch images	Yes
Typical uses	Testing image filtering functions
Remarks	Also see *gsn*
Illustration	
Input images	Circuit (grey-scale) and scissors (binary)
File numbers	136 and 49
Commands	*spn*(*0.1*) for the grey-scale image (circuit, 136) *spn*(*0.2*) for the binary image (scissors, 49)

Left: processing a grey-scale image



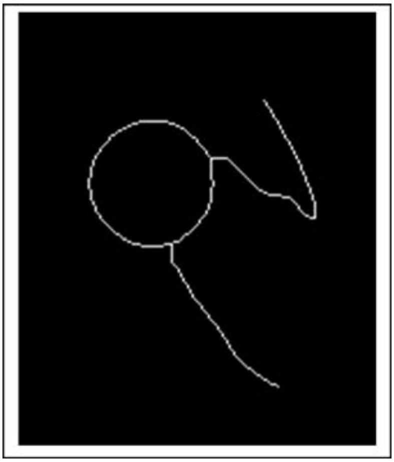
Right: processing a binary image (49)



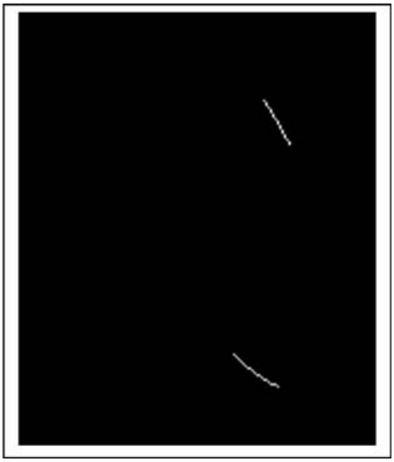
41.328 spr

Format	*spr*(*a*)
Arguments	*a* Number of end points to be removed
Defaults	*a* 1
Output values	None
Image type	Binary
Type	Morphology
Function	Remove all white pixels that have only one white 8-neighbour. (These are the end points of skeleton figures created by *ske* or *thn*). This is repeated recursively *a* times
Switch images	Yes
Typical uses	Part of a routine for analysing binary images arising in robot vision applications, for example grasping the ends of flying leads (e.g., images no. 18 and 19). First, the image is processed to create a skeleton and then *spr* is applied. The resulting short line-features are then analysed (using *lmb*), to find their ends. Then, the position coordinates and orientation of each wire end can be calculated. These values are then sent to a robot, enabling it to grasp the ends of the wires (one at a time)
Remarks	Also see *ske*, *thn*, *lmb*
Illustration	
Input images	Skeletons derived from images 19 (coil with flying leads) and 46 (silhouettes of four metal components)
Command	*spr(20) and spr(15)*

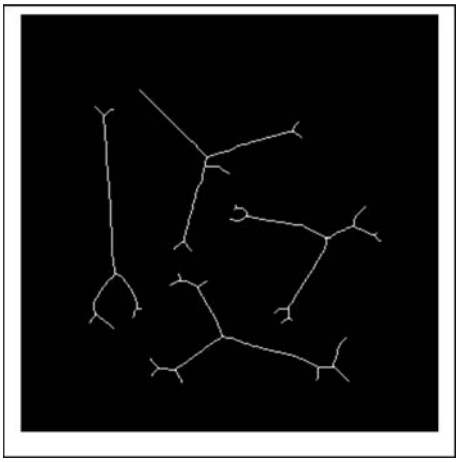
Top-left: skeleton derived from image no. 19



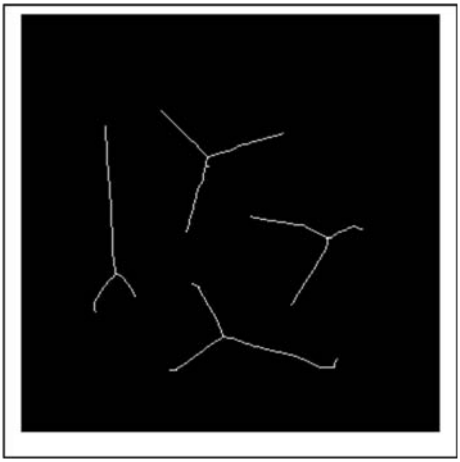
Top-right: after *spr*(*20*), exr



Bottom-left: skeleton derived from image no. 46



Bottom-right: after *spr*(*15*)



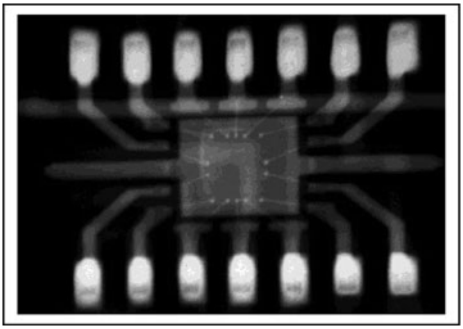
41.329 sqi

Format	*sqi*(*a*)	
Arguments	*a*	Intensity value for the padding area to be added to the image
Defaults	a	0 (zero)
Output values	None	
Image type	Grey-scale or Binary	
Type	Image shaping	
Function	Pad the image to make it square. The new rectangular padding area is added to the right/below the original image, depending on its aspect ratio	
Switch images	Yes	
Typical uses	Certain QT operators/sequences require a square image to work properly (e.g., *tur* and *yxt*, *adi*)	
Remarks	Also see *pad*	

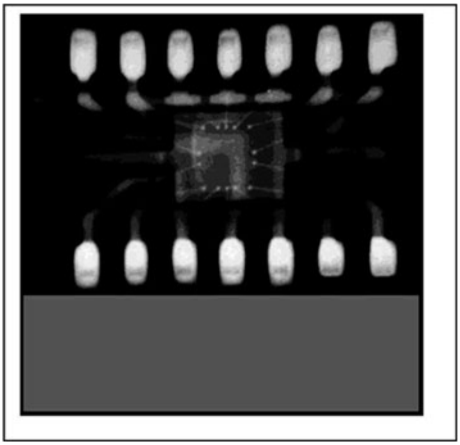
Illustration

Input image	Integrated circuit x-ray
File number	113
Command	*sqi(128)*

Left: original image



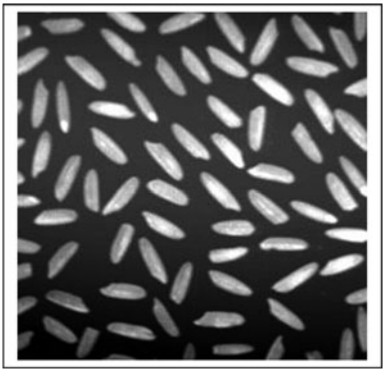
Right: after processing



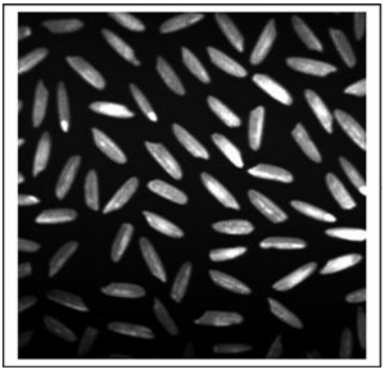
41.330 sqr

Format	*sqr*
Arguments	None
Defaults	None
Output values	None
Image type	Grey
Type	Monadic
Function	Square of intensities in the current image
Definition	$C \Leftarrow A^2/255$
Switch images	Yes
Typical uses	Used to improve contrast in bright regions of the image
Remarks	Also see *alg*, *nlg and sqt*
Illustration	
Input image	Rice grains after contrast enhancement (*enc*)
File number	3
Command	*sqr*

Left: original



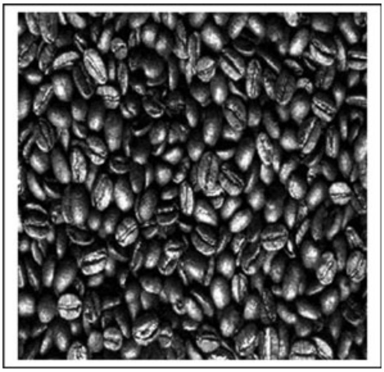
Right: after processing



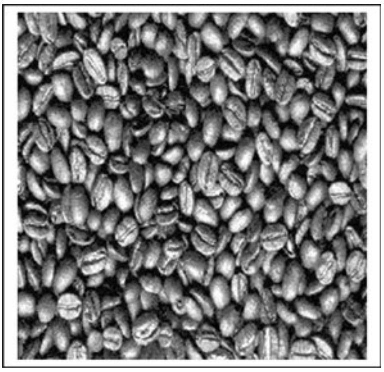
41.331 sqt

Format	*sqt*
Arguments	None
Defaults	None
Output values	None
Image type	Grey
Type	Monadic
Function	Square root of intensities in the current image
Definition	$C \Leftarrow \surd(255 * A)$
Switch images	Yes
Typical uses	Used to improve contrast in dark regions of the image
Remarks	Also see *alg*, *nlg* and *sqr*
Illustration	
Input image	Coffee beans after contrast enhancement (*enc*)
File number	4
Command	*sqt*

Left: original



Right: after processing



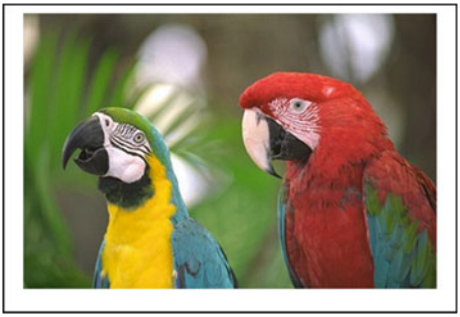
41.332 srgb

Format	*srgb*(*a*, *b*)
Arguments	*a* Colour channel (plotted along the horizontal axis) *b* Colour channel (plotted along the vertical axis)
Defaults	Not applicable
Output values	None
Image type	Colour
Type	Colour image analysis
Switch	Yes
Function	2-dimensional colour scattergram
Switch images	Yes
Typical uses	Analysing the structure of a colour image
Remarks	The scattergram is displayed in the current image with 256 × 256 pixels. The input parameters define the colour channels to be plotted, according to the following scheme: 1. *R*-channel ('red') 2. *G*-channel ('green') 3. *B-channel* ('blue') This function is most useful if there are strong clustering of colours; *srgb* works particularly well with 'block colours' (printing terminology, refers to colours that are discrete that do not blend with one another)

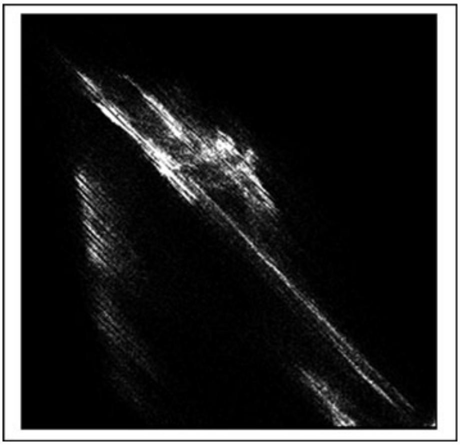
Illustration

Input image	Birds
File number	144
Commands	*srgb*(*1*, *2*)/*srgb*(*2*, *3*)/*srgb*(*3*, *2*)

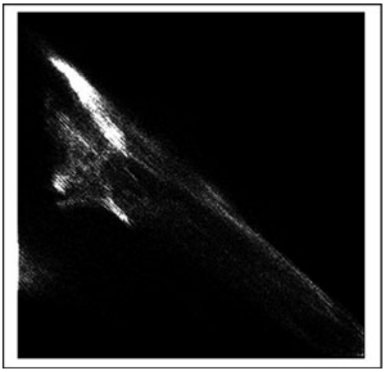
Top-left: original colour image



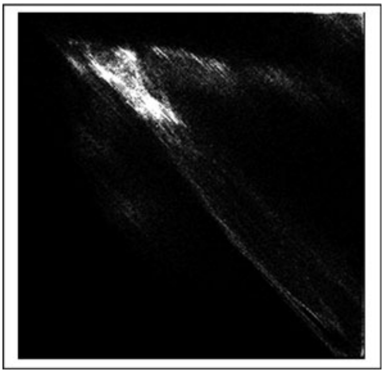
Right: *srgb*(*1*, *2*) Horizontal axis: *R*-channel Vertical axis: *G*-channel



Bottom-left: *srgb*(*2*, *3*) Horizontal axis: *G*-channel Vertical axis: *B*-channel



Bottom-right: *srgb*(*3*, *1*) Horizontal axis: *B*-channel Vertical axis: *R*-channel



41.333 ssk

Format	*ssk*
Arguments	None
Output values	None
Image type	Any
Type	Stack operator
Function	Part of the image stack is displayed in Figure 2. The top image on the stack is labelled *0*. Other images are labelled *1*, *2*, *3*, ... The previous image at the top of the stack is labelled ‘−*1*’. (It is possible to recover this image as the stack is actually implemented as a circular data structure)
Switch images	No
Remarks	Also see *push*, *pop*, *rsk*, *csk*

41.334 startup

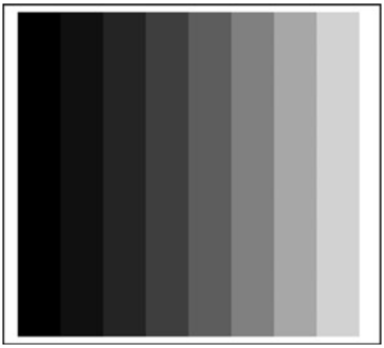
Format	*startup*	
Arguments	None	
Output values	None	
Image type	Not relevant	
Type	Not relevant	
Function	*startup*	Automatically initialises QT. May be run manually at other times
	qtstart	Manually initialises QT in versions that do not have automatic initialisation
Switch images	No	
Remarks	MATLAB automatically runs the M-file *startup.m* immediately after loading. QT uses this feature for initialisation, before any user input is accepted. The global variable *qt_control_vector*(*1*) controls the display of results in functions such as *avg*, *cgr*, *cwp*, *cbl*, *gli*, etc. The stand-alone version of QT allows the display of results from these functions, while the web version does not (To do so, would create a very messy display)	

41.335 stp

Format	*stp*(*a*)	
Arguments	*a*	Determines the number of steps ($1 \leq a \leq 8$. See *sca*)
Defaults	*a*	3
Output values	None	
Type	Drawing and image generation	
Switch	Yes	
Function	Draw a series of intensity steps; intensity varies across the image	
Switch images	Yes	
Typical uses	Testing prototype image processing functions. Generating other test patterns	
Remarks	This function is particularly useful for setting up a display monitor, so that the contrast is subjectively uniform over whole image	

Illustration

Command *stp*(*3*)



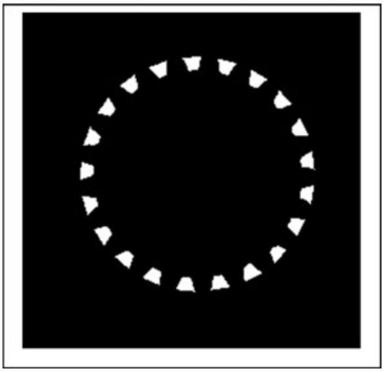
41.336 str

Format	*str*
Arguments	*None*
Output values	*None*
Image type	Binary
Type	Binary image analysis
Function	Draw a straight line from the overall centroid (i.e., considering all blobs) and the centroid of each individual blob
Switch images	Yes
Remarks	*str* is used in *chu_all* to connect disjoint blobs together so that *chu* can be applied to find the convex hull of the whole assembly. In addition, *str*, *mxi*, *nxy* can be used as an approximate way to implement *nxy_all*. Similarly for *npo*

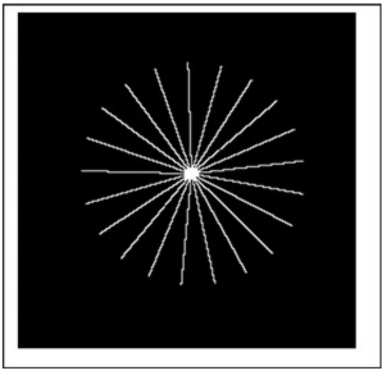
Illustration

Input generation	*rni*(*171*), *enc*, *thr*, *neg*, *big*, *blf*, *chu*, *exr*, *ero*
Command	*str*

Left: original



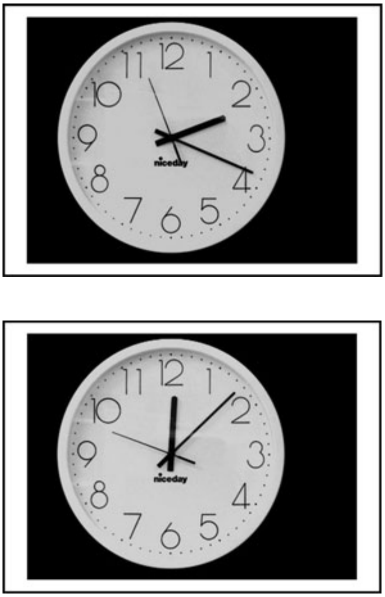
Right: after processing



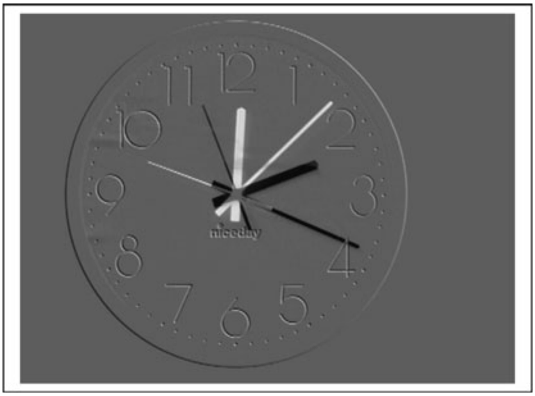
41.337 sub

Format	*sub*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey
Type	Dyadic
Function	Subtract the intensities in the current and alternate images
Definition	C ⇐ (255 + A − B)/2
Switch images	Yes
Typical uses	High-pass filtering; subtracting the filtered (blurred) and original images Finding differences between two images derived from the same camera, under identical lighting conditions
Remarks	*sub* is a component of numerous algorithms
Illustration	
Input image	Clock 1 and Clock 2
File number	142 and 143
Command	*sub*

Left: two originals



Right: after processing



41.338 swi

Format	*swi*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Any
Type	Miscellaneous
Function	Interchange the current and alternate images

41.339 tbt

Format	*tbt*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey scale or binary
Type	Affine transform
Switch	Yes
Function	Invert the vertical axis
Definition	$C(i, j) \Leftarrow A(i, h + 1 - j)$, where h is the picture height
Switch images	Yes

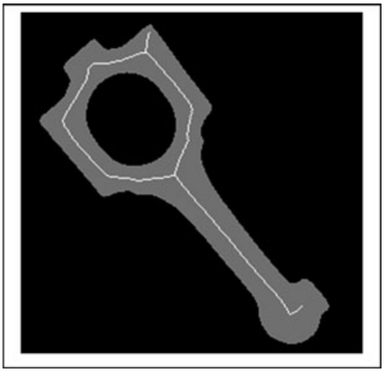
41.340 tha – thh

Formats	*tha*(*a*) *thb*(*a*, *b*, *c*) *thc*(*a*) *thd*(*a*) *the* *thf* *thg* *thh*
Arguments	See below
Defaults	See below
Output values	None
Image type	Grey
Type	Monadic
Function	Threshold at some level derived by analysing the image. Functions *tha*, *thb*, *thc* and *thd* require additional parameters (e.g., offset or bias values), supplied by the user
Definition	See below
Switch images	Yes
Typical uses	Image segmentation (i.e., isolating an object silhouette from its background)
Remarks	This family of operators all use the function *thr*, calculating the threshold parameter automatically or semi-automatically. Commands *tha*(*0.5*) *and thb* provide useful ways to make thresholding insensitive to lighting variations, when processing images obtained by back-lighting. Alphabetic naming of these functions was adopted, since it is difficult to define their functions exactly in a few words. See program listings for more details.

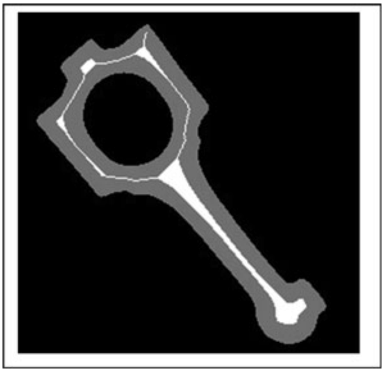
41.341 thn

Format	*thn*(*a*)
Arguments	*a* Number of times the thinning operation is applied
Defaults	Indefinitely large
Output values	None
Image type	Binary
Type	Blob transform
Function	*thn(a)* thins each blob in a binary image *thn* generates a match-stick figure (‘skeleton’)
Switch images	Yes
Typical uses	Blob shape analysis
Remarks	*thn* creates a match-stick figure that is usually cleaner (i.e., has fewer small spurs) than *ske*. *thn* creates a skeleton for each blob in the image, whereas *ske* draws only one. *thn*(*a*) may produce partially thinned blobs
Illustration	
Input image	Con-rod
File number	47
Command	*Left*: *thn*, *adi* *Right*: *thn*(*10*), *adi*

Left: after *thn*, *adi*



Right: after *thn*(*10*), *adi*



41.342 thr

Format *thr*(*a*, *b*)

Arguments *a* Lower intensity threshold limit

b Upper intensity threshold limit

Defaults *a* 128

b 255

Output values None

Image type Grey

Type Monadic

Function Threshold the input image. All pixels whose intensities lie within the range *[a, b]* are mapped to white (255). All other are set to black (0)

Switch images Yes

Typical uses Image segmentation (i.e., isolating an object silhouette from its background). Reducing data content in an image, to make subsequent analysis easier

Remarks To novice users, thresholding is an obvious function to use in many applications. However, it is sensitive to lighting variations and will not produce reliable image segmentation in any but the simplest applications. For this reason, *thr* is almost always used in conjunction with a suitable filter, edge detector, or an intensity-normalisation function, such as *enc* or *heq*

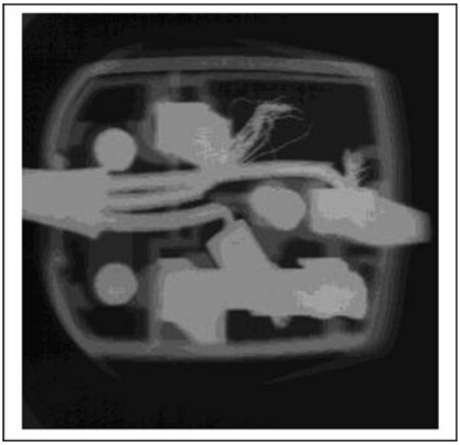
Illustration

Input image UK power plug

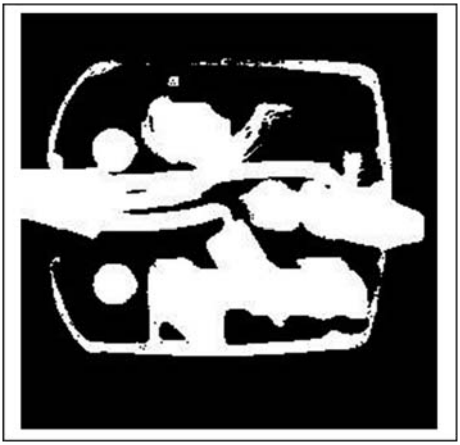
File number 112

Command *thr*(*115*)

Left: original



Right: after processing



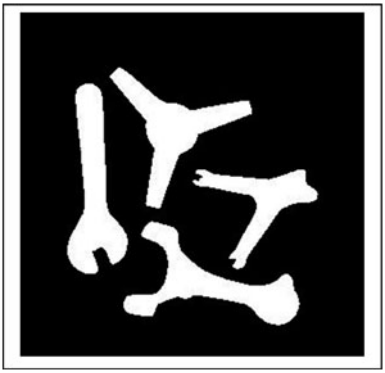
41.343 tim

Format	$b = tim(a)$
Arguments	*a* Function call
Defaults	None
Output values	*b* Execution time
Image type	Any
Type	Miscellaneous
Switch	Possible; depends on function specified by *a*
Function	Function timer
Definition	Time the execution of the specified function *a*. (Notice that the function to be timed is enclosed between single quotes, as shown in the example below)
Typical uses	Program analysis
Remarks	Example of use: *b = tim('maj(20)')* Output: *Execution time for function call 'maj(20)' was 498 ms*

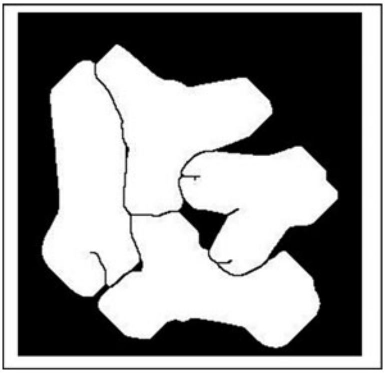
41.344 tkn

Format	*tkn*(*a*)
Arguments	*a* Number of times the (dilation) operation is repeated
Defaults	*a* 1
Output values	None
Image type	Binary
Type	Blob transform
Switch	Yes
Function	'Thicken' objects in a binary image; add pixels to the exterior, until doing so would result in previously unconnected objects being 8-connected
Switch images	Yes
Typical uses	Blob-shape analysis
Remarks	Unlike *dil*, this operation preserves the Euler number. Also see *gft* and *wsh*. Execution can take a long time for large values of *a* (Typically $a > 10$)
Illustration	
Input image	Four metal parts, after thresholding
File number	46
Command	*thr*, *tkn*(*20*) (Thresholding eliminates artifacts due to JPEG coding)

Top-left: original



Top-right: *after processing*



41.345 tlp/trp/rtp/rbp/brp/blp/lbp/ltp

Format [*x*, *y*] = *tlp*

Arguments None

Defaults Not applicable

Output values x X-coordinate of the top-left-most white point (*tlp*)

y Y-coordinate of the top-left-most white point (*tlp*)

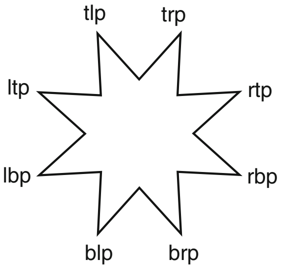
Image type Binary

Type Image analysis

Switch No

Function Use these functions to find the XY-coordinates of the other extremal points as follows:

tlp	Top-left
trp	Top-right
rtp	Right-top
rbp	Right-bottom
brp	Bottom-right
blp	Bottom-left
lbp	Left-bottom
ltp	Left-top



Typical uses Locating binary objects for further analysis

Remarks The other functions are defined in a similar way to *tlp*

41.346 tsk

Format	*tsk*(*a*)
Arguments	*a* Index number of the image to be read from the stack *a* 0 (Top of the stack)
Output values	None
Image type	Any
Type	Stack operator
Function	Read the file at position *a* in the image stack. If *a* is zero, the image at the top of the stack is read. If *a* is negative, an image that was recently located at the top of the stack is read. When an image is popped from the stack, it is not destroyed
Switch images	Yes
Remarks	The stack is implemented as a circular array of TIFF image files, labelled *stack_image*(*0*), *stack_image*(*1*), . . ., *stack_image*(*stack_size-1*) Image *stack_image*(*stack_pointer*) is at the top of the stack. When we perform *pop*, the top of the tack moves to *mod*((*stack_pointer* + *1*), *stack_size*) The image which had been at the top of the stack immediately before *pop* is now at position (*stack_pointer-1*). Assuming that not more than (*stack_size-k*) images have been pushed (using *push*) onto the stack, the previous *k* images can be accessed successfully by *tsk*. (There is no warning, to indicate whether or not this can be accomplished successfully) Also see *push*, *pop*, *rsi*

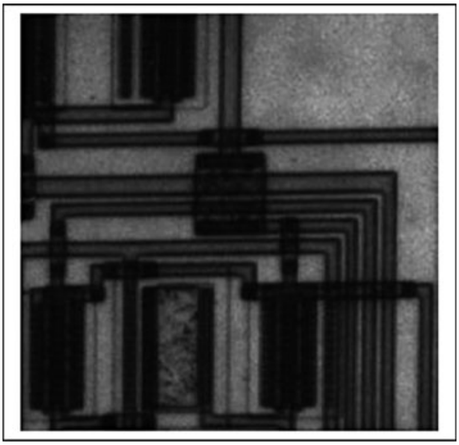
41.347 tur

Format	*tur*(*a*)	
Arguments	*a*	Angle of rotation (degrees)
	b	Control cropping/no cropping
Defaults	*b*	0 (no cropping)
Output values	None	
Image type	Grey scale or binary	
Type	Affine	
Switch	Yes	
Function	Rotate the image by a given amount, about the centre of the image	
Switch images	Yes	
Typical uses	Defining circularly symmetric filters	
Remarks	*tur*(*X*, *0*) *or tur*(*X*) keeps the image size constant; the corners are cropped. On the other hand, *tur*(*X*, *1*), rotates the image without cropping. This makes the output image larger. In both cases, bilinear interpolation is used to avoid producing ragged edges	

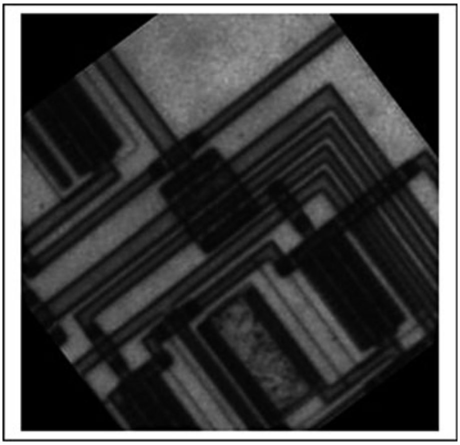
Illustration

Input image	Circuit
File number	136
Command	*tur*(*37*) *tur*(*37*, *1*)

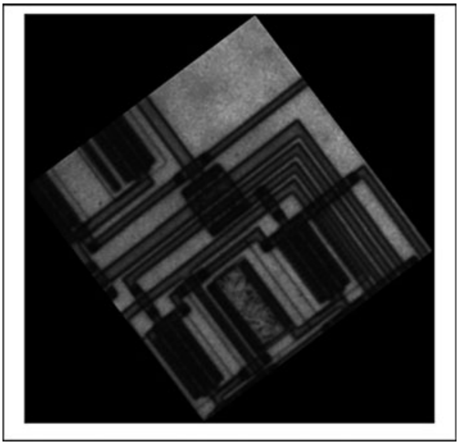
Top-left: original



Top-right: after *tur*(*37*). Image size is unchanged



Bottom-left: after *tur*(*37*, *1*) *I*mage size has increased

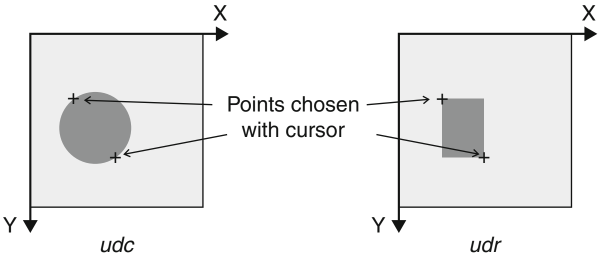


41.348 txt

Format	*txt*(*a*, *b*)	
Arguments	*a*	Text to be superimposed on the current image
	b	Colour of lettering
Output values	*a*	See below
	b	‘r’ (red lettering)
Image type	Any	
Type	Binary image analysis	
Function	Argument *a* not supplied: Superimpose *Current* and *Alternate* labels on the displayed images Argument *a* supplied: Superimpose text defined by *a* on the display of the current image (In neither case is the image changed)	
Switch images	Yes	
Typical uses	Teaching and demonstrations	
Remarks	The text disappears when the current image is modified and redisplayed	

41.349 udc/udr

Format	*udc*(*a*)	
Arguments	*a*	0, fill the circle/rectangle; otherwise draw an open circle/rectangle
Defaults	*a*	0
Output values	None	
Image type	Grey	
Type	Graphics	
Function	Draw the largest possible circle/rectangle intersecting two points specified by the user with the cursor. See drawing below	
Switch images	Yes	
Typical uses	Defining structuring elements for dilation/erosion	
Remarks	Also see *pgn*, *use* Figure – For both functions. two points, denoted by crosses, are selected by the user with the cursor. *udc* draws the circle whose diameter intersects those two points. *udr* draws that rectangle whose diagonally opposite vertices coincides with these points.	



41.350 uds

Format	*uds*(*a*)
Arguments	*a* Increment the image stack pointer
Output values	None
Image type	Any
Type	Stack operator
Function	In Stack Mode, or when watching the stack with *ssk* in Terminal mode, use *uds*(*a*) to read the *a*th stack image
Switch images	Yes
Remarks	The variable stack_pointer is incremented: $stack\_pointer = mod(stack\_pointer\text{-}a,\ stack\_size)$ This is equivalent to rotating the circular array of images forming the image stack. Also see *tsk*

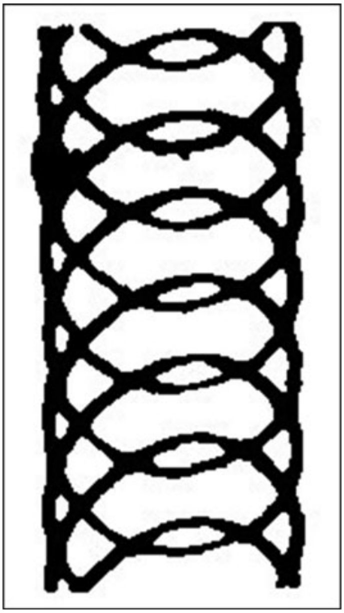
41.351 ult

Format	*ult*
Arguments	None
Output values	None
Image type	Binary
Type	Binary image transformation
Function	Ultimate erosion of the negative image
Switch images	Yes
Typical uses	Inspecting pre-defined patterns, by reducing them to a simpler form that is easier to analyse
Remarks	This generates a number of white spots, each one indicating a regional maximum of the Euclidean distance transform. Also see *rmx*, *gft*

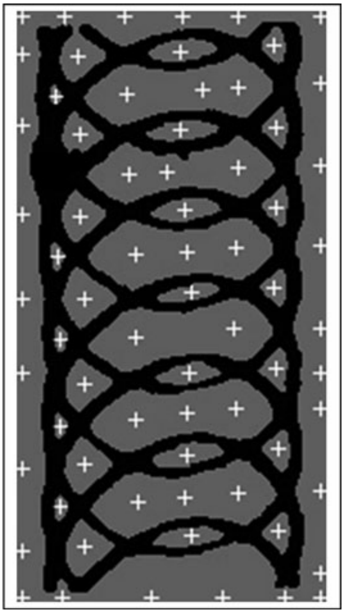
Illustration

Input image	Cake decoration pattern
File number	55
Command	*ult*

Left: original



Right: after processing



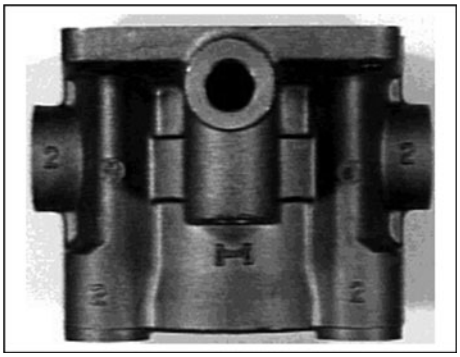
41.352 uns

Format	*uns*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Grey scale
Type	Convolution operator
Switch	Yes
Function	Unsharp mask; high-frequency accentuating filter
Definition	Linear convolution in a 3 × 3 window with the following weight matrix

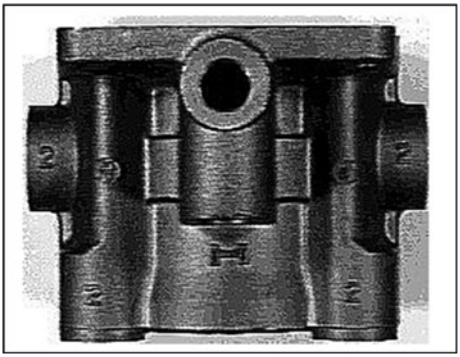
−0.1667	−0.6667	−0.1667
−0.6667	4.3333	−0.6667
−0.1667	−0.6667	−0.1667

Switch images	Yes
Typical uses	Enhancing low-contrast images for display
Remarks	
Illustration	
Input image	Hydraulics manifold
File number	38
Command	*uns*

Left: original

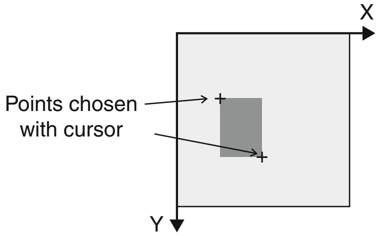


Top-right: after processing



41.353 use

Format	*use*(*a*)
Arguments	*a* Dilation control
Defaults	*a* 1
Output values	None
Image type	Two binary images: current image; alternate images
Type	Binary image analysis
Function	1. The user defines a structuring element (SE) window using the cursor (See diagram below) 2. The current image is then thinned (*thn*) and dilated *a* times 3. The contents of this window is then used to define the SE 4. Erode the current image using the SE just calculated (*gbe*)
Switch images	Yes
Typical uses	Designing/choosing SEs
Remarks	*use* is an interactive design tool for morphology; it is not intended for batch mode operation. Also see *jas*. The user defines the top-left and bottom-right vertices of the SE window.

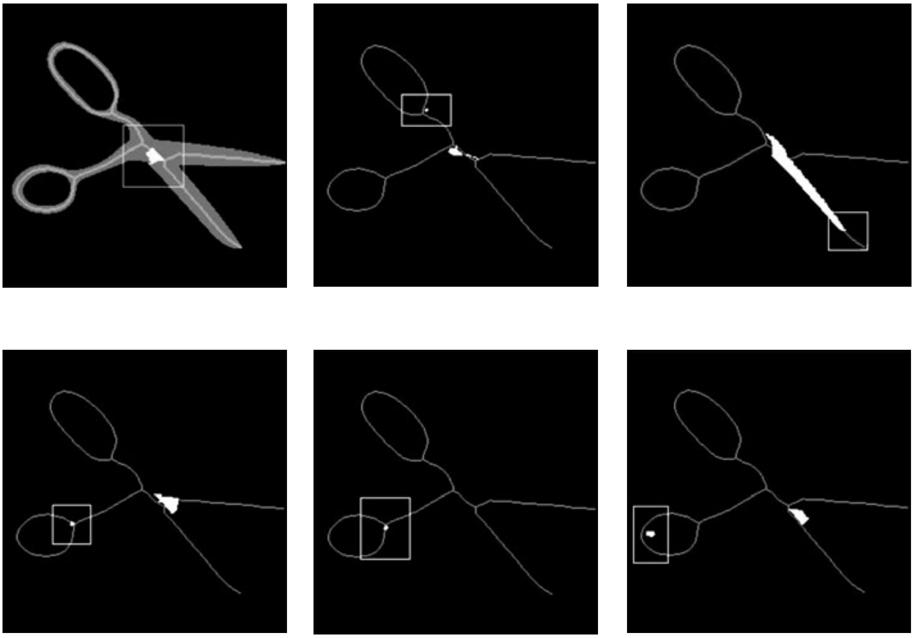


Illustrations

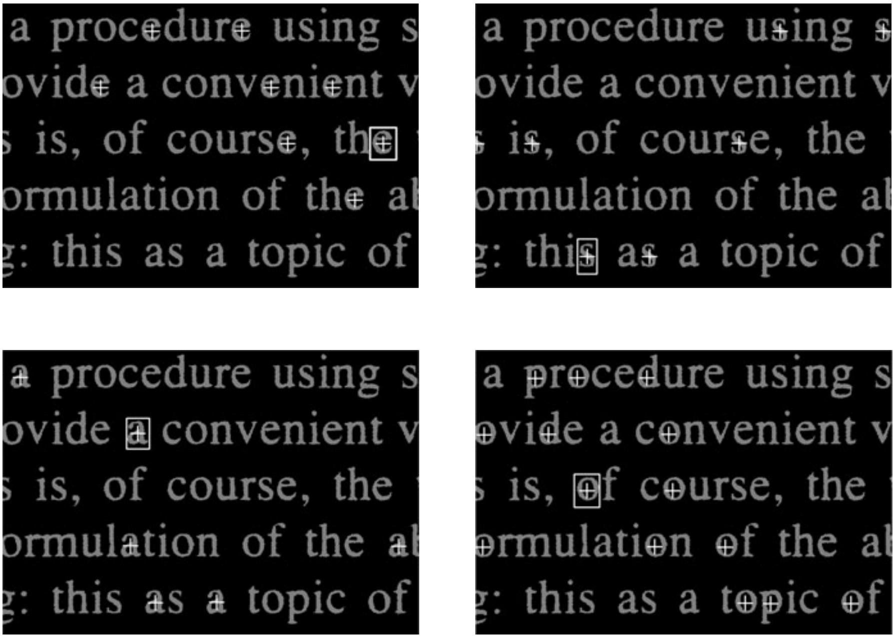
Example 1: *Scissors Images*

Input image	Scissors
File number	48
Command	*rni(48), a = use(1); neg, sqr, neg, rni(48), swi, adi, mxi*

Top left: The SE defined by the user consists of that part of the dilated skeleton lying within the white rectangle. Erosion of the original "scissors" image with this SE, produces a single blob in the area close to the hinge of the scissors. Processing: *rni(48), a = use(1); neg, sqr, neg, rni(48), swi, adi, mxi.*
Top centre: The SE is based on the Y-shaped feature at the base of the larger finger-hole. Erosion detects this and some points near the hinge; the SE is too small to be more specific.
Top-right: The SE is a close approximation to a diagonal line. Erosion with this SE detects most of one blade.
Bottom-left: Erosion with a small SE detects the Y-shaped feature and points in the hinge area.
Bottom-centre: Erosion with this larger SE detects the Y-shaped feature but not the hinge area.
Bottom-right: This small SE is based on the loop of the smaller finger-hole. Erosion with it incorrectly detects the 'V' formed by the blades.



Example 2: *Printed text*
Top-left: The SE is a dilated skeleton of the lower-case letter 'e'. Erosion with this SE correctly detects all other occurences of 'e' and nothing else.
Top-right: The SE is a dilated skeleton of the lower-case letter 's'. Erosion correctly detects all other occurences of 's' and nothing else.
Top-left: The SE is a dilated skeleton of the lower-case letter 'a'. Erosion correctly detects all other occurences of 'a' and nothing else.
Top-left: The SE is a dilated skeleton of the lower-case letter 'o'. Erosion correctly detects all other occurences of 'o' but erroneously detects 'p' and 'd' as well



41.354 vfs

Format *vfs*(*a*)

Arguments *a* Upper limit on spatial frequency, measured along the horizontal axis

Defaults None

Output values None

Type Drawing and image generation

Switch Yes

Function Generate a pattern with a sinusoidal intensity variation along each row. The spatial frequency varies linearly with vertical position in the image; low frequencies are generated near the top and high frequencies near the bottom

Switch images Yes

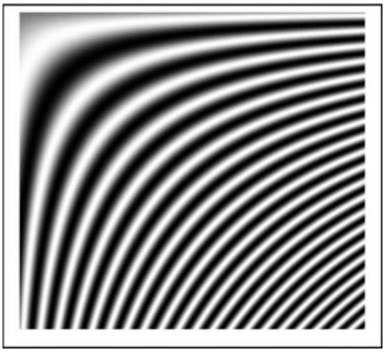
Typical uses Analysing the behaviour of 1-d (horizontal) filters

Remarks When a filter is applied to the image generated by *vfs*, the contrast varies according to the vertical position (i.e., spatial frequency). The contrast can be measured using a simple QT command sequence and a graph plotted of filter response versus frequency; the use of this function is illustrated below

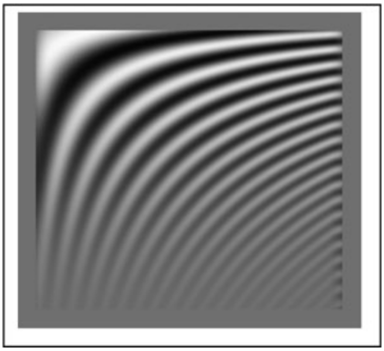
Illustration

Notes: The image generated by *vfs*(*20*) is shown below, together with the result of horizontal smoothing (i.e., *con*[*1, 1, 1, 1, 1, 1, 1, 1, 1, 1*; *1, 1, 1, 1, 1, 1, 1, 1, 1, 1*; *1, 1, 1, 1, 1, 1, 1, 1, 1, 1*]) applied three times. The row-minimum (I_{min}) and row-maximum (I_{max}) were then found for each row. A graph of ($I_{max} - I_{min}$) was then plotted against spatial frequency. (Graph labelled 'Filter response')

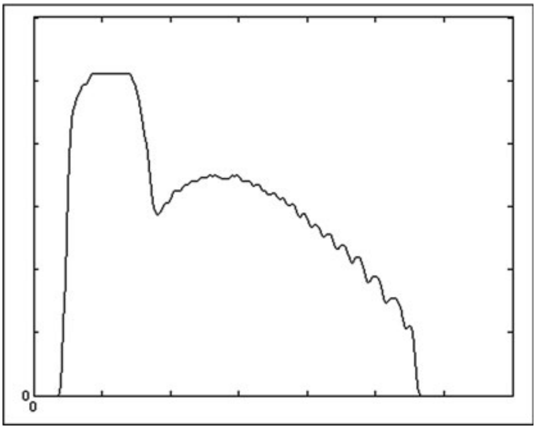
Left: vsf(20)



Right: after filtering



Filter response (Abscissa, Spatial frequency. Ordinate, Filter response, i.e., minimum-to-maximum difference in intensity along each row)



41.355 vgr

See *con*

41.356 vpl

Format	*vpl*(*x1*, *y1*, *x2*, *y2*, *v*)	
Arguments	*x1*	X-coordinate of the beginning of the line
	y1	Y-coordinate of the beginning of the line
	x2	X-coordinate of the end of the line
	y2	Y-coordinate of the end of the line
	v	Intensity of the line
Defaults	*v*	255
Output values	None	
Type	Drawing and image generation	
Switch	Yes	
Function	Draw a digital straight line of intensity *v*, beginning at [*x1*, *y1*] and ending at [*x2*, *y2*].	
Switch images	Yes	
Typical uses	Annotating images for publication	
Remarks	Either or both end points may lie outside the image;	

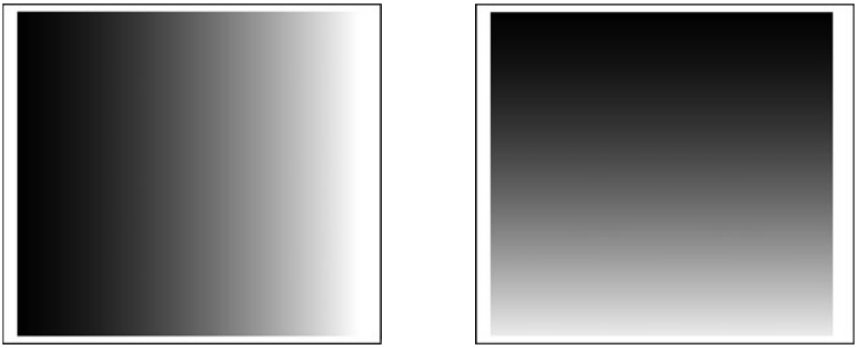
41.357 w/wis

Format	*w*(*a*)
Function	Copy the current image into the image stack
Arguments	*a*
Defaults	0
Output values	None
Image type	Any
Type	User interface
Switch images	No
Function	Synonym for *wis*
Remarks	Alternatively, CNTRL/click on the appropriate thumbnail in Figure 2

41.358 wgx/wgy

Format	*wgx*
Arguments	None
Defaults	Not applicable
Output values	None
Type	Drawing and image generation
Switch	Yes
Function	Draw an intensity wedge; intensity varies linearly with distance across the image
Definition	$c(i, j) \Leftarrow 255*i/I_w$ where I_w is the width of the image
Switch images	Yes
Typical uses	Plotting intensity profiles. Analysing depth maps. Run-length coding. Generating test patterns
Remarks	Use *wgx*, *sca*(*3*) to generate a series of (8) intensity steps. *wgy* is defined in a similar way and generates a wedge in which the intensity varies in the vertical direction

Illustration



Left: *wgx*

Right: *wgy*

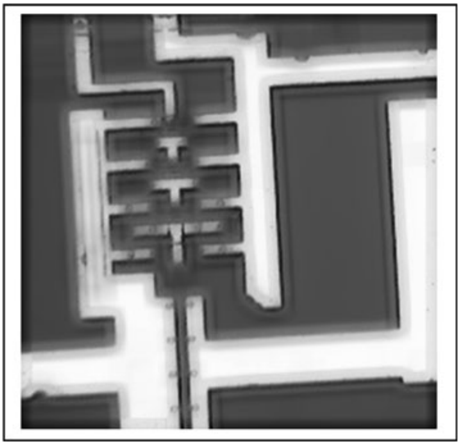
41.359 wnf

Format	*wnf*(*a*, *b*)	
Arguments	*a*	Width of the processing window
	b	Height of the processing window
Defaults	*No arguments*:	$a = b = 3$
	One argument:	$b = a$
Output values	None	
Image type	Grey scale	
Type	Linear filter	
Switch	Yes	
Function	Adaptive filter (Wiener filter), within a rectangle of size a × b pixels	
Typical uses	Defining circularly symmetric filters	
Remarks	The Wiener filter is a type of linear filter that adaptively tailors itself to the local image variance. Where the variance is large, it performs little smoothing and where the variance is small, the Wiener filter performs more smoothing. It often produces better results than fixed-kernel linear filtering (e.g., *raf*, *caf*), as it is more selective; it preserves edges and other high frequency parts of an image better, while reducing noise levels in areas of nearly constant intensity. The Wiener filter works best when the image is degraded with constant-power ('white') additive noise (e.g., Gaussian noise)	

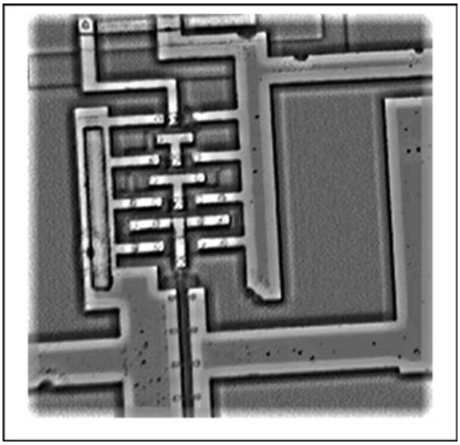
Illustration

Input image	Circuit
File number	257
Command	*wnf(15)*, *sub*, *enc*

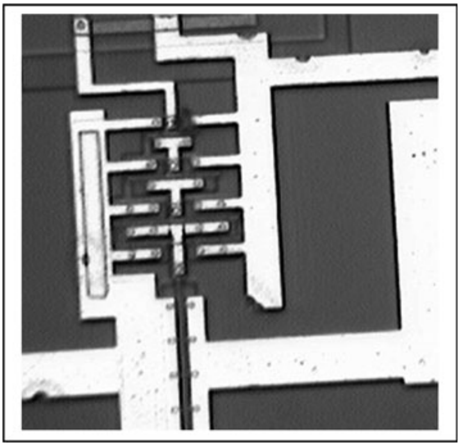
Top-left: original



Top-right: after *wnf(15)*



Bottom-left: after *wnf(15)*, *sub*, *enc* Image size has increased



41.360 wri

Format	*wri*(*a*, *b*)	
Arguments	*a*	File name/number
	b	0, save as TIFF file (loss-less, image can be reproduced exactly), or 1, save as JPEG file (compact image is degraded slightly)
Defaults	*a*	0
	b	0, save as TIFF file (loss-less, image can be reproduced exactly)
Output values	*d*	File path
Image type	Any	
Type	Input/output	
Function	Save the current image in TIFF format (if $b = 0$) or JPEG format ($b = 1$)	

Remarks: The file name (*a*) can be a number, or character string. An image saved in TIFF format can be read back again, using *rei*. Example:

Write: *wri*(*123*)   Read: *rei*(*123*)

The image is saved in a folder called *QT_work_files*, which must be located on MATLAB's search path. Sample commands:

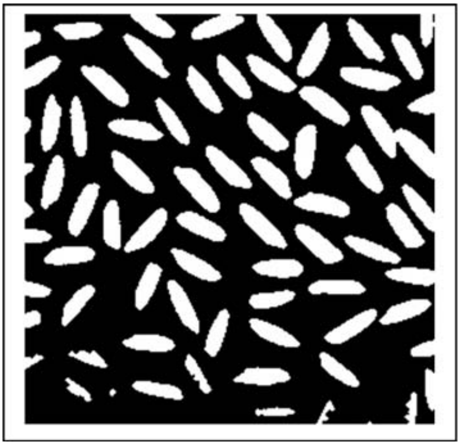
Command	Equivalent to	Image type
wri	wri(0, 0)	TIFF
wri(123)	wri(123, 0)	TIFF
wri(123, 1)	wri(123, 1)	JPEG
wri('im_file12')	wri('im_file12', 0)	TIFF

Also see *wsi* and *w*

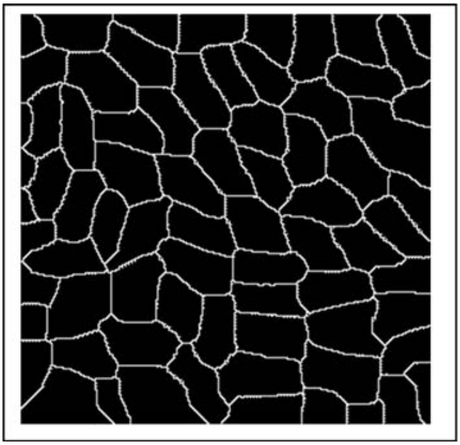
41.361 wsh

Format	*wsh*(*a*)
Arguments	*a* Connectivity (*a* = 4 or 8)
Defaults	*a* 8
Output values	None
Image type	Binary
Type	Blob transform
Switch	Yes
Function	Watershed function
Definition	The output of *wsh* is an image in which white pixels indicate points that are equidistant from the two (or more) nearest blobs
Switch images	Yes
Typical uses	Analysis of granular materials and webs that are sparsely populated with 'point' features
Remarks	The image resembles a honey-comb; each 'cell' consists of a 4- or 8-connected set of points each of which perceives the same blob in the input image as its nearest neighbour
Illustration	
Input image	Rice grains (file number 3) after thresholding and clean-up
Image generation	rni(3), enc, thr, ero, dil
Command	*wsh or wsh*(*8*)

Left: original



Top-right: after *wsh*



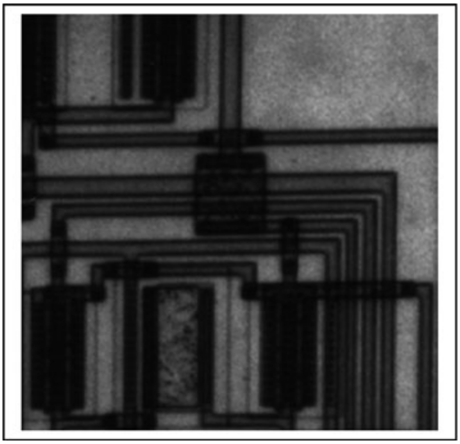
41.362 wsi

Format	*wsi*(a, b)	
Arguments	a	Multiplier for current image, before addition
	b	Multiplier for alternate image, before addition
Defaults	No arguments:	$a = b = 0.5$
	One argument:	$b = 1 - a$
Output values	None	
Image type	Grey scale or binary	
Type	Combining two images	
Switch	Yes	
Function	Linear weighted sum of the current and alternate images	
Definition	The current image is multiplied by a and the alternate image by b. These weighted images are then added together	
Typical uses	Defining new types of image filter, by adding outputs from standard filters together	
Remarks	In the illustration below, the output of *sed* (Sobel edge detector) is used to enhance the ‘crispness’ of an image	

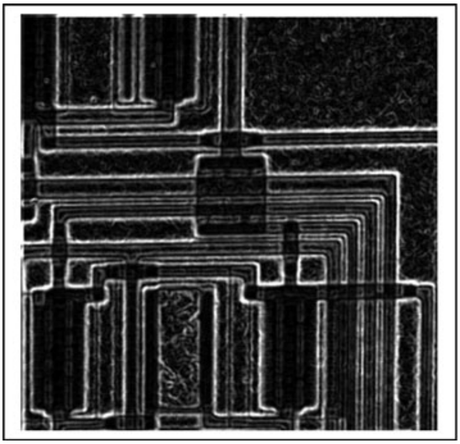
Illustration

Input image	Circuit
File number	136
Commands	*rni*(*136*), *sed*, *enc*, *rni*(*136*), *wsi*(*0.75*), *enc*

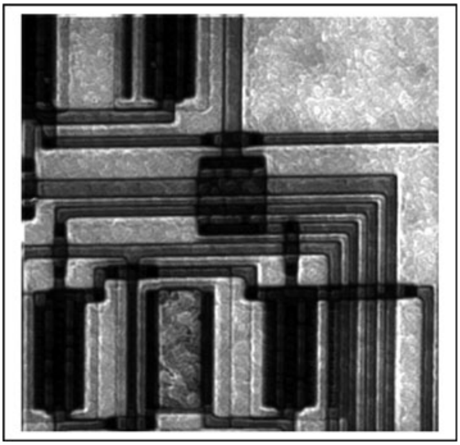
Left: original



Right: after *sed*, *enc*



Bottom-left: after processing



41.363 xyl

Format	[*xmin*, *xmax*, *ymin*, *ymax*] = *xyl*(*a*)	
Arguments	*a*	Function control
Defaults	*a*	0 (Do nothing, except calculate limit values)
Output values	*xmin*	Minimum X-coordinate
	xmax	Maximum X-coordinate
	ymin	Minimum Y-coordinate
	ymax	Maximum Y-coordinate
Image type	Binary	
Type	Measurement and anlysis	
Function	Calculate the minimum and maximum X and Y limits considering all white points. Three other functions are selected by the input parameter *a*:	
	a = 0 (default)	Do nothing else
	a = 1	Draw smallest enclosing rectangle
	a = 2	Crop at smallest enclosing rectangle
Switch images	No	
Typical uses	*bbx* and *mar* operate on a single blob, whereas *xyl* considers all white pixels. Also see *blp*/*brt*/. . .	

41.364 xyz

Format *b* = *xyz*(*a*)

Arguments *a* Text string

Default *a* Allowed

Output values *b* Text string

Image type Not relevant

Type Test m-function

Function Demonstrating/testing the update operator (*qtu*). *xyz* has no other useful function

Switch No

Remarks Updating of QT operators can be demonstrated thus:

1. Delete the file *xyz.m* from the folder *QT_new_mfunctions*
2. Enter *qtp*(*0*) (This ensures that all uploaded functions are ignored)
3. Run any one of the following, with or without the semi-colons.
   xyz; *q* = *xyz*; *q* = *xyz*('*My message*');
4. Enter *qtu*('*xyz*'). This down-loads Version 2 but does not run it. Check this, by inspecting the folder *QT_new_mfunctions* again
5. Repeat Step 3. Notice that this runs Version 1 of *xyz* again
6. Enter *qtp*(*1*). This enables the updated version to be run (in Step 9)
7. Repeat Step 3. This time, Version 2 of *xyz* runs

Notice that the user can switch between Versions 1 and 2 by repeatedly running *qtp*, *qtp*(*0*) or *qtp*(*1*)

41.365 yxt

Format *yxt*

Arguments None

Defaults Not applicable

Output values None

Image type Grey scale or binary

Type Affine

Switch Yes

Function Interchange the vertical and horizontal axes

Definition $C(j, i) \Leftarrow A(i, j)$

Switch images Yes

Remarks Non-square images will produce a result that cannot be combined with the original by using dyadic operators: *adi, sub, mni, mxi,* etc. Also see *bxy*

41.366 z

Format	*z*
Arguments	None
Defaults	Not applicable
Output values	None
Image type	Any
Type	Miscellaneous
Function	Interchange the current and alternate images
Remarks	A quick alternative to *swi* (The closest that MATLAB can approach *CNTRL/Z*)

41.367 zed

See *ced*

41.368 zer

See sci

Appendix A: Glossary

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

The following terms are defined in the context of Machine Vision, even though some have a wider range of use. Relevant terms relating to visual perception in humans and animals are also included but anatomical and physiological features of biological vision systems are not. Notice that British spelling is used.

2D. 2 dimensional.

2-dimensional (colour) scattergram. Derived by plotting one colour component against another, on a point-by-point basis, for a polychromatic image. E.g., we might plot the amount of red light against the amount of blue light.

2-dimensional array. (a) See Array-Scan Camera. (b) Digital image represented by a 2-dimensional matrix or array.

3D. 3-dimensional.

4-connected. In a binary image, pixels P and Q are 4-connected if P is a 4-neighbour of pixel Q, or P is a 4-neighbour of pixel R and R is 4-connected to Q. A blob is said to be 4-connected if all its members are 4-connected to each other.

4-neighbour. The 4-neighbours of a pixel (i, j) in a digital image are located at (i − 1, j), (i + 1, j), (i, j − 1) and (i, j + 1).

8-connected. In a binary image, pixels P and Q are 8-connected, if P is a 8-neighbour of pixel Q; or P is a 8-neighbour of pixel R and R is 8-connected to Q. A blob is said to be 8-connected if all its members are 8-connected to each other.

8-neighbour. The 8-neighbours of a pixel (i, j) in a digital image are located at (i − 1, j − 1), (i − 1, j), (i − 1, j + 1), (i, j − 1), (i, j + 1), (i + 1, j − 1) (i + 1, j) and (i + 1, j + 1).

A

Aberration. Failure of an optical lens to produce an exact point-to-point correspondence between the object and its resulting image. Types: chromatic, spherical, coma, astigmatism and distortion.

Absorption. Loss of radiant energy (e.g., light) as it passes through a transparent medium, such as glass, water, etc. Absorption is a function of wavelength, type of material and path length.

Accuracy. Performance parameter for a vision system that derives numeric value(s). (e.g., position/orientation of a specified feature) Not to be confused with Precision. (q.v.).

Achromatic (colour). A neutral colour such as black, white or grey (also see Chromatic).

Achromatic response function. (Also see Chromatic Response Function.)

Achromatopsia. (See Central Achromatopsia.)

Acousto-optic scanner. Type of scanning element in a flying spot scanner in which the scanning is performed by a laser beam directed through a piece of glass or clear crystal driven by a piezoelectric transducer. The glass behaves like a diffraction grating with a number of lines per mm proportional to the exciting frequency. By varying the exciting frequency a scanning action is produced.

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_42,

Active Illumination. Illuminating a scene with a light source that is co-ordinated with image acquisition. Examples: strobed flash tubes, steered structured light beams, Scanned LIDAR (q.v.) beams.

Additive mixture (of light). A mixture of light rays of various wavelengths, which reach the eye, creating an impression of colour that none of them alone might provide Ciné film, television and computer displays rely on light addition. (Also see Subtractive Mixture.)

Add-on lens. a simple lens that screws onto the front of a lens assembly and alters its Focal Length (q.v.) and Minimum Object Distance (q.v.). Also see Diopter, Extension Tube.

AGC. See Automatic Gain Control.

Algorithm. A sequence of well-defined computational steps, for solving a problem. Unlike an Heuristic (q.v.), an algorithm has a precisely known outcome, within a given input domain, although the term is often used informally, to include heuristics as well.

Aliasing. Phenomenon which gives rise to spurious results if an image is sampled at a spatial frequency below the Nyquist limit.

Ambient light. Light present in the optical environment of an imaging system that is generated from uncontrolled "external" sources, such as room lights, sun-light, light from nearby (human) operator's work-station, etc. Ambient light is unpredictable and highly variable. It therefore represents a major source of noise for a vision system.

Analogue (or analog). A smooth, continuous signal, voltage, current, or other physical value, representing a real number.

Analogue-to-Digital Converter (ADC). Device for converting an analogue signal into a series of discrete, digitally-encoded numbers (integers), for processing in electronic hardware or a computer.

Anamorphic mirror. Mirror that produces different magnifications along different directions in the image plane. Other distortion effects are also possible.

Angle of view. Angular width of the cone in 3D space within which objects are visible to a camera.

Angstrom. Unit of length: 10^{-10} m, 10^{-4} μm.

Anomaloscope. An instrument used for detecting anomalies of colour vision. The test subject adjusts the ratio of two monochromatic lights to form a match with a third monochromatic light.

Anomalous trichromacy. Abnormality in human vision in which there are three functioning cone types but one or more of them has abnormal spectral sensitivity. As normal, people with this condition require three primaries for colour matching but accept matches that normal subjects will not.

Aperture response. Cross-sectional sensitivity profile of a photosensitive element forming part of a solid-state array.

Aperture. Controls the amount of light passing through a lens. The F number scale (1.4, 2, 2.8, 4, 5.6, 8, 11, 16) for a standard photographic lens reduces the amount of light passing through the lens by half with each step increase in the scale.

Architecture. Organisation of the opto-electronic hardware of a vision system.

Array processor. Hardware accelerator for arithmetic calculations, using parallel processing (SIMD) techniques.

Array representation. Digital image representation in the form of a 2D array of pixels. Allows binary, monochrome (grey-scale) and colour images to be coded.

Array-scan camera. Imaging device in which photo-detectors are packed in a 2D array. Cameras used for broad-casting are invariably array-scan type.

Artifact. An artificially created structure (by accident or on purpose), form or shape, usually part of the background, used to assist in measurement or object location.

Artificial Intelligence. Subject of academic study devoted to providing computers with capability of performing functions normally attributed to human intelligence: learning, adapting, recognising, classifying, reasoning, self- correction and improvement.

Aspect ratio. Width:height ratio of a video image. (Broadcast standard is usually 4:3.)

Aspherical lens. Lens, or lens, assembly which has one, or more, non-spherical surfaces. Used principally to reduce spherical and other aberrations.

Assimilation. A perceptual phenomenon in which the colour of part of a scene is perceived as closer to the colour of the surround than it would if viewed in isolation. A thin coloured stripe between two black bars will look darker than it would between two white bars. (Also known as the *Von Bezold spreading effect.*)

Astigmatism. Optical aberration in which, instead of a point image being formed of a point object, two short line images are produced at right angles to each other.

Asynchronous camera. Video camera in which the scan starts on demand, rather than waiting for the end of the present frame period.

Auto iris lens. Lens whose aperture automatically adjusts itself to allow for variations in the ambient light.

Autocollimation. Procedure for collimating an optical instrument with objective and cross hairs in which the instrument is directed towards a plane mirror and the cross hairs and lens are adjusted so that the cross hairs coincide with their reflected image.

Autocorrelation function. Image correlated with itself shifted by various amounts. See Correlation.

Auto-focus. Facility in an imaging system allowing it to control the focus of the lens, to obtain the sharpest image on the detector.

Auto-iris lens. Lens with an electronically controlled iris. Maintains nearly constant mean light level when light conditions vary. Useful addition to AGC. (q.v.).

Automatic gain control. Function of a circuit (e.g., video amplifier) whose gain is automatically adjusted to allow for changes (in ambient light). In a video camera, the system gain is changes in an attempt to maintain a constant mean output level. Hence, as the input light level decreases, the gain increases, thereby maintaining a constant video level. Also see Auto-Iris Lens.

B

Back light compensation (BLC). Camera function used to avoid blooming and other effects of optical over-load.

Back-focal length. Distance from the rear surface of a lens to its focal plane.

Back-ground. That part of a scene behind the object to be inspected. Also used to describe the region surrounding a blob-like figure in a binary image.

Back-lighting (Back-illumination) Lighting arrangement for generating a high-contrast image of an object, by forming its silhouette.

Back-propagation. Training scheme for a neural network, by adjusting weights in the hidden and input layers.

Baffle. shield used to block light from entering/leaving an optical system.

Band-pass filter. Optical, electronic or image filter, which allows a narrow range of frequencies/wave-lengths to pass, blocking all others.

Bar code. Product/document/carton identification system that employs a series of machine-readable lines of varying widths of black and white. Usually read with a laser scanner.

Barrel distortion. Imperfection of an optical system in which a rectangle appears to bulge on all sides. Opposite of Pin-Cushion Distortion (q.v.).

Basic colour term. A word naming a basic colour category. (e.g., red, yellow, etc. but excluding terms such as “reddish-yellow”).

Bay. Imagine that a blob in a binary image is an island. A bay is an area of sea between headlands. One type of constituent part of the Convex Deficiency (q.v.) of the blob.

Beam expander. Optical system for increasing the width of a light beam.

Beam splitter. Partially silvered or aluminised mirror which splits an incident beam of light into a reflected beam and a perpendicular transmitted beam’. Other forms of beam splitter are also available.

Bezold-Brücke hue shift. A shift in the apparent colour of a long-wavelength stimulus towards yellow or blue as intensity increases. If a pair of long-wavelength lights differing only in intensity are compared, the brighter one will appear to be more yellow (less red) than the other. At short wavelengths, the higher intensity light will appear more blue (less green). There are three invariant points in the spectrum at which the colour appearance of monochromatic lights does not change with intensity. (See Invariant Hue.)

Binary hue. A hue that is perceptually mixed. E.g., orange appears to be a mixture of red and yellow. All binary hues are mixtures of two of the unique hues.

Binary image. Image in which each pixel can be represented by only the two binary digits 0 or 1, (i.e., black or white).

Birefringence. Phenomenon in certain naturally occurring materials, such as fluorspar, that has different refractive indices for light with horizontal and vertical polarisation.

Bit map. Representation of an image, graphics or characters by a 2D array of pixels arranged in rows and columns. In image processing, it is more common to refer to this an array representation.

Blanking. Interval during raster-scanning when the video signal is suppressed. (Originally introduced to allow frame/frame/field fly-back in CRT cameras.)

Blob analysis. Identification and measurement of distinct objects in an image, based on their geometric properties (e.g., area, perimeter, shape factor, length, number of holes).

Blob. 4- or 8-Connected (q.v.) blob-like shape in a binary image

Blooming. An effect by which a highly illuminated point on an image sensor spreads out to form a disc; caused by the high intensity of the incident beam saturating the image sensor at that point.

Borescope. Device for internal inspection of pipes, engines, gun barrels, etc. Used where access is difficult. It consists of a long narrow tube, containing a telescope optical system, with a number of relay lenses. Light is provided via the optical path, LEDs or fibre-optic bundles. A 45° mirror at the end allows inspection of tube walls. Also called endoscope.

Boundary. Curve delineating the boundary between two image regions, separated by intensity, colour or texture.

Brewster’s Angle. Angle of incidence for which a light wave polarised in the plane of incidence is totally transmitted at a dielectric boundary (about 67.4° for air-glass).

Brightness. That attribute of a visual sensation in which parts of a scene appear to emit more/less light than others. Often used synonymously with Intensity (q.v.). (Also see Luminance.)

Buried channel CCD. Type of CCD with a buried layer of doping material which together with the electrodes causes the charge packets to move below the surface; giving a high-charge transfer efficiency.

C

Calibration. Procedure for relating image- and world co-ordinates. Important for both metrology and robot control. May require use of specially made calibration targets, i.e., objects/charts with known dimensions.

Camera format. Length of the diagonal of a camera photo-detector array, or the diameter of a CRT camera. Standard sizes: 1″, 0.667″, 0.333 and 0.25″.

Candela. Unit of luminous intensity defined as 1/60th of the luminous intensity per cm^2 of a black body radiator at the temperature of freezing platinum.

Canny. Name of a type of Edge Detector. (q.v.).

Carbon dioxide (CO_2) laser. Powerful, continuous, IR laser that can emit several hundred watts at a wavelength of 10.6 μm. Used for welding and cutting.

Catadioptic. Mirror-lens assembly providing the function of a light-weight, long-focus (telephoto) lens. When the object-camera distance has to be large, a catadioptic system can be very effective, providing a brighter, clearer image than a multi-element lens assembly. However, the large working distance requires vibration-free mounting of the camera, to avoid image blur.

CCD. See Charge Coupled Device. Popular solid-state image sensor; the "work-horse" for the MV industry. Implemented with standard large scale integration technology. In a *Frame-transfer CCD*, the charge pattern, representing the entire image, is transferred from the sensing area to an on-chip storage buffer. From here it can then be sampled at random, to generate an RS170, CCIR, or a non-standard raster-scan video signal. The output from the camera is always one frame behind the image currently being captured.

CCPD. See Charge Coupled Photodiode Array.

Central achromatopsia. A defect of colour vision due to damage to the Visual Cortex (q.v.) or other parts of the brain. (The more common forms of colour defect are due to photoreceptor loss or abnormality.)

Centroid (properly *Geometric Centroid*). Standard method for defining blob positions in a digital image. The co-ordinates of the centroid of a blob-like object S are determined by calculating the centre of mass of a uniform laminate object with the same outline as S. In practice, the moments $M_{0,1}$ and $M_{1,0}$ are computed.

Chain code. Code used for describing a curve such as the periphery of an object. Each discrete point on the curve is represented by an integer from 0 to 7, representing the direction of the next point as an angle ranging from 0° to 315° in 45° steps. If the number of possible directions of travel is reduced to four (0°, 90°, 180° and 270°), the resulting sequence is called a Crack code.

Chalnicon. Type of vidicon with high resolution and sensitivity; not easily damaged by intense light.

Character. Single printed, typed or hand-written letter, digit, punctuation mark, mathematical or other symbol. Usually represented within a computer using one byte of storage.

Charge coupled device (CCD). Monolithic semiconductor device arranged as an array of elements such that the output from one element provides the stimulus for the next. Implemented with standard large scale integration technology. In a *Frame-transfer CCD*, the charge pattern, representing the entire image, is transferred from the sensing area to an on-chip storage buffer. From here it can then be sampled at random, to generate an RS170, CCIR, or a non-standard raster-scan video signal. The output from the camera is always one frame behind the image currently being captured. This is the most popular camera for the MV industry.

Charge coupled photodiode array (CCPD). Image sensor which combines the best properties of CCDs (low noise, high clock rate) and photodiode arrays (good spectral response, resistance to blooming), i.e., the image sensors are photodiodes but the scanning function follows the principles of operation of a CCD.

Charge injection device (CID). Charge transfer device used as an image sensor in which the image points are accessed by reference to their Cartesian co-ordinates. CIDs have low dark current but are relatively noisy. A CID array can be sub-scanned (i.e., a small ROI (q.v.) is sampled), addressed at random and read without destroying stored information. It is resistant to optical overload. Photo-detector sites are contiguous on the chip, providing the maximum area for capturing light.

Charge transfer efficiency. Efficiency with which charges are transferred between neighbouring locations in a CCD.

Chirality. Property of the shape of a blob. By "turning over" the shape, the chirality is reversed. Is used to determine Handedness. (q.v.).

Chirp scanner. Type of acousto-optic scanner excited by a "chirp" pulse, i.e., a pulse of linearly increasing frequency. The pulse, as it moves through the crystal, produces the effect of a travelling lens which performs the scan.

Chroma. Subjective measure of the amounts of pure chromatic colour present, irrespective of the amount of achromatic colour. For colours of the same hue and brightness, chroma and saturation are equivalent. Chroma increases with brightness, even if saturation is constant. Chroma is used in the Munsell Colour System (q.v.).

Chromatic. All colours other than the neutral colours (i.e., white, black, and pure grey) are chromatic. The word "colour" in ordinary language is often used to refer exclusively to chromatic colours. For example, we refer o to "coloured socks".

Chromatic aberration. An optical aberration in which the component colours in a point source of white light are brought to a focus at different points; caused by the variation of the refractive index of the glass with wavelength.

Chromatic response function. Perceived colour expressed as a function of the wavelength of a stimulus.

Chromaticity coordinates. Coordinates that specify position in a chromaticity diagram. (See CIE.) The chromaticity coordinates (x, y, z) are derived from the tristimulus values (X, Y, Z) by the formulae $x = X/X + Y + Z$, $y = Y/X + Y + Z$.

Chromaticity diagram. A diagram that represents the plane defined by the equation $X + Y + Z = 1$ in a Tristimulus Space (q.v.). Points in this plane represent variations in variations in the orientation of the vector [X, Y, Z] (hue and saturation) but not its length (intensity). (Also see CIE.)

Chromaticity. Denoted by a point in a chromaticity diagram. Chromaticity is often used as a convenient way to approximate hue and saturation, while ignoring brightness.

Chromophore. Light-absorbing part of a photo-pigment. In mammals and humans, the chromophore is a form of vitamin A, called *retinal.* The photo-pigment dissociates into two parts (opsin and chromophore) in response to the arrival of a photon of suitable energy, thereby evoking the visual response. The differences in spectral sensitivity among photo-pigments are due to variations in the structure of their opsins.

CID. See Charge Injection Device.

CIE (Commission Internationale de l'Eclairage). International body that recommends standards and procedures for quantifying lighting and colourimetry. *CIE Chromaticity Diagram.* Diagrammatic representation of the chromaticity co-ordinate system used to represent colours, by relating two of the normalised standard primary colours.

CIE illuminant C. Standard illuminant (i.e., idealised light source) that is an approximation to average daylight. Specified by its spectral power distribution.

CIE Standard Observer. A set of tri-stimulus values providing a standard and objective way of describing the colours of different light sources. While few people agree completely with the colour classification provided by the Standard Observer, it is close to what most people with normal colour vision regard as reasonable. More than one Standard Observer has been defined, to accommodate a variety of different viewing conditions.

Classification. Assignment of patterns/features/images/objects, etc. to one of two, or more, sets (classes) whose members are distinguished by some ill-defined notion of "similarity." A device for performing classification is called a *classifier* and is usually designed through self-adaptive learning.

Closing. Morphology operator useful for "filling holes" and smoothing edge boundaries in a binary image. Equivalent to *dilation* followed by Erosion. (q.v.).

C-mount. Standard threaded lens mounting for a video camera. A 1″ × 32 thread is employed. Distance from the shoulder of the lens to the image plane is 17.52 mm. A C-mount lens may be used on a CS-Mount (q.v.) camera, but a 5mm-adapter ring is required.

Coaxial illumination and viewing. Collimated front illumination, parallel to the camera's optical axis, produced by introducing a Beam Splitter (q.v.) in front of the camera.

Coherent fibre-optic bundle. Bundle of optical fibres which preserves the spatial relationship between its input and output. Also see Image Conduit.

Coherent illumination/light. Lighting in which the wave-front is well defined e.g., lies on a plane normal to the direction of travel.

Collimator. Optical device for producing a parallel beam of light.

Colour. Refer to items in this glossary using the British spelling*: colour.*

Colour anomia. Inability of a person to name colours, despite intact colour vision as demonstrated by non-verbal tests.

Colour constancy. Stability in the perceived colour of a surface with varying illumination.

Colour space. An abstract 2D-, 3D- or hyper-space for representing colours so that the similarity relationships that exist among them are preserved.

Colour temperature. The temperature of the perfect black-body radiator whose chromaticity is closest to that of a given light source. Colour temperature is often used as a convenient way to specify the quality of a light source, rather than completely defining its spectral power distribution. Colour temperature is most appropriately applied to sources that have broad, smoothly changing spectral power distributions.

Colourimetry. The science of measuring colour by matching. The main focus of colourimetry has been the development of methods for predicting perceptual matches on the basis of physical measurements.

Colour cube. Refers to a 3-dimensional space in which we plot the RGB colour information.

Colour scattergram. Graphical representation of the colour variation to be found in a scene and consists of a set of points superimposed onto the colour triangle. It is convenient, in practice, to plot the colour scattergram as an image, so that it can be filtered, threshold and processed using the usual image processing software.

Colour Space. A 2- or 3-dimensional space representing colour using a quantitative co-ordinate system. Provides an absolute and objective definition of colour. Common colour-space representations: RGB, HSI, LAB and CIE. (q.v.).

Colour Temperature. A concept in Colour Science describing the *apparent* visual appearance of a source of radiation, not its actual temperature.

Colour triangle. Abstract geometric figure that is useful when analysing and discussing colours. Each point in the triangle corresponds to a different colour in the scene being viewed. Points that are close together usually have similar colours.

Colour. A sensation created in the visual system of human beings (and certain higher animals) by varying wavelengths of light falling on the retinal rods. Cones are not sensitive to colour changes. A variety of factors, including culture, language, drugs, ageing and disease, affect human perception of colour. Colour categories can only be defined subjectively. Colour Science is the study of colour perception in nature and guides the design of printing, photography, television and computer displays, as well as paint mixing and even the automobile and fashion industries.

Coloured. Term used to signify an object or scene that has some discernible colour, other than neutral, which has low values of saturation.

Colourimetry. Experimental techniques in which colour is assessed against objective, quantitative scales, or by comparison with stable reference surfaces.

Coma. Optical aberration of an optical system which gives a point object a pear shaped image.

Complementary colours. Two colours that can be additively mixed to produce an achromatic colour. (See Additive Mixture.)

Composite Video. Electrical signal combining (a) analogue wave-form representing a succession of intensity profiles across an image, (b) horizontal synchronisation pulses, (c) vertical synchronisation pulses and (d) blanking signals.

Computer Vision. The science and mathematical theory of artificial vision systems. The subject is concerned principally with solving "difficult" problems using sophisticated computational procedures. It is academic rather than practical.

Concavity tree. Hierarchical description of a blob in which each Bay (q.v.) and Lake (q.v.) is represented (recursively) by a concavity tree. The nodes in the tree can be labelled in a variety of ways. Combines local and global shape information. Can be used to determine Chirality (q.v.) in an absolute sense. Also see Handedness.

*Concurrent processo*r. Computer system architecture that lends itself very well to inspecting objects on a conveyor belt. It consists of a number (N) of identical processing units but which operate [360/N] degrees out of phase in their execution cycles.

Condenser Lens. Used to collect and redirect light from an odd-shaped source (such as a curved lamp filament), to provide better-controlled illumination. Often used to collect light from a small lamp and project a near-uniform light beam onto an object. May be combined with a concave mirror, behind the light source, to collect more light.

Cone. One of the two main classes of photo-receptor found in the vertebrate eye. Cones produce respond to relatively high levels of light and provide the main sensors for colour vision. There are three types of cones in the normal human eye, each type being most sensitive at a different point in the spectrum.

Confusion line. A line on a chromaticity diagram representing colours that are not discriminable by a subject with a particular defect of colour vision. For example, the protan confusion line represents colours that are indistinguishable by person with Protanopia (q.v.).

Conjugate points. In optics any pair of points such that all rays emanating from one point are imaged at the other.

Connectivity analysis. Determining sets of 4- or 8-connected pixels in a binary image, in order to isolate distinct blob-like objects, which are then measured individually.

Connectivity paradox. An 8-connected arc of unit width should bisect the image plane. In fact, pixels adjacent to that arc, on opposite sides, are 8-connected to each other.

Connectivity. Topological property of a binary image related to the number of 'holes' (lakes) it contains.

Contrast. Synonymous with Spatial Contrast (q.v.) that the darkest pixel becomes black and the brightest white. Stretching the intensity scale improves subjective contrast and hence visibility but does not usually assist feature detection.

Contrast transfer function. Complex function of spatial frequency characterising an optical system; gives a measure of the extent to which an optical system degrades the contrast of a test pattern of equally spaced lines with a square wave intensity profile. Loosely called modulation transfer function.

Convex deficiency. The set of points within a convex hull that are not in the object. It includes *Lakes* (regions totally enclosed by an object), and *Bays* (regions lying between the convex hull perimeter and the object).

Convex hull. Shape obtained by stretching a rubber band around a given blob-like object.

Convolution. Image filtering operation. Typically, a linear weighted sum of the intensities is computed for each M × N Neighbourhood (q.v.) in the input image (e.g., 3 × 3, or 5 × 9, 11 × 7 mask). Operators of this kind are able to enhance edges, blur/sharpen images remove noise and detect small spots.

Correlation. Mathematical measure of the similarity between (parts of) images. The correlation score (S) is formed by deriving the sum of the squares of the differences between intensities within the two images to be correlated. One image is shifted relative to the other to obtain the best fit (i.e., to maximise S). Also called Template Matching. (q.v.).

Crack code. See Chain Code.

Crack detector. Non-linear local operator, intended to detect thin, dark streaks and small spots. Latterly, identified in the terminology of mathematical morphology as: [*dilate; erode; subtract original image*].

Crack. Thin dark feature in a (grey-scale) image.

Cross talk. Process by which an unwanted signal is induced in a circuit because of its close proximity to a neighbouring circuit. The term can be applied to adjacent photo-sensors in a solid-state array.

CRT (Cathode ray tube). Old-fashioned imaging device. Large, heavy, fragile susceptible to magentic fields, short working life. Image geometry is not fixed accurately. Now largely superceded by solid-state sensors in standard applications but is still used for specialised tasks: UV, IR, low-light, in areas *CS-mount.* Standard mounting for a lens to a video camera. The screw thread is identical to that for the C-mount but the distance from the shoulder of the lens to the image plane is 12.52mm. Hence, a CS-mount lens cannot be used on a C-mount camera.

Cumulative histogram. See Histogram.

D

Dark current. Current that flows in the output circuit of an image sensor even in the absence of illumination.

Dark-field illumination. Method of microscope illumination in which the illuminating beam is a hollow cone of light, formed by an opaque stop large enough to prevent direct light from entering the camera lens; the specimen is placed at the apex of the cone and is seen only with light scattered, diffracted or refracted by the specimen.

Data reduction. Selectively discarding unwanted data, while retaining important information, in an image, or sequence of images, by some suitable coding, algorithm.

Decision tree. Hierarchical classification technique based on classical AI.

Dedicated (vision) system. System which is pre-configured for a specific application. Able to function when plugged in with no further programming. Also called *Turn-key System.*

Denisyuk process. A technique for producing white-light holograms.

Depth map. See Height Map.

Depth of field (Dof). Width of that region on either side of the focal plane, where the image produced by an optical system remains in focus. Normally, a large Dof is advantageous. Normally, a small lens aperture and bright object illumination will improve the Dof.

Depth of focus. The range of image distances over which the image remains sharp for a given object distance.

Depth perception. Ability of a vision system to measure/judge distance from the image sensor (i.e., eye/camera) to the various parts of the scene being viewed.

Descriptive syntactic process. A pattern recognition technique which models an object by a set of features and by the spatial relationships between these features.

Deuteranopia. One of the two varieties of red-green colour blindness (also known as green-dichromacy). People with deuteranopia results have lost the function of the M-Cones but display essentially normal visual sensitivity throughout the spectrum. (Also see Protanopia.)

Diascopic illumination. Form of illumination in which an object is viewed in transmitted light.

Dichoptic matching. A colour matching experiment in which different coloured stimuli are presented separately to the two eyes.

Dichroic filter. Optical filter used to transmit certain wavelength of light and reflect others. Dichroic filters are often built in to quartz-halogen-tungsten lamps, to reduce IR content of output beam.

Dichromacy. Term relating to people possessing two independent channels for conveying colour information. Most "colour blind" people are dichromatic having lost the function of type of cone types. They are not strictly colour blind, since their vision is sensitive to a single chromatic dimension as well as to brightness.

Diffraction pattern sampling. Inspection procedure which compares interference patterns, produced by diffraction, rather than by "direct" viewing.

Diffraction. Wave phenomenon of light whereby the intensity distribution in a wave is spatially redistributed after encountering an obstacle. It accounts for the ability of waves to go round corners.

Diffraction-limited. An optical system is said to be diffraction-limited when its performance is limited only by the wave nature of light and not the more usual optical aberrations. The finite aperture of a lens limits its resolution.

Diffuse lighting. Illumination technique, projecting light onto a matt surface first. This acts as the illuminator for the object being viewed. Baffles avoid direct illumination of the object. Avoids sharp shadows and glinting due to Specular Reflection (q.v.).

Diffuse reflection. Light returned from a matt (i.e., non shiny) surface is diffuse; light is scattered in all directions (above the object surface). It is not strongly polarised.

Diffuser. Translucent material, e.g., polypropylene, used to produce diffuse illumination.

Digital camera. Video camera whose output is already digitised and formatted as a digital image. Interfacing to a computer is simple.

Digital image processing (DIP, or Image processing). Mathematical theory and practice relating to the manipulation of digital images. The term is used in two ways: (a) specifically, to referring to image-to-image mapping procedures and (b) generically, to encompass other processes, such as coding, compression, description and analysis. Computer Vision (q.v.) is usually regarded as being concerned with higher-level functions, such as scene modelling and understanding.

Digital image. Image represented by an array of pixels. Each pixel in a Monochrome (q.v.) digital image is assigned one integer value (typically 8 bits), representing its intensity. Colour digital images require the storage of three integers per pixel representing its RGB colour components. These formats allow easy storage and processing within a computer.

Digital signal processor (DSP). VLSI integrated circuit, designed for high speed arithmetic processing and often imbedded in a dedicated vision system.

Digital-to-analogue converter (DAC). VLSI circuit used to convert digital data (e.g., processed images) into Analogue (q.v.) form, for display on a monitor.

Digitisation. Process of sampling and converting a video, or other analogue, signal into a digital value (i.e., integer) for subsequent storage and processing. (See Analogue-to-Digital Converter.)

Dilation. Morphology operator for binary or grey-scale images. White/bright regions become larger. (Also see Erosion.)

Diopter. Measure of the power of a lens, equal to the reciprocal of its Focal Length. (q.v.) Usually applied to Add-on Lenses (q.v.).

Dispersion. (a) Separation of a beam of light (e.g., by a prism) into its component wavelengths. (Also called chromatic dispersion.) (b) Scattering of light as it travels through a transparent material (e.g., glass, plastic) or fluid (e.g., air, water). Minute material defects, suspended particles and impurities, bubbles, droplets, mist, aerosols, emulsions, smoke, fumes are primary causes of scattering.

Distal. Located away from the point of origin or attachment.

Distortion. Defect of an optical system in which magnification varies with angular distance from the axis, causing straight lines to appear curved.

Dither coding. Display technique, enabling monochrome images to be represented on a binary (i.e., 2-level) screen. Each image pixel (intensity I) is assigned a group of display pixels (binary), arranged in a square. Of these, a number proportional to I are selected at random and switched on. The remainder are off.

Dominant wavelength (of an optical stimulus) The wavelength of the monochromatic light, which when mixed with a specified achromatic light in suitable proportions, subjectively matches the stimulus. The dominant wavelength varies with the choice of achromatic reference. (Illuminant C (q.v.) is an obvious option.) Dominant wavelength is the colourimetric equivalent of Hue. (q.v.).

Dot-matrix code. Method of recording binary data on a packet, product, etc. in the form of a 2D array of spots, which can be read by a simple vision system.

Duality. Relationship that exists between certain pairs of morphological operators. (e.g., Erosion and Dilation, q.v.) If P produces the same result as the sequence [*negate, Q, negate*], then operations P and Q are said to be duals of one another.

Dust. Airborne pollution causes serious damage to vision systems, since it contaminates optical surfaces. (e.g., lamps, diffusing reflectors, lenses, etc.). It is a mistake to use unfiltered compressed air to eliminate dust, since it will deposit oil.

DUV. See UV-C.

Dyadic operator. Image processing operator which combines two input images. See Monadic.

Dye laser. Type of tunable laser in which the active medium is a dye such as acridine red, with very large molecules. Lasing action takes place between the ground state and the continuum of higher excited states.

Dye penetrant. Liquid dye used for detecting cracks or surface defects in non-magnetic materials.

Dynamic aperture. The effective transverse aperture of a linear image sensor which is being mechanically scanned in the transverse direction.

Dynamic range. A characteristic property of any measuring instrument; it is equal to the ratio of the maximum to minimum measurable values of the physical quantity which the instrument was designed to measure. No camera can yet match the dynamic range of the eye, which is about 10^{14}:1.

E

Edge detector. Computational procedure for enhancing edges in images.

Edge point. In a binary image, a white pixel which has a least one black neighbour, or vice versa. In a grey-scale image, a pixel whose intensity value is significantly different from that of at least one of its neighbours. For colour images, edge pixels can be similarly defined, in terms of differences in intensity, hue or saturation.

Electromagnetic (EM) spectrum. A wide range of wavelengths, extending from microwaves to x-rays and gamma rays is potentially useful for imaging. The visible spectrum, VIS (q.v.), forms just a small part of this.

Electron gain. The number of electrons at the output stage of photomultiplier for each photoelectron generated at the photocathode.

Electronic speckle pattern interferometry (ESPI). Holographic technique in which only a speckle pattern characterising an object is recorded as opposed to the full interference pattern used in normal holography; the lower resolution required enables a TV camera to be used as the recording medium with all the added flexibility of computer processing of the image.

Electron-volt. The amount of energy required to move an electron through a potential difference of one volt. (Used to measure energy of sub-atomic particles and photons.)

Emitron. Early form of video camera developed by EMI Ltd., based on the principle of the iconoscope.

Encoder (shaft or position). Provides positional/rotational information for control of image acquisition for web and conveyor inspection systems.

Endoscope. Tube-like instrument widely used in medicine and industry, to view inside regions providing limited access. Now being challenged by small flexible probes, fitted with miniature cameras and LED illumination. Also see Borescope.

Episcopic illumination. Form of illumination in which an object is viewed in reflected light.

Equiluminant. (Equal in luminance) Two light sources may be equally bright to a human observer, although chromatically different (i.e., they have different spectral power distributions).

Erosion. Dual morphological operation of Dilation. (q.v.) Defined for binary and grey-scale images. White/bright regions become smaller.

Error rate. See False-Positive, False-Negative.

Euler number. Topological property of a binary image. Equal to the number of distinct blobs minus the number of holes (lakes).

Extension tube. Short cylindrical threaded tube, inserted between a camera and its lens. Reduces the minimum working distance but is not recommended for use with zoom lenses due to the loss of focus when altering magnification.

F

False positive/negative: Error in a vision system which incorrectly determines that the object being inspected is *acceptable/unacceptable.*

Farnsworth-Munsell 100-hue test. A standard test for deficiencies of human colour vision in which the subject is asked to arrange a set of 100 coloured chips around a circle. People with normal colour vision will usually arrange the chips with very few deviations from the predicted order. On the other hand, people with abnormal colour vision will produce a different arrangement, thereby providing useful information about the nature of their defect.

Fast Fourier transform (FFT). Fast procedure for computing the Fourier transform of a digitised signal or image. Produces an image which is quite unlike the original in appearance. The input is approximated by a linear weighted sum of an harmonically-related series of sinusoids of varying phases.

Feature extraction. Process of measuring image features, in order to characterise an object/image. This data will be used, in a Pattern Recognition system, to classify it.

Feature vector. List of numeric measures of an object (e.g., area, shape factor, number of lakes, etc.) that can be used to classify it.

Feature. Abstract image attribute, such as top-most edge point, centroid, limb-end of the skeleton, centroid of largest bay, etc.

Fibre optics. Flexible bundle of glass, plastic or quartz fibres, with a cladding of lower refractive index. Light is transmitted along individual fibres by total internal reflection. Used for illumination and imaging. See Coherent Fibre Optic Bundle and Image Conduit.

Fibrescope. Optical instrument, similar to a Borescope (q.v.), but uses a flexible, coherent optical fibre bundle. Used to view inaccessible objects.

FIC. See Flexible Inspection Cell.

Fiducial mark. Line, cross or target-like feature, used as a reference for locating/measuring pattern features on a printed document, textile, etc. Usually printed on the edge and removed from finished product by trimming. Occasionally embedded in the pattern itself.

Field curvature. Aberration of an optical system causing the surface of sharpest focus to be curved.

Field of view. (FOV) Cone in space that a vision sensor (eye or camera) can observe. Described by two angles, measuring horizontal and vertical limits of sight. Might also be given as distances measured across the plane normal to the central line of sight.

Field. A frame in an interlaced scanning system, consists of two parts: odd-field and even-field. In an odd-field, odd-numbered lines are scanned. Similarly, for the even field.

Filter. A device or algorithm that selectively transmits certain signals, while attenuating others. In Digital Image Processing (q.v.), a (software) filter might typically accentuate, or blur, regions where the intensity gradient is high. In optics, a colour filter (multi-layer dielectric film) can be used to absorb light within a given band of wavelengths, before it reaches the image sensor. Correct use of optical filters can help to simplify the image processing in a vision system Most optical filters used are wavelength (colour) selective but polarising filters are used to suppress glinting due to specular reflection.

FIR. See IR-C.

Fixture. A device or jig to hold and locate a work-piece during processing or inspection operations.

Flange Focal Length (or the *Flange Focal Distance*) is the distance between the locating surface of the lens mount and the image plane. Its value is practical, not theoretical.

Flexible inspection cell. Lighting and viewing arrangement that provides a variety of lighting conditions, cameras and lenses. Useful for Remote Prototyping (q.v.) of vision systems.

Fluorescence. Emission of light, or other EM radiation, at one wavelength by matter, when it is irradiated at a shorter wavelength. Unlike Phosphorescence (q.v.), the emission lasts only while the stimulating radiation is present. Most people are familiar with UV-to-VIS fluorescence but this is not unique, as the phenomenon also occurs at other wavelengths, e.g., x-ray-to-violet, blue-to-red. VIS-to-IR. Fluorescence plays an important role in the perceived colour of many objects. Many biological materials fluoresce. The unnatural brightness of "day-glo" paints is due to fluorescence.

Fluorescent lamp. Cheap light source, providing a spectrum that is a reasonably good approximation to day-light. Often in the form of long tube. To avoid Aliasing (q.v.) with the camera, it is necessary to drive fluorescent tubes using a high-frequency inverter, operating at about 50KHz. Failure to do so will produce hum bars in the video image.

Flying-spot laser scanner. Scanner used for optical inspection of fast-moving web materials, when very fine detail must be detected. A pencil- or fan-shaped laser beam is scanned across the web, probably by galvanometer mirrors, or a rotating multi-faceted mirror. Light scattered from the surface is collected by a suitable photo-detector (e.g., photo-multiplier tube).

f-number (or f-stop). The ratio of the focal length of a lens to the diameter of its input aperture. For a given lens, reducing its f-number by closing its aperture, increases the image brightness and reduces the Depth of Field. (q.v.).

Focal length. Distance from the Focal Point to the principal point of a lens, or mirror. This is a constant defined by the lens material and geometry.

Focal plane. Plane perpendicular to the lens axis at the focal point.

Focal point. Point at which rays parallel to the axis of an optical system converge or from which they appear to diverge. Most optical systems have two principal foci, produced by rays incident from the left and from the right.

Focus following. Ranging and tracking technique that uses image processing to find the position of best focus of the camera, as the focal-length of its focus is adjusted, and thereby estimate the object-camera distance.

Foot-candle. Unit of illumination equal to 1 lumen (q.v.) per square foot.

Fourier-domain inspection. Inspection protocol based on parameters derived from the Fourier transform of an image. 1D and 2D periodic patterns and textured images are appropriately analysed in this way.

Fovea The central area of the human retina, containing the densest concentration of photo-receptors (i.e., cones only). It also possesses other adaptations for high resolution vision.

Frame buffer. Digital image store in a frame grabber or image processing hardware.

Frame grabber. Interface unit that connects a video camera to a computer. On command from the associated driver software, it samples the video signal and creates a digital image that is then be transferred to the computer's memory for subsequent processing. Also see Variable-Scan Frame Grabber.

Frame. Video picture, representing a snapshot of a scene, with an effective exposure time of 40 ms (Europe) or 33.3 ms (USA). In an interlaced system, it comprises one odd- and one even-field. See Field.

Freeman code: See Chain Code.

Frei-Chen. Name of a type of Edge Detector. (q.v.).

Fresnel lens. A thin lens constructed in the form of a series of prismatic ridges, giving a conventional lens. Very large lenses can be produced. Image quality is inferior to a lens but may be adequate for many Machine Vision applications.

Front focal length (or *Front focal distance*, *FFD*) is the distance from the front focal point of the system to the vertex of the first optical surface.

Front lighting. Illumination from source(s) on the camera side of an object, enabling its surface features to be observed.

f-stop. See f-Number.

FUV. See UV-C.

G

Gallium arsenide laser. A laser that emits infrared radiation ($\lambda = 900$ nm) at right angles to a junction region in gallium arsenide. Since it can be modulated directly at microwave frequencies, it is useful for optical communication. Cryogenic cooling is required.

Gamma (γ). Numeric parameter describing the power-law relationship between the intensity of the optical input (I) to a camera and its output signal (S). The formula is of the form $S = K.I^{\gamma}$, where K is a constant. If $\gamma = 1$, the camera is linear but produces a lower contrast compared to cameras in which $\gamma > 1$.

Gamma correction. Non-linear transformation, of the form $Y = X^{-\gamma}$ performed electronically and applied to a camera output to make its response approximately linear.

Gamma ray. Short-wave-length, penetrating EM radiation, emitted by certain radio-active isotopes. Wave-length is shorter than that of x-rays. Gamma-rays are highly dangerous, being a prime cause of cancer. Although conceptually very simple, gamma ray imaging requires highly sophisticated handling techniques. Gamma-ray cameras are used in astronomy and for medical imaging. Potentially very useful for industrial inspection, providing a flexible alternative to x-rays but are not popular on account of the dangers.

Gauging. Non-contact dimensional examination of an object to determine whether it conforms the specified physical dimensions.

Geometric transform. Type of image processing operator in which the transformed image is still recognisable but is rotated, distorted or warped.

Glancing illumination. See Low-Angle Illumination.

Glinting. Shiny, specular reflections from polished, glossy, wet, or oily surfaces.

Global operator. Image processing operation applied to the whole image. c.f. *Region of interest.*

Gradient (of intensity). Gradient (of intensity). Strictly, the magnitude of the first derivative of the (continuous) intensity function with respect to the image co-ordinates. When processing digital images, it is necessary to accept an approximation. The direction of maximum gradient is used less often but is valuable for analysing texture and surface geometry, given knowledge about the angle of illumination from a point source. See Edge Detector.

Gradient vector. The orientation and magnitude of the rate of change in intensity at a point or pixel location in the image.

Grassman's laws. Empirical laws governing the results of additive colour mixing. (See Additive Mixture.) Grassman's laws assert the existence of exactly three dimensions of variation in perceived colour: *hue*, *brightness*, and Saturation. (q.v.) In addition, they imply that colour mixing is linearly additive: if stimulus A matches stimulus B, and stimulus C matches stimulus D, then the additive mixture of A and C will match the mixture of B and D, irrespective of their spectral composition.

Grating (Human vision). A convenient test pattern consisting of alternating stripes, which can differ in colour (chromatic grating), or brightness (achromatic grating).

*Grating (*optical element). Array of uniformly-spaced opaque stripes with transparent spaces between. A fine grating can separate (disperse) different wavelengths of light. Also used for *Moiré* and Structured-Light Projection Techniques. (q.v.).

Grey level/scale. Numerical representation of brightness in a monochrome image. Typically, black is represented by zero, white by some positive constant (e.g., $255 = 2^8 - 1$) and grey by intermediate integer values.

Grey-scale image. Consists of an array of pixels, each of which can have more than two intensity values.

GUI. Acronym for *Graphical User Interface.* Macintosh and Windows operating systems both enable GUI operation.

H

Hair. Fine white arcs in a binary image. (One/few pixels wide.) c.f. *Crack.*

*Handednes*s. Refers to the function of the widget from which a blob in a binary image was derived. For example, a certain blob may represent the silhouette of a piece of leather used to make a right-handed glove. If we turn over that piece of leather, its Chirality (q.v.) changes but its handedness does not. (It is still part of a right glove.)

Hardware. Electronic image-processing equipment designed specifically for applications requiring high-speed operation. Probably of fixed/limited functionality.

Hard-wired. Relatively inflexible electronic circuit designed to perform a specific function. While software could perform the same function it would probably be slower.

HDTV (High Definition television). Modern broadcast standard for terrestrial and cable television. Not much used for Machine Vision.

Height map. Digital image in which intensity measures the 3D profile of an object or surface.

Height. Point measurement of object profile. Normally found by measuring distances from the sensor. Used synonymously with *Range or Depth.*

Helium-neon (HeNe) laser. Low-power, red laser. (λ = 612 and 632 nm) Large and fragile compared to modern diode lasers.

Helson-Judd effect. In human vision, the tendency of lighter achromatic surfaces to take on the hue of the illumination under which they are viewed and darker achromatic surfaces to take on the complementary hue.

Heuristic. Intuitive "rule-of-thumb" procedure for solving difficult problems. Often rule-based. May be faster than an Algorithm (q.v.), if one exists. In many situations, no algorithm is known, so heuristics offer only solution. Heuristics can be used to speed up algorithms. A procedure may always give correct result but cannot (yet) be proved to do so; it is an algorithm but this is unknown to us. Hence, it must be accepted on its known merits as an heuristic. Since heuristics are not provably correct in every situation, they are often spurned. It is essential therefore to view their performance on a statistical basis.

High-pass filter. Device or algorithm that retains fine (high-frequency) features in an image, while attenuating low frequency (slow changing) ones.

Histogram equalisation. Procedure for adjusting the intensity scale of a grey-scale image, so that the histogram of the resulting image is nearly flat. One of the prime methods of improving image contrast.

Histogram. Vector summarising the distribution of intensities in a digital image. The ith element (H(i)) indicates how many pixels have intensity *i.* The sum C(i) = [H(0) + H(1) + H(2) + ... + H(i)] is called the *Cumulative histogram* and is the basis for the procedure known as Histogram equalisation.

Hologram. Representation of a 3D surface, obtained using coherent radiation. When an holographic image (on a plane surface) is illuminated appropriately, the observer obtains the impression that he is looking at the original object, albeit with strange colouration. For practical reasons, not yet used significantly in Machine Vision.

Hough transform. Procedure for identifying "broken" straight lines and curved arcs (e.g., circle segments) of known form. It is easy to identify points lying on a line or curve since they all map into a single spot in the transform space. Gaps in the curve merely reduce the contrast between this spot and its back-ground.

HSI (Hue, saturation and intensity). Alternative to the RGB representation for describing colour. HSI was proposed as a model describing human colour perception. Since is possible to calculate HSI from RGB and vice versa, it is fallacious to claim that the former is "better" than RGB; in mathematical terms, any calculation in HSI space can be performed in RGB space.

Hue coefficient. The ratio between the response in one of the chromatic channels to the total chromatic response. A stimulus that is seen as equally bluish and reddish would for example have a hue coefficient of 0.5 for both red and blue while the unique hues all have coefficients of 1.0 for one channel and zero for the others.

Hue. Component in the description of colour, which determines its name. One of the three standard elements of colour appearance. (The other two are brightness and saturation). Its colourimetric equivalent is Dominant Wavelength. (q.v.) Although colour labels such as "yellow," "turquoise," "scarlet," etc. are defined subjectively, they can be predicted reasonably accurately by measuring hue alone. However, other factors (e.g., back-ground colour, gloss, texture and motion) also influence the naming of colours. Hue differences depend primarily on variations in the wavelength of light reaching the eye.

Hueckel. Name of a type of Edge Detector. (q.v.).

Hum bar. Disturbance of a digitised video signal that occurs when the light intensity generated by the lamps changes significantly during the camera's frame-scan period. It consists of a pattern of broad a dark bands modulating the image intensity.

Hunting. Used to describe a control system that oscillates, failing to reach its ideal goal state. Object transport mechanism may exhibit oscillatory behaviour as they try to maintain a constant speed. Also, describes the inability of an auto-iris lens to reach a stable state under certain light conditions.

Huygen's principle. Idea that each point on the wave-front of a light beam may be regarded as a point-source of secondary waves, that spread uniformly outwards, like ripples on a pond but in 3D space. The sum of all secondary waves generated in this way uniquely determines the wave-front at a later time.

I

Iconoscope. Early form of camera tube in which a high-velocity electron beam scans a mosaic of tiny capacitors on which the image is formed as a pattern of electrical charges. The scanning beam discharges the capacitors sequentially, thereby generating the video signal.

Illuminant point. That point on the chromaticity diagram that indicates the colour of a light source.

Illumination. Incident radiation impinging on an object or surface. The term is applicable to all parts of the EM spectrum. It is commonly found that the spectral content of the illuminating beam includes UV and/or IR, as well as VIS. This may reduce image quality significantly. Illumination includes wanted light and troublesome ambient light, which is highly variable and effectively constitutes noise.

Image acquisition sub-system. That part of a vision system concerned with the mechanical handling of the object/material being examined, illumination, optics, sensing (camera) and pre-processing, finally forming a digital image. The importance of the image acquisition stage cannot be over-emphasised.

Image analysis. Extraction and analysis of information regarding the contents of a digital image, leading to a final decision about further action to be taken (e.g., guide a robot to pick up an object, accept/reject a component, process it further) Image analysis may be extremely simple (e.g., find the centroid of a blob), or more complicated (read a product label) The term is sometimes subsumed under the umbrella of Image Processing. (q.v.).

Image capture. Process of converting a camera output, usually a video signal, into a digital image. A device for this is usually called a Frame-Grabber. (q.v.).

Image conduit. Fused coherent fibre-optic bundle, often in form of straight or L-shaped "pencil." Sometimes available as frustrum, used for changing image size or aspect ratio. It may have twist (e.g., 90°, 180°).

Image distortion. Used to describe warping effects of an optical system, or (CRT) camera scan non-linearity. While distortion can be corrected by software, if the warping function is known or can be measured, this places an extra burden on the image processor.

Image enhancement. Image processing operations which improve the visibility of low-contrast image detail. Usually performed for display to human beings, rather than machine analysis.

Image format. Describes the dimensions of the light-sensitive area of an imager. It is usual to quote the diagonal dimension. Common formats: 1″, 2/3″ and 1/2″.

Image formation. Generation of the image of an object on the camera's photo-detector array. The quality of this image depends on a variety of factors. See Image Acquisition.

Image intensifier. Electron tube, with a light-sensitive electron emitter at one end electron-accelerating electrodes and a phosphor screen at the other end. Acts as an "optical amplifier," for imaging in low-light conditions. Devices of this type are often built into extremely sensitive tube cameras, which give usable images on a cloudy, moonless night.

Image interpretation. Making decisions, based on the information obtained from Image Analysis. (q.v.) Relies heavily on Artificial Intelligence, Fuzzy Logic and Pattern Recognition techniques.

Image memory. High-speed, storage area in a frame-grabber card, or in a computer. Since it is dedicated to image retention, it may have certain special features, for example, the ability to output one image for processing, while another is being received from the camera.

Image orthicon. Camera tube. An optical image is focused on the rear of a photo-emissive cathode which emits photo-electrons at each point in proportion to the amount of light falling on it. These photo-electrons are accelerated and focused on a metal target, producing a secondary emission and creating a positive-charge image. The rear of the target is scanned by a low-velocity electron beam, which supplies it with just enough electrons to neutralise the charge image. The remaining electrons form a beam modulated in accordance to picture brightness that returns towards the electron gun on a path parallel with the outward beam. The return beam strikes the first dynode of a photo-multiplier to form the picture signal.

Image plane. Focal plane of the camera lens. perpendicular to the optical axis. The photo-detector is placed here, to obtain the sharpest possible image.

Image processing. See Digital Image Processing.

I*mage transform.* An image processing operator in which each pixel is replaced by a function of, many or, all of the pixels in the original image, e.g., autocorrelation.

Image. According to the Oxford English Dictionary "An optical appearance or counterpart of an object, such as is produced by rays of light either reflected as from a mirror, refracted as through a lens, or falling on a surface after passing through a small aperture." Also used when referring to a Digital Image (q.v.).

Incandescent lamp. Common light bulb. Simple cheap light source. Relies on hot filament heated by electric current. Gives directional illumination which causes shadows. Emits a lot of infra-red. Not commonly used in industrial applications.

Incident light. Light impinging on an object from the light source.

Incoherent illumination. Light in which the phase relation between nearby points in space varies randomly. See Coherent Illumination.

Industrial vision systems. General purpose systems can be configured by either the supplier or the user, for different tasks and in different industries. Special-purpose systems are designed for a particular task and are likely to be fast and efficient. However, they cannot be diverted easily to a different application. A system installed in a factory is called a *Target system*, to distinguish it from a laboratory-based (interactive) system used to design/optimise it.

Infra-red imaging. Image formation, using IR wavelengths, which are longer than those for visible red light but shorter than microwaves. (approximately 0.78 μm^{-1}.) See NIR, IR-A, IR-B, IR-C.

Integral transform. Type of image transform, analogous borrowed from classical mathematical/time-series analysis. Most common types: Discrete cosine Fourier, Walsh and Haar transforms.

Integration. Solid-state and CRT imagers accumulate photo-electric charge formed by irradiating the sensor during a time interval specified by timing signals. (i.e., the filed/frame period). The resultant signal is determined by the product of (a) the time-integral of the light level, (b) the area of the photo-detector element, (c) the exposure time and (d) the conversion efficiency.

Intensity. In optics, intensity measures the amount of light reflected from a surface. Intensity is not sensitive to colour. The term is also applied in image processing, to the values associated with pixels in a digital image. Intensity may be measured by a VIS, IR or UV camera, range (3D) sensor, x-ray sensor, or computed. Visual scientists often use a system of units (*photometric units*) that scale the physical power (*radiometric units*) by matching the spectral sensitivity of the eye.

Interferometer. Optical set-up, commonly used for very accurate measurement of distance. A beam of Coherent Light (q.v.) is divided into two separate beams, possibly using a beam splitter. These are then directed along different paths, before being reunited, thereby forming an interference pattern from which appropriate measurements are made.

Interlaced scanning. Method for scanning used in broadcast television. A Complete Frame (q.v.) consists of the odd-numbered lines (odd-field) followed by the even-numbered lines (even field).

Interline transfer. Type of CCD image sensor with special scanning circuitry to minimise image smear.

Intrascope. A flexible endoscope using fibre-optic bundle. Capable of navigating and imaging deep inside holes, crevices, etc.

Invariant hue. The perceived hues of monochromatic light that do not change with intensity. There are three invariant hues: blue, green, and yellow. (See Bezold-Brücke Hue Shift.)

IR-A. Infra-red radiation in the wave-band λ = 0.78–1.4 μm. Near IR (NIR).

IR-B. Infra-red radiation in the wave-band λ = 1.4–3.0 μm.

IR-C. Infra-red radiation in the wave-band λ = 3.0 μm–1 Far IR (FIR).

Iris. Mechanical diaphragm which can be controlled manually or automatically, to adjust the lens aperture. See Auto-Iris Lens.

Ishihara test. A test for colour blindness using a set of pseudo-isochromatic plates.

Isocon. High-sensitivity image orthicon.

Isoluminant. Synonym for Equiluminant (q.v.).

Isomerization. The change in conformation upon absorption of a photon that the chromophores in photo-pigments undergo.

*Isophot*e Intensity contour. Curve formed by joining together points of equal intensity in an image.

L

LAB. An acronym for the LAB colour co-ordinate system.

Lake. Imagine that a blob in a binary image is an island. A lake is an area of water totally enclosed by land. One type of constituent part of the Convex Deficiency (q.v.) of the blob.

Lambertian reflection. Diffuse light returned from an ideal matt surface.

Laplacian Operator. The sum of the magnitudes of the second partial-derivatives of the image intensity function. An approximation to the Laplacian operator is used to high-light spots, streaks and find edge regions of a grey-scale image.

Laser (Light Amplification by Stimulated Emission of Radiation). A laser produces a highly coherent, parallel beam of monochromatic light. It is used principally for range measurement and holography. Not normally used for wide-area “flood” lighting, since the high level of coherence causes speckle effects as the beam travels through air. Common types used: diode and HeNe laser. Safety precautions are important for high-power lasers.

Laser illumination. Lighting an object with a laser source for frequency selection, pulse width (strobe) control or for accurate positioning.

Laser Radar. See LIDAR.

Latency. See Processing Speed.

L-cone. One of three cone types that contribute to human colour vision. The L-cones have their peak spectral sensitivity at a longer wavelength than the other two cone types: *M-cones* and *S-cones.*

LED (Light emitting diode). Small, efficient, low-voltage, low-power, cold light source. Approximates to a point source of light. Often red but can be supplied at any VIS wave-length. Narrow spectrum light emission. Tri-colour (RGB) LEDs are available but their beams are not well structured and need external optical focussing. Varying the drive currents of the RGB diodes alters the subjective impression of colour but this does not sweep the spectrum as a monochromator does. Blue LEDs with overlying phosphors produce approximation to white light, with bluish tinge. LEDs can be used in a strobe, for medium-speed applications. Complex multi-LED arrays can be built for special applications.

Lens format. The approximate size of the image projected by a lens onto the Focal Plane. (q.v.) Usually, the lens projects an image slightly larger than the designated image size to ensure the sensor array is fully covered. The camera and lenses should both be of the same format size. A large format lens can be used on a small format camera. However, to avoid Vignetting (q.v.), a large format camera should not be used with a smaller format lens.

Lens speed. Term borrowed from photography. Actually, "speed" is misapplied when discussing to video, since the light-integration period is fixed, by the frame rate. See f-Number. (q.v.).

Lens. Transparent material, usually glass or plastic, with curved surfaces, to divert light rays, so that they either converge or diverge. Several lenses are often used in tandem, for controlling illumination and imaging. Common types: standard (35 mm), CCTV (non-specific meaning), cylindrical (magnifies in one direction only), enlarger (controlled magnification), condenser (collects light in ill-conditioned light beam to generate illumination in a controlled direction) in defined direction, macro (short working distance), micrographic (viewing small objects), video (non-specific meaning), zoom, telecentric (depth of focus is very large), auto-iris (self-adjusting aperture), telephoto (long working distance), wide-angle (viewing cone has wide angle), fish-eye (viewing cone angle 150–180°), Fresnel (moulded plastic lens with prismatic ridges), catadioptic (lens-mirror combination, light-weight, long-focus telephoto), aspherical (lens, often moulded, which ha non-spherical surfaces, to reduce aberrations), micro-lens array (array of minute lenses, each one slightly different. to achieve certain special effects).

LIDAR (Light Detection And Ranging). Similar system to radar but uses time of-flight of a pulsed laser beam instead of microwaves, for measuring range and mapping 3D surfaces.

Light Emitting Diode. See LED.

Light tent. Arrangement for Omni-Directional Lighting. (q.v.).

Lighting. Synonymous with Illumination. (q.v.).

Lightness. Synonym for Brightness. (q.v.).

Line of light. See Structured Light.

Line pairs per mm. Measurement of spatial frequency, used to describe the resolving power of a lens. Good standard lens can resolve 100 line-pairs mm^{-1}.

Line scan camera. Solid-state video camera consisting of a single row of pixels. Also called a linear-array camera.

Linear array. See Line Scan Camera.

Loca-area histogram equalisation. Non-linear, local-operator filter. Used principally for enhancing texture in a grey-scale image, by eliminating slow back-ground intensity changes. Has a serious effect on visibility of edges. Name is derived from the theoretical conceptual basis of the technique and has little in common with the procedure known as Histogram Equalisation. (q.v.).

Local operator. Image-processing operator in which each intensity value is replaced by a function of those pixel intensities found in a defined neighbourhood. For example, the local average, or median value, of the intensities might be computed for each 5×7-pixel area in a digital image.

Location. Address of a in a digital image Also, XY-coordinate position of a defined image feature (e.g., blob centroid).

Look-up table (LUT). Table of numbers held in RAM or ROM, used in hardware or software to speed up common functions such as Fourier transform, intensity mapping (e.g., logarithm, exponential, square, square-root), Gamma Correction (q.v.), Histogram Equalisation (q.v.), pseudo-colour image display and implementing finite state-machines. (See SKIPSM.)

Low-angle illumination. Useful to enhance minor surface irregularities such as pits, scratches, gouges. Also called glancing illumination.

Low-pass filter (LPF). Digital, analogue or optical filter which passes low frequencies, while attenuating high frequencies. When applied to images, LPF blurs edges, reduces the effects of noise and the contrast of small spots and narrow streaks. Regions where the intensity is changing slowly are unchanged.

Lumen. Unit of luminous flux. The amount of light emitted per second into unit solid angle (1 steradian) by a uniform point source with intensity equal to one Candela. (q.v.).

Luminance. The optical power weighted by the overall spectral sensitivity of the eye, per unit area of a light source.

Luminosity. See Brightness.

LUT. See Look-Up Table.

Lux. Unit of illumination. Equal to 1 lumen per square metre.

M

Machine Vision (MV). MV is concerned with the engineering of integrated mechanical-optical-electronic-software systems for examining natural objects and materials, human artifacts and manufacturing processes, in order to detect defects and improve quality, operating efficiency and the safety of both products and processes. It is also used to control machines used in manufacturing.

Machine Vision system. (*MVs*, See Vision System.)

Magnification (of a lens). Ratio of the length of a linear feature of a feature in the image plane to the length of the same feature in the object plane.

Manual iris lens. Lens with a manual adjustment for the iris opening (aperture), to set a fixed f-Number. (q.v.) Normally, used in fixed lighting conditions.

Mask. Has several meanings: (a) Pixels covered by the mask are transformed in some pre-defined way (e.g., set to a constant value), while other pixels are unchanged. (b) Pixels covered by the (rectangular) mask are removed. (Called cropping.) (c) Synonymous with Weight Matrix. (q.v.) (d) Physical structure placed in an optical system to block the transmission of certain light rays. (e) In an optical system forming a Fourier Transform (q.v.), unwanted spatial frequencies can be eliminated by masking as in (d).

Mathematical morphology, Also called Morphology. Non-linear local or N-tuple operator. The input image (binary or grey scale) is scanned by overlaying a small template shape, called a Structuring Element. (q.v.). The output image indicates how the structuring element fits, or does not fit, at each point within the input image. See SKIPSM.

Matrix-matrix mapping. General class of image processing operations in which one digital image matrix (or array) is transformed into another. Common types: monadic (one input image), dyadic (two input images), local operator/N-tuple, Mathematical Morphology (q.v.), global.

M-cone One of the three cone types contributing to human colour vision. The peak spectral sensitivity of M-cones lies those of the other two cone types: L-cones and S-cones.

Mechanical handling system. Hardware that ensures that objects/materials pass the camera in a controlled manner. Motion may be linear or rotational, indexed or continuous. Occasionally, the camera is moved past a stationary object. Sometimes, the object and camera are both stationary. Panning is then achieved using scanning mirrors, mounted on gimbals.

Median Filter. Non-linear filtering method for reducing camera-noise and other noise-like effects in a image. Each pixel intensity is replaced by the median value of the intensities found among pixels immediately surrounding it, in a fixed neighbourhood.

Metamerism. Inability of the human eye to distinguish colours with quite different spectral compositions. For example, light with a 2-peak spectrum may produce a similar colour sensation to narrow-band radiation with a single spectral peak.

Metrology. The science of measurement.

Michelson interferometer. Optical instrument in which a monochromatic light beam is split into two component beams. These are reflected from separate mirrors to be reunited at the beam splitter where they combine to produce an interference pattern.

Micron (micrometer, μm). Unit of length: 10^{-6} m, 10^4 Angstroms. Frequently used to measure EM wave-length.

Microspectrophotometry. A technique for obtaining measurements of the spectral absorption of a single photo-receptor cell. Useful as a means for investigating colour vision without behavioural tests.

Minimum object distance (MoD). Shortest lens-object distance that gives a sharp focus. The smaller the focal length of a lens, the shorter is its MoD, although this can be altered with use of Extension Tubes (q.v.) or Add-On Lenses (q.v.). See Diopter.

MIPS (Millions of Instructions per second). Measure of computer processing speed. Usually a more meaningful indication of "real" speed than raw clock speed.

Mired (micro reciprocal degree). Unit used to measure the reciprocal of colour temperature; 1 mired corresponds to a surface having a colour temperature of 10 K.

Mirror. Smooth, highly polished surface, for reflecting light. It may be plane or curved, in one or two dimensions. Also see Anamorphic Mirror. Mirrors are fabricated by depositing a thin coating (e.g., silver, aluminium or gold) on a glass or plastic substrate. To produce optimal image quality, avoiding ghost images due to reflections from the uncoated surface, first-surface mirrors are coated on the front face. Unlike domestic mirrors which are rear-coated, first-surface coating is easily damaged by rough handling. Mirror-like surfaces on an object can cause serious problems for an inspection system, due to glinting (specular reflection). Cheap plastic mirrors are useful because they can be machined easily. Prisms are sometimes used in lieu of mirrors, particularly where precise alignment of two orthogonal mirrors is required. Very large. low-precision mirrors can be made by stretching aluminised Mylar film over a suitable frame.

Modulation Transfer Function (MTF). When a 1-dimensional sine-wave pattern is passed through an imperfect optical system, the contrast in the resulting image is a function of the period of that pattern. The MTF is the contrast as a function of period and represents the ability of an optical system to reproduce various levels of detail. For practical convenience, a set of parallel black-white bars is sometimes used instead of a sinusoidal test pattern.

Moiré fringes or patterns. Pattern which appears when two nearly-identical sets of parallel lines, arrays of spots, concentric circles, etc. are superimposed. Effect seen on shot silk. Moiré patterns are an artefact of human perception and should not be confused with interference fringes caused by the finite wave-length of light.

Moiré interferometry. Experimental method used to obtain 3D profile information of an object or surface. Two identical patterns (usually sets of parallel lines) of known pitch are used. The first creates a shadow on the object. The second is placed in the imaging path, and is therefore superimposed on the shadow cast by the first. This forms a Moiré pattern, in which distance between the dark bands (fringes) is directly related to object-camera range. However, it is impossible to determine whether a surface is convex or concave from a single Moiré pattern. This ambiguity can be resolved by varying the inter-line spacing, in two separate experiments, to change the sensitivity.

Monadic point-by-point operator. Image processing operator which acts on only one image. Each pixel in the transformed image is obtained from operations on only the corresponding pixel in the original image.

Monochromacy. The condition of possessing only a single channel for conveying information about colour. Monochromats are strictly colour blind and perceive only variations in brightness.

Monochromatic. Refers to either light, or the scene being viewed. Monochromatic light contains electro-magnetic radiation of a single wavelength (e.g., light from a laser). The term is also used to describe light having a narrow range of wavelengths. For example, that pair of narrow spectral bands giving light from a sodium lamp its characteristic yellow colour would be referred to as being monochromatic. Surfaces that reflect/emit light with a narrower spectrum would also be referred to as being monochromatic.

Monochrome. Used to refer to images, whereas monochromatic refers to light, or the scene being viewed. A monochrome image contains no information about colour. The signal from an RGB colour camera generates three separate images, each of which is referred to as being monochrome. In effect, these measure the amount of red, green and blue light in a coloured scene. A colour image can be represented, for the purposes of display, printing, or digital processing, by three or four monochrome images, called its colour separations, or channels.

Morphology. See Mathematical Morphology.

MRI Magnetic resonance imaging. A non-invasive technique for 3D imaging MRI has been mostly used for detecting and localizing damage to the brain and other soft body tissues. Potentially very useful for MV but very expensive.

Munsell colour system. A widely used system for describing colour appearance. The sample is matched by eye against a set of stable reference swatches. It may be necessary to use interpolation to classify a given test sample.

M×N neighbourhood. Contiguous set of pixels in an image, arranged within an M×N rectangle. Their intensity values might be combined in some way, for example by computing their average. See 4-Neighbour and 8-Neighbour.

N

Nanometer One billionth of a meter. (10^{-9}m or 0.001μm). Practical unit for measuring the wavelength of light and atomic spacing.

Neural Network. Principal approach to Statistical Pattern Recognition. Abstract computing structure which processes information based on a crude model of biological nervous systems. Decisions are made about a complex event, pattern or image, based on a set of parallel numeric inputs. Each neuron in the network, which may consist of several interconnected layers, forms a threshold linear-weighted sum of its inputs. The decision surface of a multi-layer network is piece-wise linear. Self-adaptive, error correcting learning rules are known, which allow a network to optimise its performance on a classification task that is specified only by a set of labelled samples.

Neutral. Used to signify an object or scene that has is composed only of grey, white or black regions. It does not have any colours such as yellow, red, etc.

Newvicon. Type of Vidicon (q.v.) with a double layer target, giving an improved resolution

NIR. See IR-A.

Noise. Irrelevant or meaningless data resulting from various causes unrelated to the prime application requirements. In Machine Vision, noise is caused by varying ambient light (e.g., sun-light, fluorescent tubes providing environmental lighting), dirt on the Widget, optics or light source, as well as the usual sources of noise in electronic equipment: earth loops, EM radiation, capacitative coupling, induction, etc.

Normalisation Process of adjusting the output range of an image processing operation so that is identical with that at its input. By normalising in this way, image processing modules/ routines can be concatenated, in any order without causing overflow.

N-tuple operator. Image processing operator in which each pixel is replaced by a value derived by evaluating a function of only a selected few (N) of its neighbouring pixels. See Local Operator and Mathematical Morphology.

Numerical aperture (NA). Measure of the resolving power of a lens. Equal to $\mu \sin(U)$, where μ is the refractive index of the medium adjacent to the objective lens and U is the half-angle of the maximum cone of light picked up the lens. If the object is at infinity, NA = 0.5/ (f-number).

NUV. See UV-A.

Nyquist limit. Spatial frequency equal to half the sampling frequency. If an image falls on an array of photo-detecting elements with spacing D µm, the image is said to be sampled at a spatial frequency of 1/D and the Nyquist limit is 1/(2D). Using this array it would be impossible to discern objects of diameter 2D µm, or less, reliably. Similarly, if a pattern of parallel lines of width less than 2D and spacing 2D were projected onto this sensor array, the image would be unclear. (Moiré Patterns (q.v.) would be formed.) In practice, a higher sampling frequency than 1/(2D) is required to avoid this effect. The Nyquist limit also applies to temporal sampling; sub-Nyquist sampling is responsible for wheels appearing to rotate back-wards on television and ciné film.

O

Object plane. Plane through the object being examined (the widget), normal to the camera's optical axis. Features in the object plane are focused by the optical system onto the image plane and hence are projected onto the photo-sensor array.

Object. The item to be measured, counted, sorted, inspected or manipulated. To avoid ambiguity, the specific term "Widget" (q.v.) is preferred.

Objective. Element of a compound lens that is closest to the object being examined (i.e., widget).

Oblique illumination. Grazing illumination; angle relative to camera's optical axis is a little less than 90°. Emphasises object features by producing shadows.

OCR (Optical character recognition, or Character Recognition). Recognising (i.e., "reading") previously unknown characters on a document, packet, etc. Cheap OCR software is available for standard flat-bed scanners and desk-top computers. Also see OCV.

OCV (Optical Character verification, or Character Verification). Verifying the correctness, quality and legibility of *known* printed characters on a document, packet etc. Unlike OCR (q.v.), the character sequence is known beforehand.

Off-the-shelf. Multi-purpose Machine Vision system which is not pre-configured, or optimised, for any specific application.

Oil mist. Serious environmental contaminant for optical surfaces.

Omni-directional lighting. All round lighting. Light impinging on the object comes from all points on an hemisphere.

Opening. Non-linear image filtering (morphology) operator, equivalent to an erosion operation followed by dilation, Has the effect of removing small spots and narrow streaks. The structuring element determines the shape and sizes of the features that are removed.

Opponent process theory. Based on the hypothesis that subjective colour appearances are the result of recoding the outputs of the eye's photo-receptors (cones) into three channels: two opponent chromatic channels (*yellow-blue* and *red-green*). In addition, there is one achromatic opponent channel (*white-black*). These three channels are created by forming linear combinations of the outputs of the three types of cone in the retina. The theory correctly predicts that the blueness (redness) of a source may be cancelled by adding to it light that enhances its yellowness (greenness).

Opsin. The protein component of a photo-pigment in the eye. Different opsins give rise to photo-pigments with different spectral absorption characteristics. (Also see Chromophore.)

Optical Computing. To date, only limited functionality can be implemented efficiently in an optical computer. While it is possible to form the Fourier Transform (q.v.) of a complete image on a transparency in parallel in real time at the speed of light, seemingly simpler operations, such finding the centroid of a blob, presents considerable difficulty. Optical computing is an attractive but elusive ideal for researchers in Machine Vision.

Optical transfer function. See Modulation Transfer Function (MTF).

Orientation. The angle or degree of difference between the object co-ordinate system major axis relative to a reference axis as defined in a 3D measurement space.

Orthicon. Type of CRT camera in which a beam of low-velocity electrons scans a photo-emissive mosaic that is capable of storing a pattern of electric charges.

Oscillating mirror scanner. Beam-steering element in a flying-spot scanner uses a laser beam projected onto an oscillating mirror.

P

Panchromatic. Photographic film, solid-state sensor array or camera tube that is sensitive to all visible wavelengths. The term is applied even if the response is not constant for all VIS wave-lengths.

Pantone. Colour matching system in which colours are labelled by numbers.

Parallax. Change in the apparent positions of two objects when viewed from different vantage points. The nearer object appear to move in the opposite direction to the observer.

Parallel processor. Computer which has a large number of identical processors each of which can operate on data (such as a complete row or image) at the same time.

Pattern Recognition. See Neural Network. Many other computational methods exist for learning to make decisions, when there is no explicit formula, or algorithm, for doing so.

Perceptron. Basic processing element used in neural Networks. (q.v.) Forms a thresholded linear-weighted sum of its inputs.

Phase contrast microscope. Compound microscope that has an annular diaphragm in the front focal plane of the sub-stage condenser and a phase plate at the rear focal plane of the objective. This makes visible differences in the optical path (i.e., path length, or refractive index) in transparent or reflecting media.

Phosphor. Material that glows visibly in response to some form of non-visible energy supplied to it, by for example, the electron beam in a cathode ray tube, radio-active material (gamma-ray), UV or x-ray source. Some materials continue to glow for a short period after the source of energy has been removed. This can be useful for distinguishing phosphorescence and Fluorescence. (q.v.).

Photodiode array. Solid-state array in which the photosensitive elements consist of photodiodes.

Photodiode. Reverse-biased semiconductor diode in which incident photons create hole-electron pairs, thus creating a current that is proportional to the level of illumination.

Photometry. Measurement of light which is visible to the human eye.

Photomultiplier. Very sensitive light-detector, capable of identifying the arrival of a single photon. It consists of a cold light-sensitive cathode, together with a series of electron multipliers, within an evacuated glass envelope.

Photophore. Light emitting (bioluminescent) organ found in animals such as fire-flies and deep-sea fish.

Photopic response. The colour response of the eye's retinal cones.

Photopic vision. Vision under relatively high light levels when the outputs from the cones are dominant.

Photo-pigment. Pigment molecules in a photo-receptor (cone) in the eye undergo changes when they absorb photons. This causes the photo-receptor cell to "fire," sending a signal into the surrounding network of neurons. Each photo-pigment tends to absorb photons within a specific range of wavelengths, thereby giving rise to the different spectral sensitivity characteristics of the cones.

Photo-receptor. Light-sensitive neuron in the retina of a human or animal eye. Photo-receptors interact with light, causing them to "fire" and generate an electrical signal, which is then transmitted to other neurons nearby. The human retina contains two broad classes of photo-receptors: rods (for monochrome night vision) and cones (for colour vision in high light levels).

Pin-cushion distortion. Imperfection of an optical system in which a rectangle appears "pinched" on all sides. Opposite of Barrel Distortion. (q.v.).

Pin-hole camera. Simple image-formation technique in which a single small hole is formed in an opaque screen. If the hole is large enough to avoid diffraction effects, rays pass through the hole onto the camera's photo-detector array. Simple to analyse the image-formation process by straightforward ray geometry. Unlike glass lenses, the pin-hole produces no aberrations (ignoring diffraction) but is very inefficient at light gathering.

Pipe-line image processor. Data processing architecture, which can perform a series of operations simultaneously on different items in a stream of data, thereby achieving considerable speed-up, compared to a serial processor.

Pixel (Pel, or Picture element). Smallest addressable portion of a digital image. Within the boundaries of a given pixel, the intensity and colour can be considered constant.

Pixel counting. Simple technique for characterising objects in a binary image, by counting the number of pixels contained within its boundaries. (Equivalent to finding the blob area.)

Plumbicon (leddicon). Form of vidicon in which the target consists of a photo-conductive layer of lead monoxide. Once used for broadcast-quality colour cameras.

Polar vector representation (PVR). Code used for describing a curve, such as the periphery of an object. Each "break point" on a piece-wise linear arc is represented by the polar co-ordinates of the next point with the current point as reference. Thus a PVR sequence consists of a series of the form $[(R_1, \theta_1), (R_2, \theta_2), (R_3, \theta_3),..., (R_n, \theta_n)]$, where R_i $(i \leq n)$ denotes the length of the ith linear edge segment and θ_i its orientation.

Polariser/Polarised light. Linearly polarised light has its E-vector (electric-field vector) lying in one plane, perpendicular to its direction of travel. It is created by a type of filter (polariser) which absorbs the light rays polarised in the orthogonal direction. Polarisers are usually fabricated from stretched plastic sheets with aligned, parallel bi-refringent crystals. Polarizers can also be constructed using parallel wires, or arrays of very fine conductors formed by metal-evaporation techniques on a glass substrate. A prism with light striking its face at the Brewster Angle (q.v.) is polarised. Circularly-polarised light also has its E-vector lying in a plane normal to the direction of travel but it rotates as it travels.

Polarization. Refers to the spatial orientation of the electric-field vector of an electromagnetic wave. If all rays have the spatial orientation of this vector, the beam is said to be polarised. Most light sources, such as the sun, incandescent lamps, discharge tubes, fluorescent lamps, flames, etc. emit non-polarised waves. Certain physical processes, such as scattering in the atmosphere and reflections from glossy surfaces, achieve partial polarisation. Optical-quality polarisers achieve a far purer polarisation effect.

Polychromatic scene. Contains a finite number of Monochromatic Regions (q.v.), each one being clearly distinct from all of the others. This is a specialised use of the term in this book. Whereas Chrominance (q.v.) is continuous in a Pan-Chromatic Scene (q.v.), it assumes only a finite number of discrete values in a polychromatic scene.

Positioning equipment. Electro-mechanical device used to bring the widget into the camera's field of view, or to move it when multiple or views are required.

Positron emission tomography (PET). A method for medical imaging that uses tracers that emit positrons. The tracer is introduced into the subject's blood and then its concentration is measured by collecting the emitted positrons. PET is currently too expensive for use in Machine Vision.

Post-objective scanner. Type of flying-spot scanner in which the scanning element is placed after the objective lens, which focuses the beam into a spot.

Precision. Accuracy claimed for a measurement. We may be able to estimate the position of a feature, such as the centroid of a blob in a given binary image, with a precision that is better than one pixel. However, this is a spurious assertion if quantisation effects lead to much larger changes in this parameter as we move the widget slightly. See Repeatability.

Pre-objective scanner. Type of flying spot scanner in which the scanning element is placed before the objective lens, which focuses the beam into a spot.

Pre-position lenses. Pre-position lenses are specially designed with facilities for computer control of aperture, zoom and focus, without operator intervention.

Prewitt. Name of a type of Edge Detector. (q.v.).

Primary. Light from a set of primary light sources can be added together to produce any other colour. For human beings with normal vision, three primaries are sufficient. However, there are indefinitely many such sets of three. For the purpose of defining the CIE Standard Observer (q.v.) and for precise colourimetry, the concept has been extended to include "imaginary" primary light source that are not physically realizable.

Primary visual cortex. An area of the occipital lobe of the human brain that performs the first stage of cortical visual processing. It receives inputs from the retina via the Dorsal lateral geniculate nucleus and sends outputs to other areas of the visual cortex. Also referred to as V1, striate cortex, and area 17.

Principal planes. Two planes passing through the principal points of an optical system, normal to its axis.

Principal points. Two points on the axis of an optical system. If the object distance (U) is measured from one and the image distance (V) from the other, the equations relating U and V are mathematically identical to those for a thin lens.

Prism. Optical device with two or more polished faces from which light is either reflected or refracted. Many types of prism exist, of which the triangular form, used for decomposing light into its spectral components, is the most familiar.

Processing speed. Loose term that is inaccurately used to measure the throughput rate of a vision system. The latter is normally expressed in parts-per-minute. This single parameter does not give any indication of the time required by a vision system to receive, analyse and interpret image information for an individual piece-part. The latter is called *latency* and may be considerably larger than the reciprocal of the throughput rate. (Parallel and Concurrent Processors (q.v.) are able to process image data from more than one widget at a time.)

Profile. The 3D contour of an object.

Programmable Colour Filter (PCF). Real-time video filtering device, consisting of a look-up table, implemented in a random access memory, whose inputs are the digitised RGB signals. The contents of the look-up table are generated from a monochrome image. The output of a PCF is a monochrome video image, but this may often be displayed to good effect using pseudo-colour.

Protanopia. One of the two varieties of red-green colour blindness (also known as red-dichromacy). Protanopia is thought to result from the loss of function of the L-Cones. (q.v.) People with this condition display a marked loss of sensitivity to light at the long wavelength end of the spectrum. (Also see Deuteranopia.)

Pseudo colour-triangle. Computer-generated image, superimposed onto the colour triangle. A series of very useful programmable Colour Filters (q.v.) can be generated by creating pseudo colour-triangles, using an image processor, or simple graphics program.

Pseudo-isochromatic plate. An array of coloured dots for testing for colour blindness. A normal subject will see a certain figure in the pattern, which is apparent to a person with a colour vision deficiency.

P-tuple (or N-tuple) operator. An image processing operator in which each pixel is replaced by a function of only a selected few (P) of its neighbouring pixels.

Purity. A pure light source emits monochromatic radiation. If this is mixed with light that has a broad spectrum, the purity is reduced. Measured by Saturation. (q.v.).

Q

Quad-tree. Hierarchical representation of a high-resolution image. It consists of a series of lower-resolution images, each of which is obtained by merging compact regions in the next highest layer. Useful for speeding up certain image processing calculations.

Quartz-halogen-tungsten lamp. Small intense source of white light produced by an electrically-heated tungsten filament, enclosed in a quartz envelope which contains a halogen vapour (e.g., iodine) at low pressure.

R

Radiance. Measurement of radiant flux, per unit solid angle, per unit of projected area of the source. (The usual unit is the watt steradian^{-1} m^{-2}.)

Radiometry. Measurement of light within an optical spectrum.

Random access. The ability to access chosen pixels, lines or ROIs (q.v.) from an imager as needed asynchronously, without waiting for the appropriate moment in a scanning cycle.

Range map. See Height Map.

Raster scan. Image/scene scanning pattern, usually from left to right, while progressing more slowly from top to bottom. Often comprised of two fields composed of odd-numbered and even-numbered lines.

Rayleigh criterion. Mathematical condition which states that the resolution of a diffraction-limited optical system of aperture D used to view an object using light of wave-length λ is given by $1.22\lambda/D$.

Rayleigh scattering. Scattering due to numerous small particles of uniform size in suspension. The intensity of Rayleigh-scattered light is proportional to λ^{-4} (λ = wavelength).

Real time processing. The popular and mistaken view is that this refers to the ability of an inspection system to complete the analysis and take appropriate action on one widget before the next one arrives for inspection. This is an incomplete view, as it does not distinguish between throughput rate and latency. (See Processing Speed.) The concept of real-time cannot be defined without reference to the application requirements. The requirements for a real-time robot control system are more straightforward; we need only specify the system band-width, or the maximum speed of movement of the robot.

Reflectance (of a surface). The proportion of incident light that a surface reflects. This is a function of wavelength. (See Spectral Reflectance.)

Reflection. (a) Specular. Optical process by which a light beam incident on a smooth conducting surface causes another beam to leave it on the same side. The reflected beam is generated by the surface so that its E-vector cancels that of the incident beam. Hence, the polarisation is unchanged. (b) Diffuse (Lambertian). The energy within the incident light beam is scattered through a solid angle of $4\pi/3$ steradian.

Polarisation (q.v.) and Coherence (q.v.) are both lost. This is a typical effect of light striking a rough dielectric surface. In practice, light reflected from surfaces such as plastic, polished wood, paint, etc. has a strong specular component and exhibits diffusion as well.

Refraction. Change of direction of a ray of light when passing from one transparent medium to another.

Refractive Index. Property of a medium that measures the degree that light bends when passing between it and a vacuum. In a material of high refractive index light travels more slowly than in one which has a low value.

Region of interest ROI. Area (usually rectangular) within an image in which an image processing operator is applied. To save time, computational effort is not applied elsewhere.

Registration. (a) Refers to the relationship between the actual position of a widget and the nominal position expected for image acquisition. Accurate registration requires that the X, Y and θ co-ordinates (orientation) all be controlled and significantly reduces computational effort and processing time for inspection. (b) When two or more images of a scene are obtained from slightly different views, joining them together to form a mosaic requires that they be registered. Many of the impressive images of the surfaces of Moon, Mars, etc. that are released by NASA are mosaics composed of accurately registered subimages. For practical reasons, the construction of mosaics is not normally used in Machine Vision.

Reject mechanism. Device used on a manufacturing line to remove defective widgets from the product conveyor. This may employ an air blast, a solenoid actuated pusher, an electromagnet (for ferrous widgets), or a flipper-like deflecting plate. The unceremonious dumping of faulty widgets into a reject bin may mean that they are further damaged, making detailed examination by a human inspector impossible. In this event, the vision system must act without human back-up.

Remote prototyping. Principle of designing an inspection system without the vision engineer visiting the factory where it will be installed. See Flexible Inspection Cell.

Repeatability. Ability of a system to reproduce the same measurement results for an identical widget on different occasions. See Precision.

Resolution. (a) Digital image: the number of rows and columns. (b) Intensity scale: the number of resolvable shades of grey. (c) Optical system: the smallest angular separation of two object points which produces two distinguishable image points.

Reticle. Optical element used for calibration, or instrument alignment. Usually, a glass plate with a pattern consisting of rulers, crossed lines, grids, arrays of spots, concentric circles, etc. Familiar to microscopists but not often used with Machine Vision systems, since the reticle image is superimposed onto that of the widget. Similar effect can be obtained using software, if the camera has previously been calibrated.

Retina. (a) The inside layer of the back of the eye that contains the photo-receptors and associated neurons. The earliest stages of visual processing take place in the neurons of the retina. (b) Light-sensitive surface of a video camera that converts a visual pattern into an electronic image.

Retinal. (a) Relating to the Retina. (q.v.) (b) The Chromophore (q.v.) contained in human photo-pigments.

Retrofocus lens. Compound lens system consisting of a diverging lens followed by a converging lens (reverse of telephoto principle). This gives a back focal length which is greater than the true focal length, facilitating convenient camera design.

Retroreflective material. Sheet material with a coating which returns a beam of light along its path of incidence. Available as an adhesive tape from automobile accessory shops.

RGB (Red, Green and Blue). Standard solid-state and CRT colour video cameras use three sets of photo-detectors, behind red, green and blue optical filters. Hence, both types of camera generate RGB signals naturally. Although some cameras have additional electronics which converts these signals to HSI, this does not add any information. Hence, any calculation that is possible in HSI, or any other, colour-co-ordinate system can also be performed in RGB space.

Rhodopsin. The photo-pigment in the rod photo-sensors in the eye.

Ring-light. Bundle of optical fibres, or a set of small lamps or LEDs, arranged around the perimeter of a circle. Light may be projected forwards (i.e., normal to the plane containing the circle), radially inwards (i.e., towards the centre), or radially outwards (i.e., away from the centre)

Roberts. Name of a type of Edge Detector. (q.v.).

Robot. According to the Robot Institute of America (1979), a robot is "a reprogrammable multi-functional manipulator designed to move material, parts, tools or specialised devices through variable programmed motions for the performance of a variety of tasks." Term is also used to refer to autonomously guided vehicles.

Rod. One of the two main classes of photo-receptor found in the vertebrate eye. Rods are very sensitive to low light levels. However, they have only a minor effect on colour vision in daylight.

ROI (Region of interest). Portion of an image that is to be analysed.

Rotating mirror drum scanner. Type of scanning element in a flying-spot scanner in which the scanning is performed mechanically by reflecting a laser beam from the facets of a rotating polygonal drum.

Rotation. (a) Widget is sometimes rotated in front of a line-scan camera, to obtain polar or cylindrical co-ordinate scan pattern. (b) Image rotation is a surprisingly difficult calculation requiring interpolation. Best avoided if possible, to avoid quantisation artefacts around edges.

RS-170. Electronic Industries Association (EIA) standard governing monochrome video signals. Provides 30 complete images (frames, two fields) per second. Widely used in North America and Japan.

RS-232-C. Electronic Industries Association (EIA) standard governing serial communications over a twisted-wire pair. Can provide effective low-speed, 2-way, digital communication for up to about 50 m.

RS-330. Electronic Industries Association (EIA) standard governing monochrome video signals. Provides 25 complete images (frames, 2 fields) per second. Equivalent to CCIR. (q.v.), which is widely used in Europe.

RS-422; RS-423; RS-449. Electronic Industries Association (EIA) protocols for low-speed, serial, digital communication. Intended to replace the widely used RS-232-C Standard (q.v.) eventually.

Run code (or Run-length encoding). Representation of a binary image, in which each chord (i.e., segment of a row in which the intensity does not change), is represented by the number of pixels in it. It consists of a sequence of the form: [($R_{1,1}$, $R_{1,2}$, $R_{1,3}$,. . .), ($R_{2,1}$, $R_{2,2}$, $R_{2,3}$,. . .), ($R_{3,1}$, $R_{3,2}$, $R_{3,3}$,. . .), ...], where $R_{i,j}$ represents a black (j odd) or white (j even) chord in the ith row. This is an efficient coding, if the image contains just a few simple blob-like objects. The idea can be extended to cover grey-scale and colour images but the efficiency is usually poor.

S

Salt-and-pepper- noise. (a) Noise in a binary image that consists of occasional random inversion of intensity. (b) Similar effect occurring in grey-scale images, as a result of electrical noise disturbing the digitisation process.

Saticon. Type of vidicon with high resolution.

Saturation. (a) Colourfulness, or strength of colour. Highly saturated red contains only light from the red part of the spectrum and is said to be of high purity. Pink and other pastels colours are non-saturated and have a considerable amount of white light present (b) Popular colour co-ordinate used in Colour Science, usually specified together with Hue and Intensity. (q.v. Also see HSI.)

Scanner. See Flying-Spot Laser Scanner.

Scattering. Redirection of light by diffuse reflection from a surface, by dispersion as it travels through a transparent or fluid medium. See Diffuse Reflection and Dispersion.

Scene Analysis. Extracting features and deriving measurements from images. Intimately linked to image processing; there is no clear demarcation between the two.

Scene. The object being examined (called a widget) and its background; that portion of space observed by a vision system for inspection, measurement, identification, etc.

S-cone. One of the three types of cone that contribute to human colour vision. The peak spectral sensitivity of the S-cones is at a longer wavelength than that of L-Cones and M-Cones.

Scotopic vision. Human vision under relatively low light levels when the rods in the retina produce the most significant signals.

Segmentation. Computational procedure that divides an image into its constituent parts, usually based on some quality such as intensity, colour, texture, or a combination of these. Segmentation is a classification process, which identifies regions in the image as being similar, if they share some defined criterion.

Shading. The variation of the brightness or relative illumination over the surface of an object, often caused by colour or reflectance variations or surface curvature.

Shape factor. Convenient shape parameter. Ratio of the area of a blob to the square of its perimeter. Maximum value is $1/(4\pi)$ (= 0.079577) for a circle.

Shape from shading. 3D shape-analysis techniques that attempt to reconstruct an object's shape from the variations of intensity across its visible surface. This is only possible if the position of the light source is known and certain assumptions are made about the optical properties of the object surface.

Shape. (a) Physical object that can be adequately represented by a blob-like figure in a binary image. There is no internal detail of any importance. (b) Abstract quality relating to a 2-dimensional blob-like object, or the silhouette of a 3D object. Many parameters describing shape have been proposed, often simply to improve computational convenience. However, shape cannot be described adequately using just a few parameters.

Sharpening. Image processing operation which enhances edges. Can be achieved by unsharp masking, i.e., subtract a Low-Pass Filtered Image (q.v.) from the original.

Shutter. Electrical, mechanical or liquid-crystal device, used to control the timing and duration of the interval when the camera's photo-detector array is exposed to light. A fast shutter can be used to eliminate motion blur, without the risks inherent in Strobed Lighting. (q.v.).

Silhouette. High-contrast grey-scale image, usually obtained by illuminating an opaque widget by back-lighting. Has a strongly bimodal distribution.

Simple lens. Has a single element.

Simultaneous contrast. Phenomenon of human vision in which the perceived colour of an object tends to take on a hue opposite to that of the surrounding area. For example, a grey figure on a red background will appear to have a greenish tint.

Size. Ill-defined characteristic of an object. The size of a blob-like object may be characterised in various ways: area, dimensions of its minimum enclosing rectangle, diameter of its circumcircle, area of its convex hull, etc.

SKIPSM. (***S**eparated **K**ernel **I**mage **P**rocessing by* [*finite **S**tate **M**achines.*]) Fast implementation of a range of morphological and other operators. It is able to perform all operations in the same time, irrespective of the size of the structuring element.

Smart camera. Self-contained vision system contained in the camera body, including lens, image sensor, image processing and decision making functions, interface electronics and software.

Snow. Type of noise in a grey-scale or colour image, with the visual appearance of snow. See Salt-and-Pepper-Noise.

Solid-state array. Type of image sensor fabricated normally in the form of a linear, or rectangular, array of photosensitive elements on a single integrated circuit. See Charge Coupled Device (CCD).

Spatial contrast. Difference in some visual property (e.g., brightness, colour or texture) between adjacent areas of a scene or image. Black objects on a white background display high spatial contrast.

Spatial frequency. A measure of how rapidly some visual property (e.g., brightness or colour) changes in space. Images with fine detail (e.g., small spots, thin lines or texture), or sharp edges, have high spatial frequency. Those where changes are more gradual have low spatial frequency. See Modulation Transfer Function.

Spatial Light Modulator (SLM). Transparent (usually LCD) screen used in optical computing systems to introduce an image into the image-processing path. It may be used as the basis for forming the Fourier Transform (q.v.) of an image, os a Mask. (q.v).

Speckle. Phenomenon in which the scattering of Coherent Light (q.v.) by a rough surface, or inhomogeneous transmission medium (e.g., turbulent air), generates an interference pattern of random intensity distribution, giving the surface a granular appearance. Small-scale temperature changes in air at room temperature are sufficient to cause phase changes that will produce speckle patterns.

Spectral analysis. Decomposition and measurement of the energy content of a light beam as a function of wave-length.

Spectral characteristics. That spectrum characterising a light beam. It specifies the energy content as a function of wave-length.

Spectral colour/light. Colour of a monochromatic, or nearly monochromatic, light, When white light is dispersed by a grating or prism, the resulting “rainbow” is composed entirely of spectral colours. Colours with a wide spectrum, are said to be *extra-spectral.*

Spectral power distribution. Energy content of an optical signal, measured as a function of wavelength.

Spectral reflectance. The Spectral Power Distribution (q.v.) of light reflected from a surface. Varies with angles of incidence and viewing.

Spectral response. Output signal level of a light sensor as a function of wave-length of the light incident on it.

Spectral sensitivity. Sensitivity of a photo-receptor/photo-detector expressed as a function of wavelength.

Spectrum. Electro-magnetic spectrum, or EM spectrum. Expressed as optical energy as a function of wave-length. The visible (VIS) wave-band for human beings is normally taken to be 0.4–0.7 μm. Outside these limits the response is so low that it rarely has a significant effect. Many animals respond to light outside this range, especially shorter wavelengths. See VIS, IR, UV and x-rays.

Specular reflection. Reflection of light (e.g., from a smooth polished or metallic surface) in which the angle of incidence is equal to the angle of reflection. This gives rise to Glinting. (q.v.).

Speed. Image blur may be caused by high widget-camera speeds. Stroboscopic illumination or high-speed shutters can be used to "freeze" motion. Can also use line-scan camera, or oscillating mirror in front of the camera to compensate for motion.

Spherical aberration. Optical aberration produced by lenses with spherical surfaces. Rays of light parallel to the optic axis, but at different distances from it, are focussed at different points along it.

SRI algorithms. (Stanford Research Institute,1970s) Set of routines for analysis and identification of blob-like objects. Main steps are: (a) Form a binary image. (b) Isolate and identify each blob. (c) Measure parameters of each blob in turn. (d) Perform additional measurements, defined by the user.

Stadimetry. Range-measurement technique, based on the apparent size of an object known size in the camera's field-of-view.

Statistical Pattern Recognition. The pattern to be classified (e.g., an image or feature within an image) is represented by a fixed number of parallel measurements (*descriptors*), forming a *pattern vector.* The objective is to design a device (*classifier*) that can predict the output that would be obtained from a *teacher* were it available. The teacher defines the correct labelling for each pattern. The classifier must be designed by analysing a finite set of pattern vectors (*training set*), each of which is labelled by the teacher. Each label indicates to which one of a small number of classes the original pattern belongs. For example, the desired classification might simply be "*acceptable*" or "*faulty.*" No other information is available to guide the design of the classifier. Thereafter, the teacher is not available and the classifier takes over the labelling process. See Neural Network.

Stereo Photogrammetry. See Shape from Shading.

Stereoscopic imaging (Stereo). (a) Passive stereo: Two cameras, with known horizontal displacement and viewing angles, observe the same scene. By identifying the same features in each image, some inferences can be made about depth/3D shape information. (b) Active stereo: Uses a structured light source and one or two cameras (e.g., a steered laser beam) to provide 3D range data.

Stroboscopic (Strobe) illumination. Short-duration, intense pulses of light, often repeated at regular intervals. Used for observing a scene during a short interval of time. Often used to "freeze" movement and eliminate image blur. Common strobe light sources: xenon flash tubes, LED arrays, or a pulsed laser. A typical strobe unit, might use a light pulse lasting 10–100 μs, which is repeated once during each frame period of the inspection camera (25 or 30 pulses/s).

Structural (Syntactic) Pattern Recognition. Analysis of the geometric relationships between features in an image to perform recognition and classification functions. Also see Statistical Pattern Recognition.

Structured light. Pattern of light projected onto a widget, to help identify/measure it. Arrays of points, single lines, sets of parallel lines, grids, individual and concentric circles, intensity "staircase," stepped and continuous "rainbows" are the most common types of projected pattern. By observing the deformation of the projected pattern when viewed from a different direction, the range and slope of an object surface can be measured.

Sub-pixel resolution. When locating certain image features (e.g., straight edges and centres of circular holes), it is common practice to employ techniques for model fitting that allow us to obtain a degree of Precision (q.v.) that is better than one pixel. Industrial systems often achieve a precision of 1/10 pixel size.

Subtractive mixture. Pigments modify light by absorbing a portion of the incident light, depending upon the wavelength. Thus, as light passes through a multi-layer film, its spectrum will be progressively modified. Since each wavelength is modified by a different amount in each layer, the film effectively forms a subtractive mixture of light. (See Additive Mixture.)

Successive contrast. A visual image influences the view of the world immediately afterwards. After-images are created by changes in *successive contrast.*

Surface channel CCD. Type of CCD in which the potential distribution used to confine the charge packets is created by the electrode voltages only. This gives a poor charge-transfer efficiency.

Surface spectral reflectance. (See Spectral Reflectance.)

Synchronisation pulse (more commonly Sync pulse). Pulses accompanying a video signal (part of a composite video signal) which trigger/signal crucial events, such as the start of a line, field or frame scan. See Composite Video.

Synchronous operation of a camera. A camera operating at the mains-power frequency (typically 50 or 60 Hz). This avoids strobing effects caused by the camera and light source operating at slightly different frequencies. See Hum Bars.

Syntactic Pattern Recognition. See Structural Pattern Recognition.

System performance. No single metric is sufficient. Commonly quoted parameters: Accuracy (q.v.) Repeatability (q.v.), Precision (q.v.), error rates (False-Positive (q.v.), False-Negative (q.v.)), throughput rate (for piece-parts: parts-per-minute. For webs: area inspected per minute), Latency. (q.v.).

Systems integration. Art of constructing an harmonious assembly of optical, lighting, mechanical hardware (transport mechanisms, jigs, mounts), electronics (opto-electronics, analogue and digital), software and environmental-protection enclosures, to produce a system that meets the agreed specification and is functionally and ergonomically acceptable to the users (factory workers not managers).

T

T mount. Camera-lens interface. 1″ diameter thread. Often preferred over C-Mount (q.v.) for industrial applications.

Tail-end system. Parts of a vision system concerned with operator interface/display, input/output and communications functions. Form an integral part of a Machine Vision system.

Target system. See Industrial Vision System.

TDI camera Modified version of the Line-Scan Camera (q.v.) that can operate with far lower light levels. Sensor possesses several rows of pixels. As a widget/web moves past the camera, the charge from one row is passed to the next, allowing incoming light to be integrated over several line-scan periods. Requires careful synchronisation between clock and the transport mechanism.

Telecentric lenses. Telescopic lens system with a pin-hole aperture located at the focus of the objective, which must be must be larger than the widget. Accepts only collimated light. Image size does not depend on widget-camera distance. Hence, is useful in metrology.

Telephoto lens. Compact lens for viewing distant scenes. Its overall length is less than its focal length.

Template Matching. Technique for shape matching. Template may be a contiguous blob, or a sparse array of separate pixels. (See N-Tuple.) It is shifted to every possible position over the image. At each position, the goodness of fit is estimated. If the that parameter is within a specified range, the feature represented by that template is assumed to be present. Also see Mathematical Morphology.

Texture. Enigmatic property of a surface, due to its geometry (and lighting angle) or innate colouration. Difficult to characterise in an objective way, since there exist few words to describe it.

Thresholding. Image-processing operation which converts a grey-scale image into a binary image, by setting all pixels above a defined value to *white* and all others to *black*. Despite its simplicity and intuitive appeal, fixed-value thresholding is unreliable and should always be preceded by appropriate filtering and careful control of the illumination level.

Throughput Rate. See Processing Speed.

Tracking. Ability of a lens to remain in focus as it traverses its entire zoom range.

Transition. As we travel along a row/column in a binary image, positions of black-to-white or white-to-black intensity changes (see Run Code).

Translation. Lateral movement of a widget by mechanical means. No change of orientation is implied.

Translucent. Material which transmits some light but that which does emerge is diffused; images are not transmitted clearly. Frosted glass is translucent, whereas window glass is transparent.

Transmittance. Ratio of the radiant power emerging from an optical element to the incident radiant power. Maybe expressed on a logarithmic scale, in decibels.

Transparent. Material which transmits some light, preserving image form. While some wavelengths and polarisations may be heavily attenuated, others preserve image structure. Window glass and plastic sheet are transparent. See Translucent.

Triangulation. Method of determining distance by off-setting a structured Light Source (q.v.), compared to the camera-widget axis. Camera-widget distance (range) can be calculated if the geometry of the optical system is known.

Trichromacy. The human eye effectively has three types of colour sensors, which respond in different ways to various wave-lengths. Provided these are stimulated appropriately, all possible sensations of colour can be produced. To a good approximation, this may be achieved by judiciously mixing red, green and blue lights. Mixing paint (e.g., red, yellow and blue, or cyan, magenta, yellow and black paint/ink) has a roughly similar effect. However, not all colours (e.g., purples and "gold") cannot be reproduced either by mixing light or pigments. The concept of trichromacy in humans is a convenient working hypothesis for Machine Vision, rather than a rigorous truth. While normal human colour vision is trichromatic, some animals (e.g., pigeons) have tetrachromatic vision. (They require the mixing of four primary lights to reproduce a given colour sensation.) Other organisms and some colour-deficient people are dichromatic: they require only two primary lights.

Tri-stimulus colourimetry. Experimental technique for measuring colour by matching a given light source with an Additive Mixture. Any two lights with the same tri-stimulus values appear similar to a person. Once the tri-stimulus values for Spectral Lights (q.v.) are determined individually, it is possible to predict the tri-stimulus values for any mixture of them.

Tri-stimulus values. (See Tri-Stimulus Colourimetry.)

Tritanopia. Yellow-blue colour blindness. Tritanopia results from the loss of function of the S-cones.

Tube-type camera. See CRT.

Tungsten filament lamp. See Incandescent Lamp.

Turn-key (vision) system. See Dedicated (Vision) System.

U

Ultra-sound. High-frequency acoustic waves, beyond human perception. In theory, useful for internal inspection of solid objects but practical problems prevent it being used in Machine Vision. Requires close acoustic coupling (i.e., immersion in a liquid), between the transducer-sensor and the object being examined. Image resolution is poor, as the wave-length is relatively large. Widely used in medicine as a safe alternative to x-rays.

Ultra-violet (UV). Term applied to that part of the electromagnetic spectrum containing radiation in the wavelength range of approximately 100–400 nm, i.e., between visible blue light and X-rays.

Unique hue. Red, green, yellow, and blue are thought to be basic components from which all other hues are composed and are therefore called *unique hues.*

Univariance. A single photo-detector site in the human eye (a single cone in the retina), or a camera (a single photo-transistor), is incapable of distinguishing between an intense light at a wavelength to which it is insensitive and a weaker light at a wavelength to which it is more sensitive. Therefore, chromatic vision, in people, animals and machines requires multiple types of photo-detector.

Unsharp masking. High-pass filtering operation in which a blurred version of an image is subtracted from the original.

UV-A. Ultra-violet radiation in the wave-band $\lambda = 315–380$ nm. Near UV (NUV).

UV-B. Ultra-violet radiation in the wave-band $\lambda = 280–315$ nm. Mid UV.

UV-C. Ultra-violet radiation in the wave-band $\lambda = 100–280$ nm. Deep/far UV (DUV or FUV).

V

Validation. Rigorous testing of a vision system. Needed to verify that it performs as claimed.

Variable-scan frame grabber. Device capable of accepting a variety of non-standard video input formats. Also see Frame Grabber.

Vari-focal. Low-cost zoom lens. Unable to track accurately. See Tracking.

Video. Signal generated by a television camera, computer, or other device in the same format. Various standards exist, most notably CCIR (called RS-330 in North America) and RS-170. See Composite Video.

Vidicon. Type of CRT (q.v.). A range of broadly similar photo-conductive camera tubes, using a transparent signal plate and a low-velocity scanning electron beam. Noted for its versatility and good performance for a modest cost.

Vignetting. Dark shadows around the corners of a video image. Created by using an optical system that produces an image that is too small to cover the camera's light-sensitive surface.

VIS. See Visible Light.

Visible light (VIS). Region of the electromagnetic spectrum in which the human retina is sensitive: wave-lengths of approximately 0.4–0.7 μm.

Vision engine. Plug-in accelerator, designed specifically for image processing and analysis functions. To achieve maximum speed, some flexibility has to be sacrificed.

Visual cortex. Those cortical areas primarily concerned with the processing of visual information. The visual areas of the cortex are primarily located in the occipital lobe: the rear, lower part of the cortex.

Vision system. (Synonym for Machine Vision system*)* An engineering system, usually containing equipment for most or all of the following functions: mechanical translation (of the Widget (q.v.), optics or camera), lighting, image formation (optics), image sensing (video camera or scanner), fast analogue and/or digital video-signal processing, image analysis (implemented in software) and user interaction (for programming, calibration, display of results and testing).

W

Wave-length. Parameter relating to visible light, which identifies its colour. Term is also applicable to characterising a much wider range of electro-magnetic radiation, including (in order of decreasing wave-length): radio waves, microwaves, IR, visible light (VIS), UV, x-rays and gamma rays. Usually denoted by λ. Related to frequency (f) and velocity of propagation (c) by the relationship c = f.λ. Also see Spectrum.

Weber's law. Natural vision systems often respond so that, for a given observed parameter, S, the following ratio is held constant:

$$[\textit{minimum perceptible change in S}]/[\textit{absolute value of S}].$$

This accounts for the fact that the smallest detectable intensity difference between two lights is larger if they are both bright.

Weight matrix. Matrix of constant coefficients defining the multiplicative constants employed by a local or N-Tuple Operator. (q.v.).

Well. Morphological operator equivalent to Closing (q.v.) followed by subtraction of the input image.

Widget. Object to be inspected. Has a specific meaning in the context of this book.

Windowing. Performing image processing operations only within a predefined area (window) of an image. Also see ROI.

Woods glass. Type of glass that transmits UV but is relatively opaque to VIS.

X

Xenon lamp. High intensity arc discharge bulb. Light from has a similar spectrum to that of daylight. Suitable for strobe lights. Dangers: high UV content, high flash rates can induce photo-sensitive epilepsy.

X-ray. A portion of the electromagnetic spectrum beyond the ultraviolet with higher frequency and shorter wavelengths. Able to penetrate solid objects for internal, non-destructive evaluation.

Y

YAG laser. Yttrium-aluminium-garnet laser, capable of providing continuous IR radiation with power output of several watts. Dangerous!.

Z

Zone plate. Transparent plate with a pattern of concentric opaque rings, designed to block off every other Fresnel half-period zone. Light from a point source passing through the plate produces a point image much like that produced by a lens.

Zoom lens. Compound lens which remains in focus as the image size is varied. May be motorised or manually operated. See Vari-Focal Lens.

Zoom ratio. Ratio of the longest and shortest focal lengths of a zoom lens. May be as high as 40:1 for a broadcast-quality lens.

Appendix B: MV Questionnaire

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_43,

B.1 Why a Questionnaire is Needed?

The *Machine Vision Questionnaire* (*MVQ*) is intended to assist in formalising the requirements and specification for a vision system that will perform a new inspection function, or one that has hitherto been performed by other means. The questionnaire raises a wide variety of issues that should be considered by a vision engineer, so that he/she can quickly decide whether Machine Vision is likely to satisfy the application requirements. Past experience has shown that, with careful initial analysis of the application requirements, a great deal of time and money can be saved; asking the right questions at an early stage can help to avoid frustration. The MVQ contains questions that help to establish technical feasibility and likely benefits of Machine Vision, as early as possible. The answers might also provide a basis for writing a detailed specification and contract for the design, construction and delivery of a vision system.

The MVQ outlined in this appendix concentrates on technical issues. In addition, a large number of managerial/contract details should be agreed between the equipment supplier and customer before work begins in earnest. These include maintenance, servicing, warranty, provision of spares, documentation, ownership of intellectual property, possible future upgrading of the system, safety, legal responsibility, ISO 9000/14000 certification, etc.

B.2 History

The *Machine Vision Application Requirements Check List* was originally devised by Nello Zuech, Vision Systems International, Yardley, PA 19067, USA. It was included in his book "*Understanding and Applying Machine Vision*," CRC Press, Boca Raton, FL 33487, USA, ISBN-13 9780824789299, 1999. Nello Zuech's version of the *Machine Vision Application Requirements Check List* was published without restrictions on use.

An updated version of the *Machine Vision Application Requirements Check List* was subsequently made available and published via a web site created by Image Labs International, Inc., Bozeman, MT 59715, USA. URL http//www.vision1.com/vsi/mvaques.html, accessed 9th October 2008. The Image Lab version was published without restrictions on use.

The *MVQ* printed in this appendix was prepared by Bruce Batchelor and derives inspiration from the *Machine Vision Application Requirements Check List*. It is also published free of copyright; permission is granted to use the *Machine Vision Questionnaire* freely without restriction.

B.3 Machine Vision Questionnaire

Note: *Widget* is used to denote the *object to be inspected.*

B.3.1 Current Production and Inspection Processes

1. Name/and part number of the widget.
2. Latency time between image acquisition and the final action (e.g., accept/reject widget)
3. What is the expected life of the widget?

4. What is the maximum acceptable pay-back period for capital expenditure on QA equipment?
5. How is inspection currently performed?
6. How the current inspection process is deficient.
7. Goal(s) for the proposed automated visual inspection/visual control system
    - Avoid accidental damage to machines due to jamming on faulty products
    - Improve process efficiency
    - Improve process efficiency
    - Improve process safety
    - Improve product reliability
    - Improve product safety
    - Match production speed
    - Reduce waste of materials
8. What is the current reject rate for bad widgets, according to your established factory-floor inspection methods?
9. What is your best estimate for the real rate of occurrence of faulty widgets. (You could, perhaps, cite careful non-real time, laboratory analysis?)
10. Why do the answers to the two previous questions differ so much?
    - What is the false reject rate?
    - What is the false accept rate?
        - Is there any realistic chance of reducing these by improving the current inspection method?
        - Which is more problematical, *false reject* or *false accept*?
11. Production mode
    - Batch production
    - Continuous process
12. Change-over of manufacturing system to alternative product
    - Time required for change-over
    - Frequency of change-over
    - Skill levels involved needed for the change-over
13. Where widgets go after inspection
    - Good widgets (“widget heaven”)
    - Faulty widgets (“widget hell”)
14. Are widget design or production process changes anticipated?
    - How do you envisage the present product range to change in the foreseeable future?
    - Anticipating future products, are there an any additional inspection functions that you would like the proposed Machine Vision system to perform?
15. Is every widget inspected?
    - Sampled inspection (specify the sampling rate)
        - 100% inspection
16. How inspection is performed
    - Manually, as part of manufacturing process
    - Manually, unaided, dedicated inspector
    - Manually, with standard charts, pictures, swatches, etc.
    - Manual application of instruments
    - Automatically (state how)

17. When a faulty widget escapes detection, what are the effects down-stream on the following?
    - Quality
    - Cost
    - Product safety
    - Process safety
    - Repair/rework
    - Loss of production
    - Serious damage to machine(s) down-stream
    - Jamming of machine(s) down-stream
    - Other
18. If inspection were effective, could any down-stream testing requirements be relaxed?
19. Cost (preferably expressed in quantitative terms)
    - Undetected faulty widget
    - Good widget that is falsely rejected
20. Previous experience/attitudes of staff towards Machine Vision/automation
    - Factory-floor personnel
    - Factory manager
    - Project champion

B.3.2 Describing the Widget

This section should be completed for each type of widget.

21. What is the expected life of the widget?
22. Product volume (total number of widgets/day, all lines making this product)
    - Is this expected to rise in the foreseeable future?
23. Production rate for each line
    - Average production rate (widgets/min)
    - Peak production rate (widgets/min)
    - Number of widgets arriving simultaneously at the inspection point
    - Is this expected to rise in the foreseeable future?
24. Manufacturing process(es) used when making the widget
    - Moulded
    - Cast
    - Forged
    - Stamped
    - Pressed
    - Rolled
    - Pressed
    - Turned
    - Extruded
    - Spun
    - Woven
    - Sintered
    - Cut from stock

- Folded from sheet
- Multi-stage machining
- Other, specify

25. What is the last machining operation, just before the point of inspection?
26. No. of materials making up the widget
    - Single monolithic widget
    - Single multi-material widget
    - Assembly of monolithic widgets
    - Assembly of multi-material widgets
27. Materials used in manufacture (e.g., steel, brass, PTFE, glass, wood, etc.)
28. Variation of appearance within a given batch
    - Fixed
    - Some variability within a batch
    - Very variable
29. Variation of appearance between batches
    - Fixed
    - Some variability
    - Very variable
30. Changes in appearance over time
    - Months
    - Days
    - Hours
    - Minutes
    - Unchanging
31. Coating (at least as important as the substrate material in determining appearance)
    - None
    - Painted
    - Lacquered
    - Chemical coating
    - Powder coating
    - Plated
    - Other, specify
32. Colouring
    - Monochrome
    - Dichromic
    - Trichromic
    - Polychromic (multiple discrete colours)
    - Subtle colouring
33. Dimensions
    - Absolute maximum, widget is allowed to be in any orientation
    - Maximum when the widget is held so that important features are visible
    - Orthogonal axis (90° to maximum visible axis)
    - Along visual axis (include only visible features)
34. Fragility
    - Tough as old boots
    - Medium (comparable to glass bottle)
    - Fragile (comparable to egg shell)

35. Hazards
    - Radiation, specify type and level
    - Biohazard
    - Toxic
    - Sharp
    - Electrical
    - Physical (e.g., liable to explode/implode, presents finger trap, etc.)
    - Heavy
    - Insignificant
36. Internal detail
    - Not relevant, only external features are important for inspection
    - Important internal features can be observed
    - Optically
    - UV
    - IR
    - X-rays
37. Light penetration
    - Opaque
    - Transparent
    - Transluscent
38. Overlay
    - None
    - Printed
    - Photography
    - Lithography
    - Other, specify
39. Patterned surface
    - Regular
    - Irregular
    - Spotty
    - Striped
    - Chequered
    - Blotchy
    - None
40. Photo-active
    - Light sensitive
    - Fluorescent
    - Phosphorescent
    - Photo-chromic
    - None
41. Physical form
    - Spider-like
    - Deep concavities
    - Shallow concavities
    - Convex

42. Stability of posture
    - Neutral
    - Effectively single stable state, guaranteed by mechanical handling system
    - Single stable state, inherent in shape and mass distribution
    - Several stable states
    - No stable state
43. Rigidity
    - Rigid
    - Flexible
    - Articulated
    - Serpentine
    - Chain-like
44. Shape, as seen by inspector
    - Spider-like
    - Linear (e.g., screwdriver)
    - Curved (e.g., banana)
    - Serpentine
    - Coiled
    - Twisted
    - Knotted
    - Compact
    - Round
    - Flat
    - Long and thin
    - Cube
    - Rectangular
    - Other (specify)
45. Surface appearance
    - Mirror-like
    - Bright shiny
    - Dull-shiny (e.g., smooth rubber)
    - Matt
    - Egg-shell
    - Dull (e.g., soot)
46. Surface condition
    - Clean
    - Wet
    - Oily
    - Rusty
    - Oxidised
    - Stained
    - Dusty
    - Biological activity
47. Surface roughness
    - Mirror finish
    - Smooth

   - Rough
   - Very rough
   - Textured – regular
   - Texture – irregular

48. Dimensions
    - Absolute maximum, widget can be in any orientation
    - Maximum when the widget is held so that important features are visible
    - Orthogonal axis (90° to maximum visible axis)
    - Along visual axis (include only visible features)
49. Size, variability
    - Highly variable
    - Some variation
    - Nearly constant
    - Highly consistent
50. Temperature
    - [<20°]
    - [−20°–0°]
    - [0°–50°]
    - [50°–100°]
    - [>100°]
51. Unit value
    - Very low value (≤ 1¢)
    - Low (¢s)
    - Medium (€/$)
    - High value (>10 €/$)
52. Weight
    - [<1 g]
    - [1 g–10 Kg]
    - [>10 Kg]

B.3.3 Faults in the Product

53. Biological damage
    - Micro-organisms
    - Rotting
    - Bird peck
    - Bite
    - Scratch
54. Colour
    - Missing imprint
    - Misplaced colour imprint
    - Low saturation
    - High saturation
    - Wrong hue
    - Faded

55. Contamination, foreign bodies/material
    - Fingerprint
    - Oil
    - Water
    - Rust
    - Fungus, other micro-organisms
    - Dirt
    - Dust
    - Staining
    - Ink
    - Paint
    - Acid
    - Animal debris
        - Bones
        - Insects
        - Rodents
        - Excrement
        - Urine
        - Hair
        - Nesting materials
    - Stones
    - Soil
    - Sand
    - Raw materials used in manufacture
    - Coolant
    - Lubricant
    - Other, specify
56. Dimensions
    - Too big
    - Too small
    - Too high
    - Too short
    - Spacing between features
    - Angle position of features
    - Radial position
57. Grading
    - Inferior grade
    - Superior grade
    - Wrong type
58. Incorrect assembly
    - Part missing
    - Part damaged
    - Part incorrectly fitted
    - Wrong part fitted
59. Light damage
    - Faded

60. Local defect
    - Cut
    - Tear
    - Crack
    - Split
    - Chip
    - Scratch
    - Pit
    - Blow hole
    - Debris
    - Sprue
61. Machining
    - Incomplete
    - Machining step missing
    - Machining error
62. Malformed
    - Bent
    - Broken
    - Twisted
    - Extra material
    - Mould seam marks
    - Sprue still attached
    - Machining fault
63. Misprinting
    - Colour imprint misregistration
    - Smudged
    - Ink spill
    - Oil
64. Number
    - Too few
    - Too many
    - Unable to count
65. Pose
    - Wrong orientation
    - Wrong position
    - Wrong posture
    - Inverted (up-side down)
66. Weave error
    - Weave
    - Broken thread
    - Pulled thread

B.3.4 Vision Task

67. How is inspection currently performed?

68. Explain how it is deficient in meeting the required performance? Why do you need to inspect the widgets or control their production process?
    - Improve process efficiency
    - Improve process safety
    - Improve product safety
    - Improve product reliability
    - Match production speed
    - What is the current reject rate of bad widgets?
69. What is the accuracy of the current inspection system?
    - False reject rate
    - False accept rate
70. What benefits do you hope to achieve by installing a Machine Vision system?
71. Function(s) to be performed by the vision system
    - Detecting local faults
    - Detecting malformation
    - Grading
    - Sorting
    - Checking feature existence
    - Measurement
        - Linear dimensions
        - Angles
        - Position (2 dimensions)
        - Position (3 dimensions)
        - 3D shape
        - Orientation
        - Posture
    - Checking colour
    - Checking texture
    - Other, specify
72. Must every widget be inspected?
    - Sampled inspection is adequate (Specify the minimum sampling rate)
    - 100% inspection is desirable
    - 100% inspection is essential
73. Are widget design or production process changes anticipated?
    - How do you envisage the present product range to change in the foreseeable future?
    - Anticipating future products, are there an any additional inspection functions that you would like the proposed Machine Vision system to perform?
74. Operating mode of the proposed vision system
    - Will the vision system be retro-fitted to a new line or an existing line?
    - Number of widgets to be examined at the same time
    - Number of different types of widget to be accommodated
    - How many lines/machines will vision system(s) be needed for?
    - Is inspection to be performed on-line or off-line?
    - Throughput rate (widgets per minute)
75. Check cleanliness
    - Floor
    - Material deposits around feed chute or hopper

- Stairs
- Tools
- Work space
- Check feed-stock

76. Check transport system
    - Discrete widgets, jumbled in a barrel/hopper
    - Discrete widgets, loaded onto specialised carrier
    - Granular material
    - Linear object (strip/bar/pipe/tube/wire/cable/extrusion)
    - Liquid level
    - Motion
    - Indexing
    - Cleanliness
77. Checking safety
    - Intruder surveillance
    - Operator hands
    - Stairs
    - Gangways
    - Flames
    - Sprays
    - Doors
    - Debris
    - Widgets transport
    - Materials transport
    - Chutes
    - “Waterfalls” of granular material
78. Classify
    - Identify
    - Identify piece widgets
    - Grading natural product species
    - Classifying natural product varieties or species
79. Machine/Robot Control
    - Robot positioning has 1/2/$2^1/_2$/3/$3^1/_2$/4/5/6 degrees of freedom
    - Function
        - Assembly
        - Cutting
        - Drilling
        - Grinding
        - Marking
        - Milling
        - Packing
        - Painting
        - Palletising
        - Polishing
        - Sanding
        - Soldering

    - Sorting
    - Spraying
    - Stacking
    - Turning
    - Washing
    - Welding
    - Other, specify

80. Counting
    - Objects
    - Features on large objects
81. Detecting
    - Broken tool
    - Damaged widget
    - Error in materials feed
    - Incomplete widget
    - Malformed widget
    - Worn tool
82. Finding
    - Objects
    - Features on a complex object
83. Locating
    - Feature
    - Grasping point
    - Object
84. Measuring
    - Angle
    - Linear dimension
    - Area
    - Distance between features
    - Angles defined by widget features
85. Inspecting printing
    - Accuracy of colour rendering (hue and saturation)
    - Character recognition (printed text is not known beforehand)
    - Character verification (printed text is known beforehand)
    - Colour imprint alignment
    - Damaged paper
        - Creased
        - Torn
        - Stained
        - Dirty
    - Deficiency of ink
    - Print quality
        - Character formation
        - Sharpness
    - Smudging
    - Surplus ink

86. Recognise
    - Industrial widgets
    - Natural product, varieties
87. Web fault detection and classification

B.3.5 Viewing Background

88. Access
    - Possible
    - Limited
    - Impossible
89. Colour
    - Monochrome
    - Dichromic
    - Trichromic
    - Polychromic
    - Subtle colouring
    - Can be selected to suit vision system
90. Constancy
    - Constant
    - Constant with minor local changes
    - Major changes
91. Light level
    - Uncontrolled
    - Low
    - Medium
    - Bright
92. Reflectivity
    - High
    - Medium
    - Low
93. Structure
    - Plain
    - Structured
    - Patterned
    - Textured
    - Factory floor
    - Equipment

B.3.6 Environment

94. Access, camera
    - Easy
    - Limited
    - Very difficult

95. Access, lamps
    - Easy
    - Limited
    - Very difficult
96. Acoustic noise
    - None-to-moderate
    - Severe
97. Ambient light level
    - Total darkness
    - Very dark (e.g., moon-light)
    - Dark (e.g., in shadow in living room at night)
    - Bright (room lit by electric lamp)
    - Very bright (close to electric lamp)
    - Sunlight
98. Atmospheric pollution
    - Very little
    - Dust
    - Fumes
    - Smoke
    - Steam
    - Chemical mist (aerosol droplets)
99. Heat from lamps
    - Not important
    - Possible damage to product
    - Possible difficulties with process
    - Critical
100. Shrouding, screen around widget, camera and lights
    - Possible
    - Difficult
    - Impossible
101. Vibration
    - Little
    - Moderate
    - Severe
102. General working environment
    - General factory
    - Laboratory
    - Clean room
    - Outdoor
103. Air quality
    - Dust
    - Smoke
    - Flying debris
    - Water droplets
    - Water mist

- Steam
- Oil mist
- Oil droplets

104. Wash-down requirements
    - Water mist
    - Water jet
    - Steam
    - Chemicals, specify
105. Ambient light
    - Incandescent
    - Fluoresecent
    - Mercury lamp
    - Sodium lamp
    - Other artificial
    - Daylight (no direct sunlight)
    - Direct sunlight
106. Light pollution from nearby equipment
    - Little
    - Constant lights
    - Flashing lights
107. Temperature range (°C)
108. Humidity
    - Relative humidity (%)
    - Dew point (°C)
109. Radiation
    - EMI
    - IR
    - Nuclear
    - RFI
    - UV
    - X-rays
110. Vacuum, yes/no
111. Compressed air
    - Available/not available
        - Content
    - Water
    - Dust
    - Oil
    - Other
112. Power
    - AC
    - Voltage
    - Frequency
    - Single-phase
    - 3-phase

- Regulated
- Action needed in event of general power failure
- DC
- Voltage (V)
- Voltage tolerance (±V)
- Current limit (A)

B.3.7 Mechanical Handling

113. Widget delivery
    - What is the last machining operation, just before the point of inspection?
    - How will the widgets be transferred from there to the vision system?
    - Rate of delivery (widgets/min)
114. Is this production rate expected to rise in the foreseeable future?
115. Where widgets go after inspection?
    - Good widgets
    - Bad widgets (bad little widgets go to the widget hell!)
116. Repair/reworked of faulty widgets
    - Is this technically possible?
    - Is this economically viable?
    - Can Machine Vision assist in the diagnosis?
117. Type of handling system
    - Linear conveyor
    - Rotary conveyor
    - Robot
    - Chute
    - Free fall
    - Manual
118. Possible to damage widget during handling?
119. Special handling precautions (e.g., keep widget moist/cool)
120. Inspection station expected to perform other functions? (specify)
121. Production rate (widgets/min)
    - Average throughput rate
    - Peak throughput rate
    - Is it possible to smooth the flow through inspection system by delaying widgets?
    - Latency time between image acquisition and the final action (e.g., *accept/reject*)
122. Indexed motion
    - Stationary time
    - Total in-dwell-out time
    - Settling time
    - Precision of placement
        - ± δX (orthogonal to line of motion)
        - ± δY (along line of motion)
        - ± δZ (vertical axis)

    - ± δ angle around X axis (degrees)
    - ± δ angle around Y axis (degrees)
    - ± δ angle around Z axis (degrees)

123. Widgets moving continuously
    - Motion
        - Linear
        - Circular
    - Direction
    - Nominal speed
        - Variation in speed (±percent)
        - Speed, regulation possible?
    - Shudder during motion
    - Maximum variation of widget position across belt
    - Spacing between widgets along the belt
        - Random
        - Nominally fixed
    - Repeatability of spacing between widgets
    - Can the orientation be controlled (e.g., by guide rails)?
124. Widgets touching one another?
125. Widgets overlapping one another?
126. Space available for inspection station
    - Open
    - Restricted
127. Minimum distance from camera to computer
128. Distance from inspection system to interfaced equipment
129. Other physical constraints around installation site (specify)
130. Conveyor
    - Type
    - Structure (e.g., belt, chain, etc.)
    - Colour
    - Appearance (e.g., bright metal)
    - Accumulation of dirt/debris?
    - Other factors affecting appearance
    - By-pass mode required?
131. Is more than one camera likely needed to see all of the significant faults?
132. Will the vision system be fitted to a new line or an existing line (retrofit)?
133. Number of widgets to be examined at the same time
134. How many manufacturing lines must be fitted with a vision system?
135. Is inspection to be performed *on-line* or *off-line*?

B.3.8 Interfacing

136. Operator interface
    - Essential features
    - Desirable features

137. User
    - Educational level/experience
        - Analogue electronics
        - Computers
        - Digital electronics
        - Digital photography
        - Illumination engineering
        - Manufacturing equipment
        - Optics
        - Video
138. Restricting personal access
    - Password protection
    - Face recognition
    - Voice recognition
    - Speaker recognition
    - Finger-print recognition
139. Display of results
    - Summary statistics (specify reporting period)
    - Tabulated data
    - Graphs
    - “Live” image from camera
    - Images recorded from recent faulty parts (how many?)
    - Images recorded from recent good parts (how many?)
        - Marking widget (e.g., spraying faulty widgets red)
        - Must reports be generated without interrupting inspection
    - Hard copy required
    - Electronic copy (format: CD, DVD, memory stick or remote computer)
140. Additional alarm
    - Audible
    - Flashing light
    - Alert remote computer
141. System failure
    - Action required
    - Power-failure action
142. Machines to be interfaced to the vision system
    - Mechanically
    - Electrically
143. Event that will trigger inspection
    - Proximity detector
    - PLC
    - Computer
    - Manufacturing machine (e.g., ejection of widget from moulding machine)
    - Other (specify)
144. Machine–machine interface
    - Interface standard (e.g., RS232, RS422, RS449, IEEE488, parallel, PLC, Ethernet, USB, IEEE1394)
    - Opto-isolation

B.3.9 Other Issues

145. Accessibility. The following are helpful in the design process. Explain any problems.
    - Collecting a large number of representative samples of good and bad widgets
    - Engineering drawings of the widgets
    - Protocols and *accept/reject* rules currently given to human inspectors
    - Diagrams/pictures/swatches currently given to human inspectors
    - Still photographs and/or video recording of the production process
    - Virtual tour of the factory
    - Direct observation of the production process in the factory
146. Calibration
    - Procedure
147. Equipment required (models, charts, swatches, etc.)
148. Computer network, integration of vision system into company intranet
149. Cost
    - Capital
    - Operating costs
150. Documentation
151. Duty cycle (number of hours use per week)
152. Fixtures and cabling
153. Hardware, reliability of supply of spares from 3rd party
154. Intellectual property, ownership
155. Legal issues, specific to customer's type of business
156. License fees for installed software
157. Maintenance and cleaning
    - Procedures
    - Materials
    - Frequency
    - Duration
158. Operator safety
159. Paint colours
160. Post-sales support
    - Help line
161. Power consumption
162. Response time for service
163. Responsibility
    - Backup
    - Installation
    - Software changes
    - Revisions
164. Safety of vision engineer
    - Protocol when vision engineer is visiting factory
    - Vision engineer handling dangerous widgets
    - Additional safety issues (e.g., standard enclosure required)
165. Shipping
166. Spare parts, stock requirements

167. Tests, installation site
    - Supply of good, faulty, and/or marginal widgets
    - Variations of parts (e.g., size/colour variations)
    - Sample size
    - Acceptability criteria
    - Widget position variations
    - Widget orientation variations
    - Lighting variations
    - Environmental conditions
168. Tests, vendor site
    - Supply of good, faulty, and/or marginal widgets
    - Variations of parts (e.g., size/colour variations)
    - Sample size
    - Acceptability criteria
    - Widget position variations
    - Widget orientation variations
    - Lighting variations
    - Environmental conditions
169. Training
    - Factory-floor personnel
    - Engineering staff
170. Verification procedure needed to system is performing correctly
171. Warranty
    - Duration
    - Terms and conditions

Appendix C: Proverbs

Bruce G. Batchelor[1] · *Paul F. Whelan*[2]
[1]Cardiff University, Cardiff, Wales, UK
[2]Dublin City University, Glasnevin, Dublin, Ireland

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_44,

The following observations, comments and suggestions are based upon the experience of the author and his colleagues. While this list is presented in a light-hearted manner, it encapsulates some important lessons that are unfortunately not universally appreciated or applied when designing or choosing a new vision system.

C.1 General

There is more to machine vision than meets the eye.

- ► A machine vision system does not see things as the human eye does.

An eye is not a camera. A brain is not a computer.

- ► Machine vision systems should not necessarily be modelled on, or intended to emulate human vision.

Machine vision is not a scientific discipline.

- ► Machine vision is not an exercise in philosophy but an engineering project.

No vision system should be required to answer the general question "What is this?"

- ► It is better for vision systems to answer more specific questions, such as "Is this widget well made?" Verification (i.e., checking that the widget is well made) is better than recognition, where few or no a priori assumptions are made.

Intelligence ≠ Computing power.

- ► Making the computer more powerful does not necessarily make the system smarter.

Optimal solutions do not always exist.

- ► If they do exist, optimal solutions may be too complex, or impossible to find. We should therefore be prepared to search for and accept satisfactory solutions, rather than optimal ones.

Use a standard solution to a vision problem but only if it is sensible to do so.

- ► Wherever possible we should provide standard solutions to industrial problems, since this helps to broaden the application base.

Avoid the application of machine vision techniques for their own sake.

- ► It is vanity on the part of the vision engineer to do so. There are plenty of other methods of solution available. Most of them are cheaper than vision.

Defect prevention is better than cure.

- ► We should consider using vision in closed loop feedback control of the manufacturing process.

Do not rely on second-hand information about the manufacturing process and environment.

- ► The vision engineer should always see the manufacturing process for himself. If the customer is unwilling to let the vision engineer into the factory, it may be necessary to abandon the application.

Vision systems need not be fully automatic.

- ► While it is more usual to use a fully automatic vision system, it can be used instead to enhance images for subsequent human analysis.

C.2 Systems

No system should be more complicated than it need be.

- ► This is a reformulation of Occam's Razor, which in its original form is "Entia non multiplicanda sunt." In its English translation, excessive complication is attributed to mere vanity. In colloquial use, this is often referred to as the KISS principle. (Keep it simple, stupid.) Simple systems are almost always the best in practice.

All parts of a properly designed machine vision system bear an equal strain.

- ► Of course, it is impossible to measure strain in any formal sense. The point is that no part of a vision system should be made more complicated because a sloppy attitude has been adopted during the design of other parts. A particularly common error is the tendency to concentrate on the image processing, to the detriment of the image acquisition (i.e., pose of the object being inspected, lighting, optics and sensor).

If it matters that we use the Sobel edge detector rather than the Roberts operator, then there is something fundamentally wrong, probably the lighting.

- ► This remark is not about the relative merits of the various edge detection operators but is a statement about the need for a broader "systems" approach. A common error is to pay much more attention to the image processing process but ignore the fact that the image contrast is low because the lighting sub-system is poorly designed.

The following inequality is always true:
$\text{Vision-system} \neq \text{PC} + \text{Framegrabber} + \text{Camera} + \text{Software}$.

- ► To many people, these are the only components needed to build a vision system. However, this neglects many important issues: lighting, optics, systems integration, mechanical handling, ergonomics and standard industrial inspection practice.

Problem constraints allow the vision engineer to simplify the design.

- ► By taking systems issues into account, it may well be possible to design a simpler, faster, cheaper and more robust system.

Vision systems can use the same aids as people to reduce task complexity.

- ► For example, special optical/lighting techniques, X-rays, fluoroscopy, multi-spectral imaging, specialised sample preparation can all be used.

Documentation is an essential part of the system.

- ► A vision system will not survive for long without sufficient documentation.

C.3 Customer

Whatever software and hardware that a machine vision system uses, the customer will want it to be different, so don't tell them.

- ► Many customer companies have a policy of using certain types of computer hardware/software, which will often conflict with the vision system. It is wise to regard the vision system as a closed box.

The customer must not be allowed to tinker with the system after it is installed.

- ► The customer should be dissuaded from making internal adjustments to the system, since this requires rare and specialised skills (lighting, optics, camera, algorithms, systems integration).

The customer's company just does not make defective widgets; the vision system is simple intended "to improve product quality."

- ► Companies are often sensitive about the way that quality (or lack of it) in their products is discussed. This must be borne in mind when designing a vision system and particularly when reporting developments at conferences, in publications, etc.

Everybody (including the customer) thinks that he/she is an expert on vision and will tell the vision engineer how to design the machine.

- ► This is, regrettably, one of the great truths. As a result, everybody will feel it is their right and duty to tell the vision engineer how to do his job. In many instances, prototyping tools need to be used for the specific purpose of convincing the customer that his intuitive approach just does not work reliably.

The widgets that were provided for the feasibility study were specially cleaned and chosen by the customer for the project.

- ► Beware of the pernicious habit of some customers who deliberately, or through ignorance, select good quality products to show the vision company, rather than providing a more representative sample.

Customer education is an integral part of vision system design.

- ► A well educated customer can help to reduce the project cost and may well help to reach a better system design.

A little knowledge is a dangerous thing.

- ► The customer will suggest many changes to the system design if he is ignorant of the subtleties which led to the present design. It is best to tell the customer all or nothing. For example, the vision engineer should not tell the customer that the system uses a camera costing $5,000, because the latter will know of a camera that costs only $100 but will not appreciate the benefits of the more expensive device.

C.4 Financial

The vision system must pay for itself in 6 months.

- ► The vision engineer must be prepared to argue against the simple-minded attitude which attempts to judge the value of a vision system solely on financial grounds. When a company buys a vision system, it is investing in the improvement of the quality/safety of its products.

Component cost is not the same thing as system cost.

- By purchasing one relatively expensive component, it may be possible make the overall system cheaper, faster, and more reliable.

Only 10% of the cost of installing a vision system is directly attributable to the requirements of image formation, acquisition and processing.

- The remaining 90% of the project cost is due to making the system work properly in the factory.

$1 spent on inspection is worth $10 in improved profits.

- Investing a little in automated visual inspection can lead to significant gains in improved efficiency.

C.5 System Specification

The specification of the vision system is not what the customer wants.

- Do not try to take short cuts in the initial dialogue. The vision engineer should be prepared to spend a considerable amount of time finding what the customer really wants.

The system specification must be agreed and fully understood by all concerned.

- All specifications should be in writing with negotiable and non-negotiable specifications noted before the design proper begins.

No machine vision system can solve the problem that the customer forgot to mention when placing the order.

- Undertake a proper and complete design study for each type of product.

The specification of a vision system should indicate its functionality and performance.

- It should not be used merely as a marketing tool.

Beware when the customer says “By the way! We would like to be able to inspect these objects as well.”

- We repeat the point just made above: undertake a proper design study for each type of product.

Simple accept/reject labelling is easier than classifying defects by type.

- If the customer wants to classify defects, they should be made aware that this could have a major bearing on the cost of the inspection system. Detailed classification of defects can greatly increase the speed/cost of the vision system.

It may not be possible to classify defects reliably.

- The classification process may not always be clear-cut. A certain product may, for example, have a combination of faults. The vision system supplier and customer must agree beforehand what bounds are to be imposed on the classification process.

Specify the operating environment.

- It is relatively easy to make a system that works well in the laboratory. However, it is much more difficult to build a target system that will work reliably in a hostile factory environment.

Defect types must be realistically prioritised.

- ► The ranking of defect types in order of importance can have a major influence on the approach taken, and hence the final cost of the solution. For example, it may be the case that 90% of defect types can be detected for a cost of 90% of the total project budget, whereas detecting the remaining 10% of defect types would cost another 90%. (This is an example of the 90:90 rule.)

C.6 Choosing Inspection System Design Samples

Maximise the number of product samples.

- ► The feasibility study, the target system design process, the testing and evaluation of the target system and any demonstrations to the customer should all be based on a large number of representative sample parts. These samples should cover the full range of part variability.

Choose design samples following proper statistical sampling techniques.

- ► Their selection should be made according to a carefully planned and agreed protocol.

If necessary, choose inspection samples manually using agreed criteria.

- ► If samples are chosen manually they will need to be cross-checked to ensure that the variation found in manual inspection is minimised. It is critical that the vision engineer establishes a reliable training set.

The customer said his widgets were made of brass. He did not think to state that they are always painted blue and oily.

- ► To the vision engineer, the surface finish is more important than the underlying material. This contrasts sharply with the customer who often regards surface finish as being of mere cosmetic value.

Classify sample defects.

- ► There are many different ways in which a product can fail to meet its criteria. Any specific application knowledge that the customer can add concerning the type and origin of the fault, will be useful in the design process.

C.7 Vision Company

A sales-person who says that their company's vision system can operate in uncontrolled lighting is lying.

- ► No. We are not exaggerating. The human eye cannot. No machine can either.

A happy vision team has (at least) seven players.

- ► This consists of engineers who specialise in mechanical handling, lighting, optics, video sensor technology, electronic hardware, software, vision system integration.

C.8 Alternative Solutions

What a person cannot see, generally cannot be detected by the machine vision system.

- The human eye is remarkably adept and versatile. In contrast, a vision system is clumsy and unsophisticated, although it may be faster and more reliable. It is a good maxim to admit defeat sometimes as this will gain customer confidence, in the long term.

It may be cheaper to hire a person to inspect the widgets.

- However, a machine may be faster, more consistent and reliable. Be prepared to argue this point with the customer.

Machines can do some things better than humans.

- Machines can sense outside the visible spectrum (X-rays, IR, UV). Line-scan cameras and laser scanners can produce high resolution images that cannot be seen directly by the eye. Depending on the technology used, a machine vision system would be expected to achieve a substantially higher inspection efficiency, and it can theoretically do this for 24 hours a day, 7 days a week. Machine vision can also be useful at detecting gradual changes in continuous processes that appear over long time periods. For example, inspecting gradual colour variations in the production of web materials. Such a gradual change in colour is unlikely to be detected by a human operator.

People can do some things better than machines.

- So far, no machine has been built that can reliably guide a car through busy traffic, safely and at speed. No machine can yet judge the aesthetic qualities of a person's dress or a fine painting.

Even the best human inspector is only 70% efficient.

- This is one of the best arguments in favour of using machine vision. A person is easily distracted, for example by a good-looking member of the opposite sex walking past. The performance of a human inspector falls as a results of boredom, dissatisfaction with employment, distress due to a recent argument, illness, fatigue, hunger, discomfort, pain, alcohol and drug ingestion.

Machines can work in situations that people cannot tolerate.

- Machines can work in radioactive, chemical and biological hazards, where there are high levels of noise, IR, UV, X-ray and microwave radiation, or it is very hot. Machines can tolerate flashing lights, which would induce epileptic fits and migraine attacks in people. A camera can operate under very high, very low, or suddenly changing pressure, and can also be used safely where there is a danger of explosion, or brittle materials are likely to shatter suddenly. A camera can be placed close to a laser cutter, which would be dangerous to a human being. A person cannot inspect the inside of a working jet engine, nor even a drain pipe.

Human inspection often comes free.

- Packing and assembly operators can inspect objects without adding (significantly) to the overall cost of the manufacturing process.

Neither a human inspector, nor a fully automated vision system, will always get the best results.

- It is sometimes better to employ a person working in symbiosis with a machine vision system.

C.9 Mechanical Handling

However deformed the widgets are, they must all pass through the inspection system without jamming.

► If the full range of defective widgets cannot be fed properly through the inspection system, then it is of no use whatsoever. It is an irony that one of the main aims of automated visual inspection is to preventing jamming of a mechanical sub-system, such as an assembly machine.

If the parts feed mechanism of the inspection system can go wrong, it most certainly will and the camera will be crushed.

► Be prepared to sacrifice the camera, lighting and/or optical sub-systems, in the event of a failure of the feed mechanism. Design the system accordingly.

C.10 Lighting

Many hands make light work.

► . . . but not very well. However, some people do apply proper engineering principles to the design of the optical sub-system and inevitably obtain better results.

The lighting is not constant.

► Lighting is never constant in either time or in space.

Never use software to compensate for a poor lighting system.

► It is not cost effective and will result in a poor system design.

It is cheaper to add a light-proof shroud to keep sun-light away from the object under inspection than to modify the software.

► Another universal truth which is often forgotten.

Nothing exceeds the speed of light.

► Any processing that can be done optically will save a lot of computer processing later.

It is all done by mirrors.

► Wishful thinking, in view of the previous remark.

C.11 Image Resolution

Any feature whose diameter is equal to 0.1% of the width of the camera's field of view, requires an image resolution better than 2000 × 2000.

► Nyquist's Sampling Theorem places a fundamental limit on the image resolution. This is often forgotten/ignored by advertisers on broadcast television, who frequently place disclaimer notices about their products on the screen, using printing that cannot be read properly because it is too small. The same principal applies to machine vision.

A (100×100) picture is worth 10,000 words.

- The ancients were very astute when they realised that a digital image requires the storage and processing of a lot of data.

One high-quality image is better than five fuzzy pictures.

- Few people would dispute this point.

Five fuzzy pictures are better than one high-quality image.

- No! This does not conflict with the previous proverb. It may be cheaper and easier to obtain the required information from a small set of low-resolution images than to process one very high resolution image. For example, it may be necessary to see several small features within a large scene. In such a case, it might be appropriate, say to use 5 low resolution images (e.g., 256×256), rather than one image of much higher resolution (e.g., 2000×2000).

C.12 Related Disciplines

Machine Vision $\neq$ Computer Vision.

- Machine vision is concerned with Systems Engineering and the solution of practical problems, such as guiding industrial robots, inspection and process monitoring. On the other hand, Computer Vision concentrates on the concepts and scientific basis of vision. The latter is concerned with generic issues and takes inspiration from and is often used to model human and animal vision.

Machine vision research is not a part-time activity for workers in Image Processing, Pattern Recognition, or Artificial Intelligence.

- Some people think it is, unfortunately. The solutions they offer to industrial inspection problems are, at best, unreliable and over-complicated, because they are unaware of the broader "systems issues," such as image acquisition, QA practices, industrial engineering etc.

C.13 Environmental Protection

Protect the machine from the work place.

- A factory is a hostile place, with lots of dirt, tampering fingers, etc.

Protect the work place from the machine.

- Protect eyes from flashing lights, lasers, etc. Make sure that the inspection machine does not shed bits, such as nuts, bolts, etc. to contaminate food products, pharmaceuticals, etc.

It is cheaper to pay for a shroud to enclose strobed light than to pay compensation for causing epileptic fits.

- Flashing lights can trigger epileptic fits and migraine attacks.

The lens may not fit the workman's camera at home, but he thinks it will.

- Be aware of light fingered workers causing damage by removing pieces of equipment.

"He is a good worker and likes to keep things clean - he washes down all of the equipment using a hose-pipe, every afternoon."

- This is quotation from one factory manager about a dedicated, but uninformed worker who did not realise the potential damage and danger his actions could cause. It is imperative therefore that the vision equipment be made safe and robust.

Adjustment of the camera is achieved using a 1 kg hammer.

- Vision engineers will be horrified at this prospect but it may happen.

Factories are dirty places.

- The electrical power supply is noisy. The air supply, for pneumatic equipment, also carries dirt, moisture and oil. Dirt, dust, moisture, fumes, spray, etc. all abound in the local atmosphere.

C.14 Proving and Working with the System in the Factory

Do not assume that the factory workers are computer literate.

- Software should be designed in such a way that it can be used with minimal computer skills.

The people who will make sure that the machine will not work are standing beside it.

- So, the vision engineer should try to persuade them that it is actually in their best interests (as well as his) to work in co-operation with the treasured vision system, not against it.

A picture is worth 10,000 words.

- Give the workers a "television program" to watch. A visual display, showing performance statistics of the vision system and explaining its operation is well worth having, even though it may not seem to be essential.

People "understand" pictures.

- A visual display is a useful way of building the confidence of factory personnel. It is also a valuable diagnostic tool: a person can easily recognise whether a sequence of images, showing the operation of the vision system is being repeated properly.

The service schedule of the vision system should be compatible with the production line.

- If it is not, the vision system will not fit into the factory environment properly.

For every hour you spend on inspecting products, your smarter competitor spends ten hours improving their process.

- Automated inspection is not always the best way to get the desired results.

Document all experiments to validate the system.

- All laboratory and on-site trials in the customer's premises should be fully documented. This should include details about the hardware and software used, parameter settings, optical and lighting set-ups, lens distance, aperture settings and mechanical handling features, how the products were selected.

Quantify the system performance.

- The ability of the system to perform to the agreed specification should be demonstrated and quantified. Accuracy, repeatability, robustness, feature delectability and tolerance of product variation should all be measured and recorded. All demonstrations should be attended by the vision application engineer(s) who are ultimately responsible for the system design and implementation.

Results may not be reproducible.

- Wherever possible, the results of all system performance tests should be reproducible and statistically characterised as to repeatability. In certain applications, for example the inspection of natural products, the variation in product characteristics make it difficult to implement this approach.

Align, calibrate and then test the system before it is used.

- A badly aligned system, or one which has not been calibrated, is likely to produce erroneous but seemingly reasonable results.

Appendix D: MV Knowledge Base

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_45,

Machine Vision spans many diverse technologies. For this reason, a vision engineer must be flexible in his thinking, even if he does not have detailed knowledge about all of them. This is the very nature of Systems Engineering! The knowledge base on which Machine Vision is built does not consist of a single monolithic body of information. Few people can be expert in all of the "component technologies"; most people coming to this subject have a good appreciation of just one of them. A few can justifiably claim skill in two areas but only very rarely more than that. Bearing this in mind, this appendix is presented to define the scope of Machine Vision, by listing those subject areas that *often* impinge on the design of a system. The author found it impossible to prepare a list that covers every eventuality, or that would be universally accepted by his peers. Every time he returned to the keyboard, to refine an earlier draft, he saw its shortcomings and made major changes, only to repeat the process next day. Eventually, it was necessary to "freeze" the list, with many shortcomings still evident to him. What has a emerged is a personal view of the scope of the subject, expressed in the form of a tree-like structure. Other workers will almost certainly disagree about some of the details that reflect their different experiences. A secondary objective of this exercise was to provide a set of web-search terms that will enable the reader to acquire up-to-date product information.

The following diagrams were prepared using the free concept-mapping program, *FreeMind* [Joerg Mueller's FreeMind, URL: http://freemind.sourceforge.net/, accessed 8th February 2010]

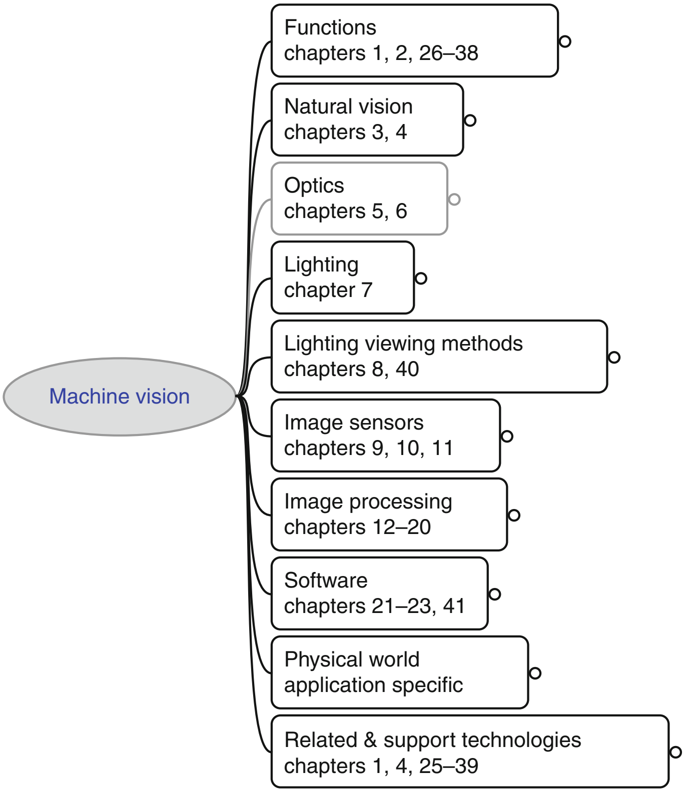

Machine vision
Functions chapters 1, 2, 26–38
Natural vision chapters 3, 4
Optics chapters 5, 6
Lighting chapter 7
Lighting viewing methods chapters 8, 40
Image sensors chapters 9, 10, 11
Image processing chapters 12–20
Software chapters 21–23, 41
Physical world application specific
Related & support technologies chapters 1, 4, 25–39

Fig. D.1
Machine vision: *top level view*

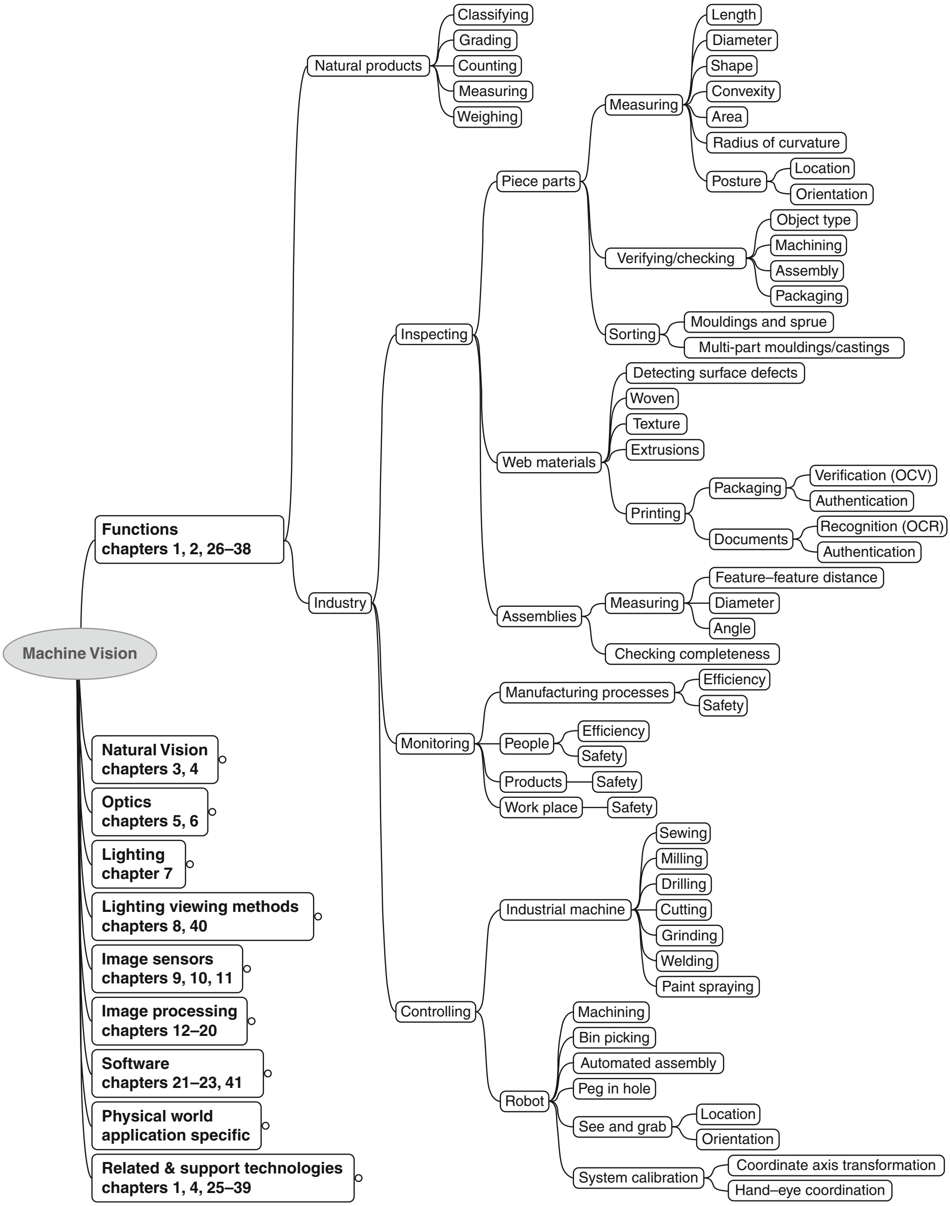


Fig. D.2

Typical functions performed by industrial machine vision systems

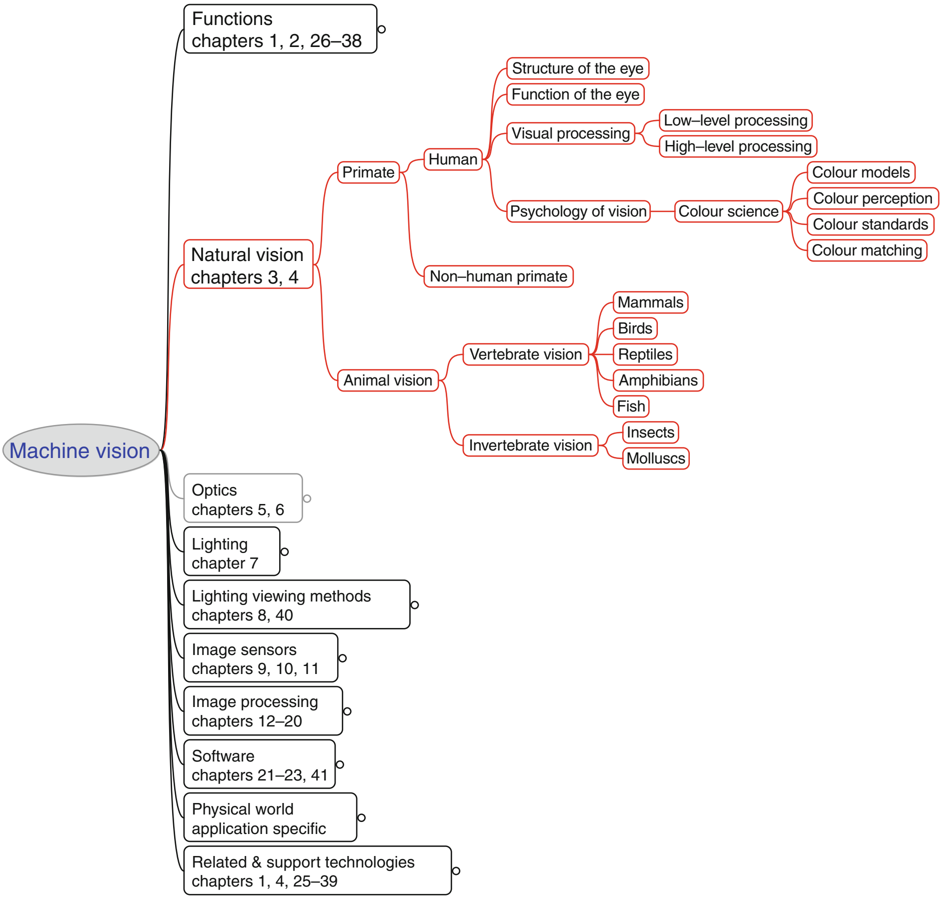
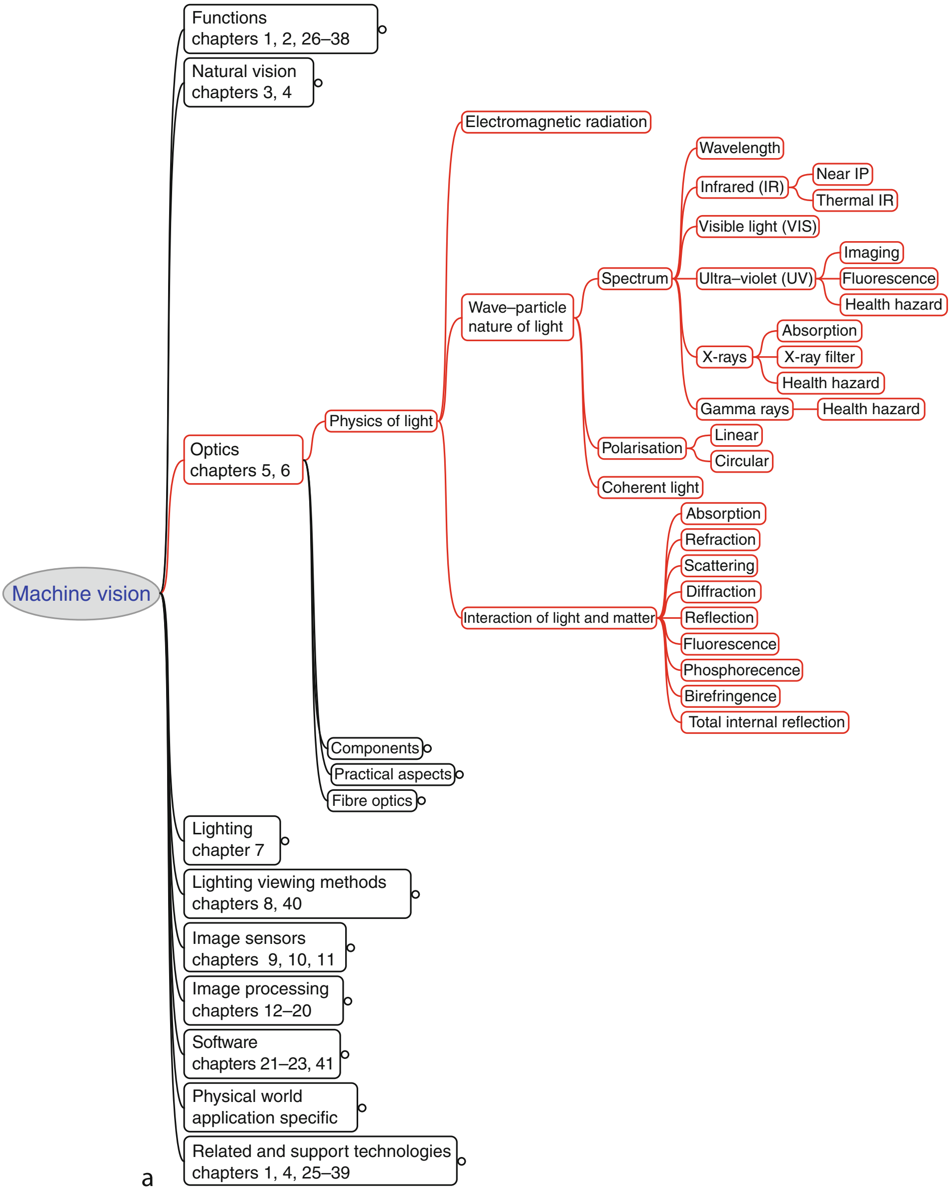


Fig. D.3
Natural vision



■ **Fig. D.4**
(Continued)

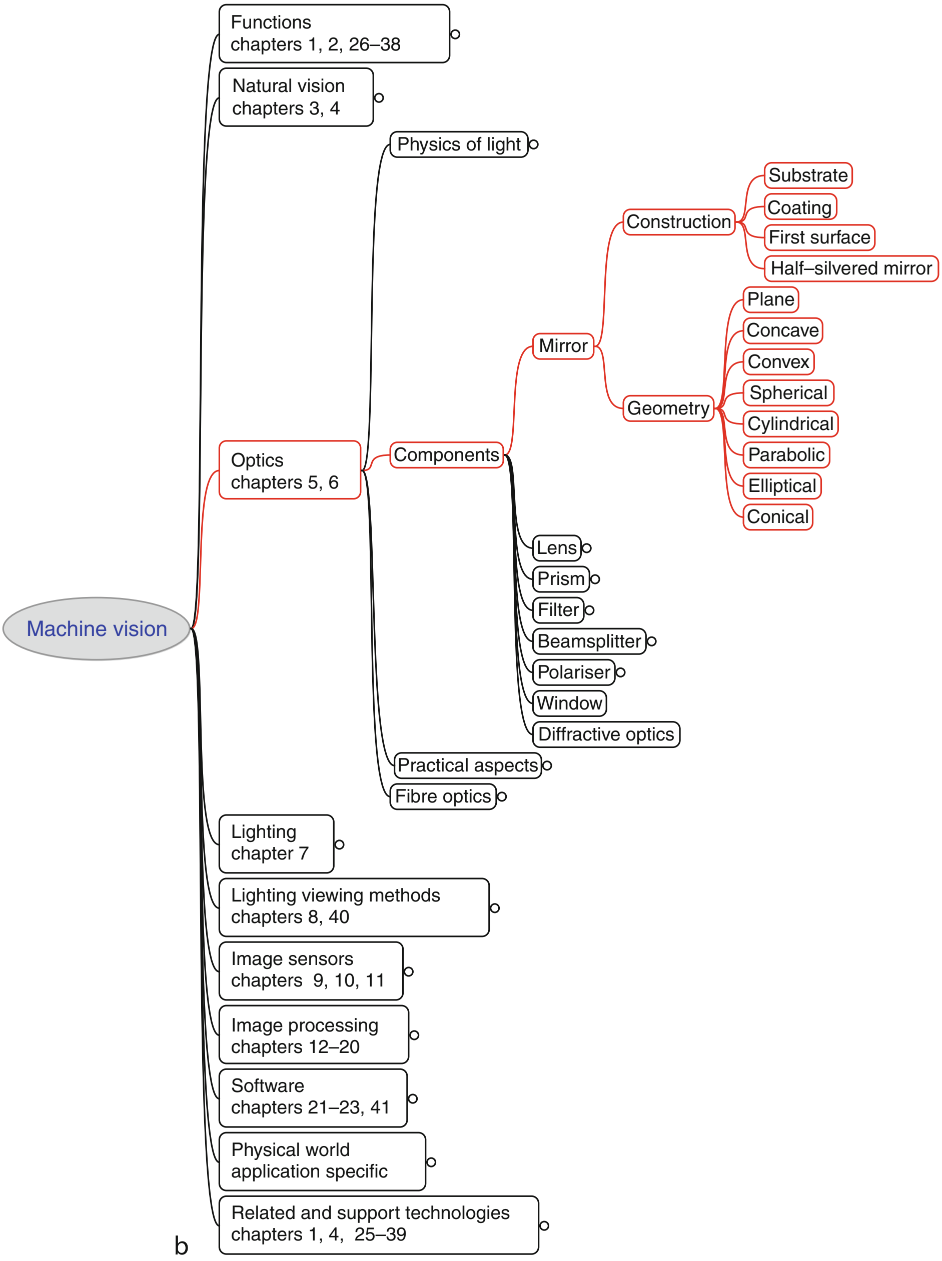
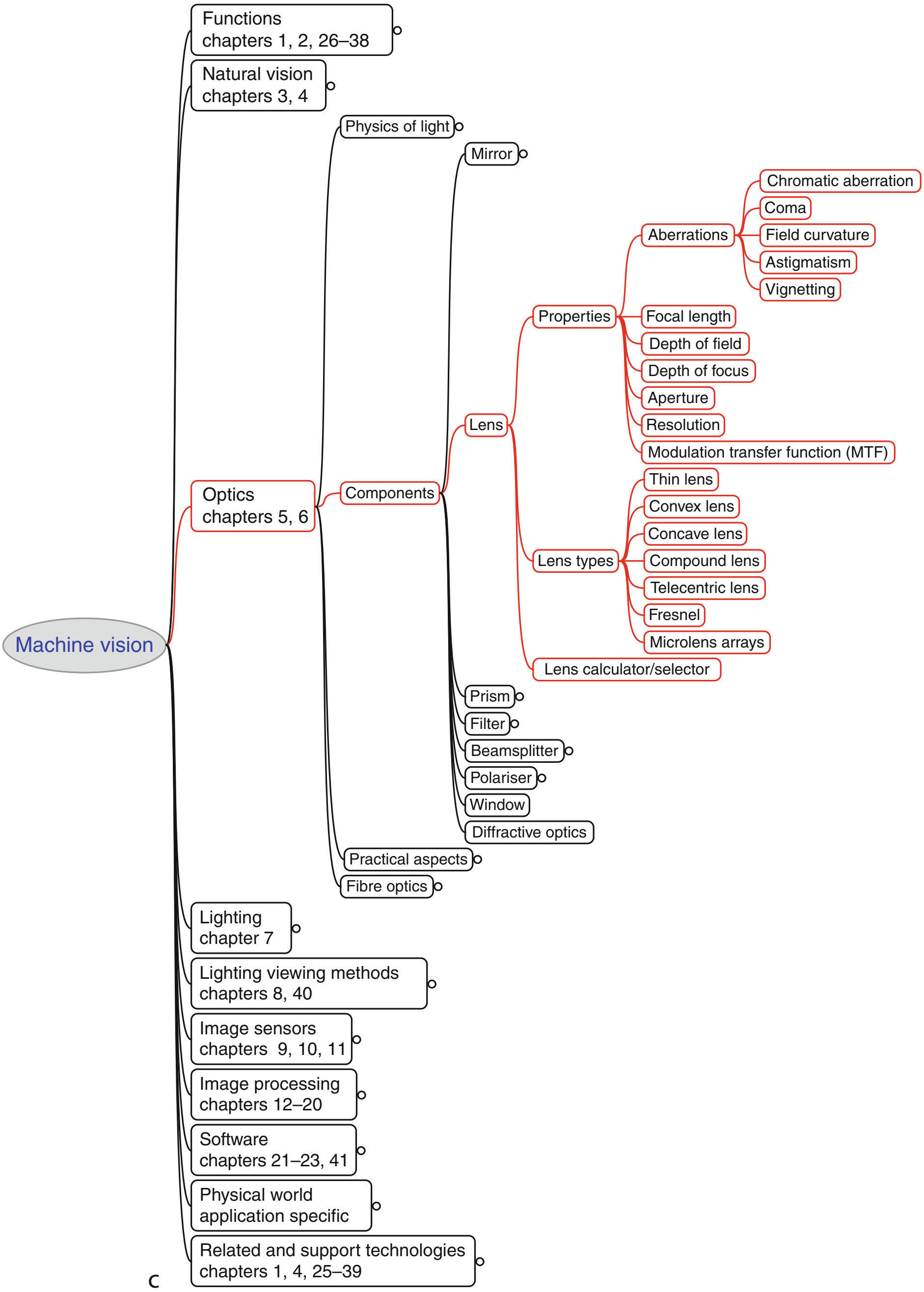


Fig. D.4
(Continued)



■ **Fig. D.4**
(Continued)

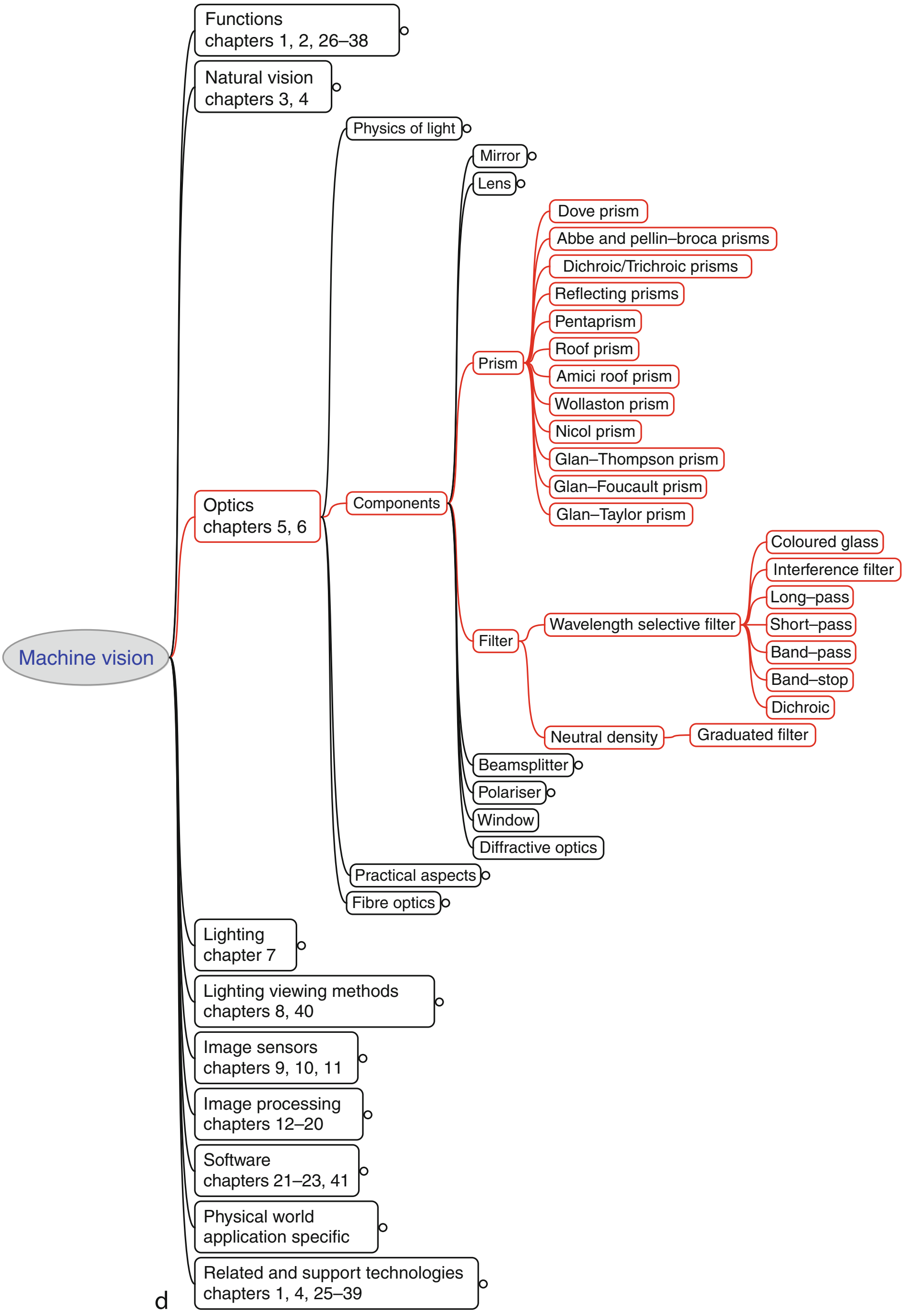


Fig. D.4 (Continued)

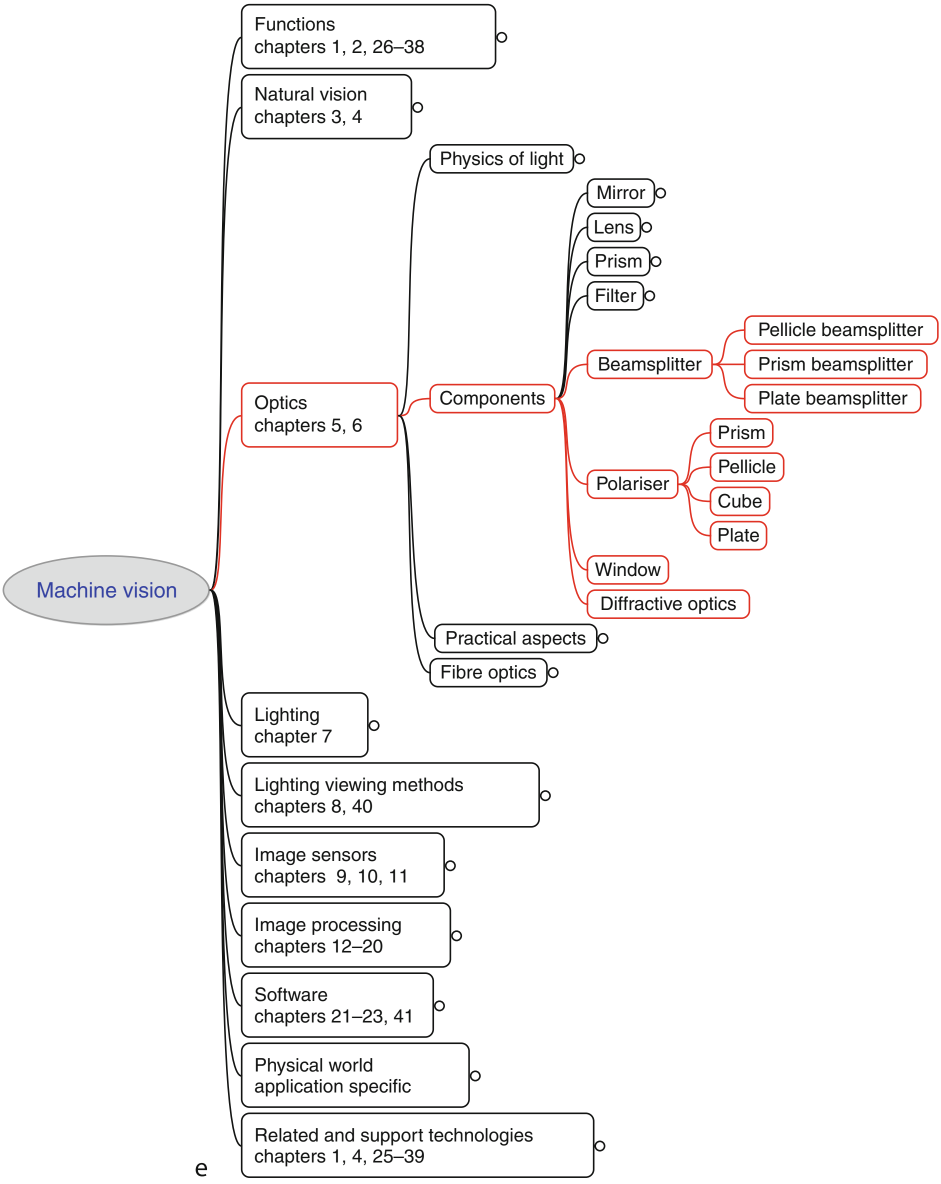


Fig. D.4
(Continued)

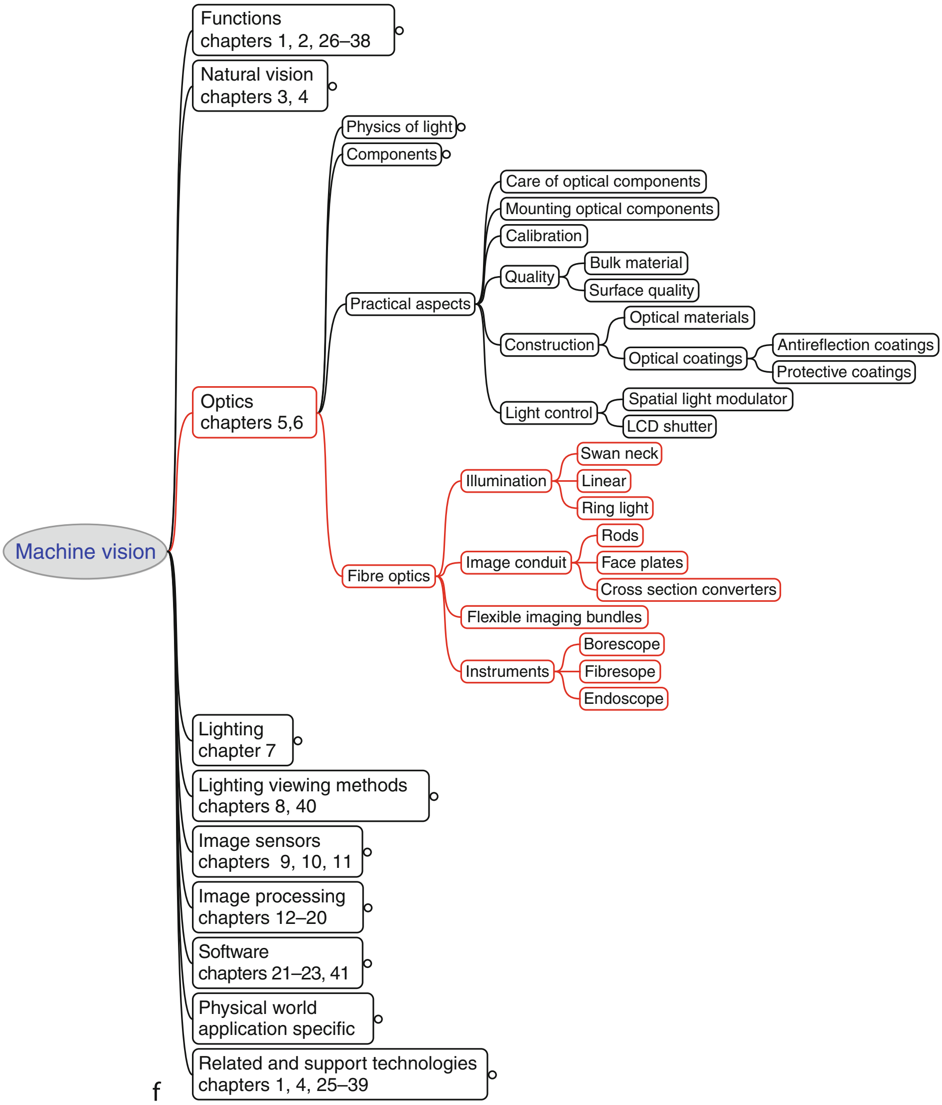
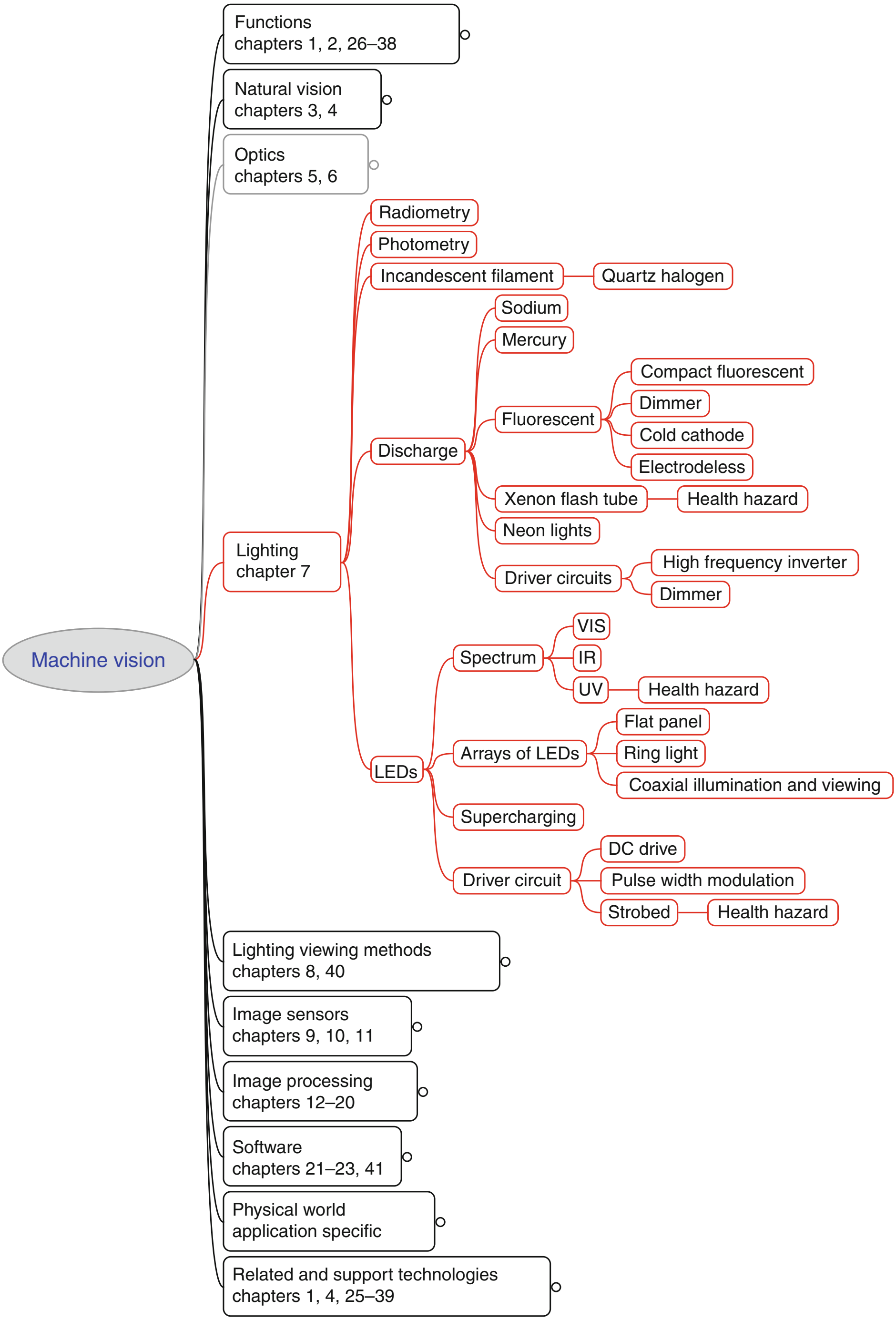


Fig. D.4

Optics (a) Physics of light. (b) Mirrors (c) Lenses (d) Prisms and Filters (e) Beamsplitters, Polarisers, Windows and Diffractive optics. (f) Practical Issues and Fibre Optics



■ **Fig. D.5**
Lighting

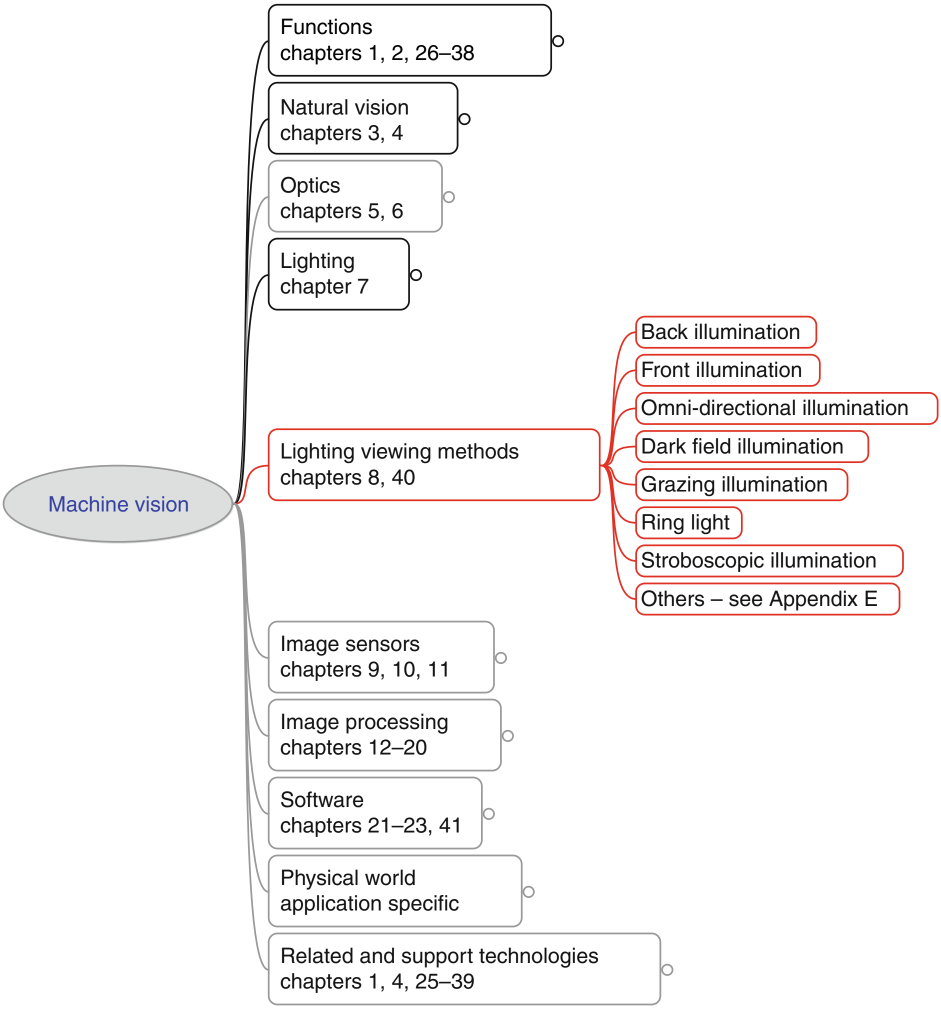


Fig. D.6

Lighting viewing methods (This branch of the tree includes just a small sample of the techniques described in Chap. 40)

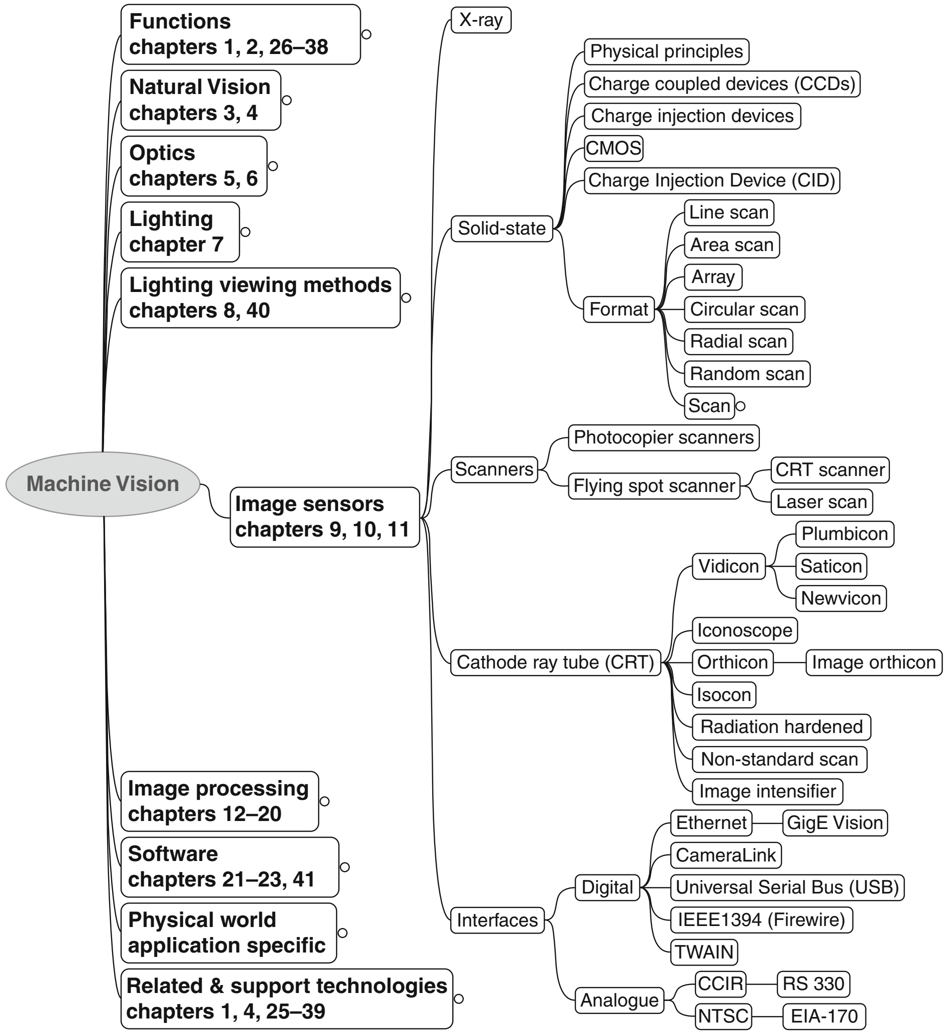


Fig. D.7
Image sensors

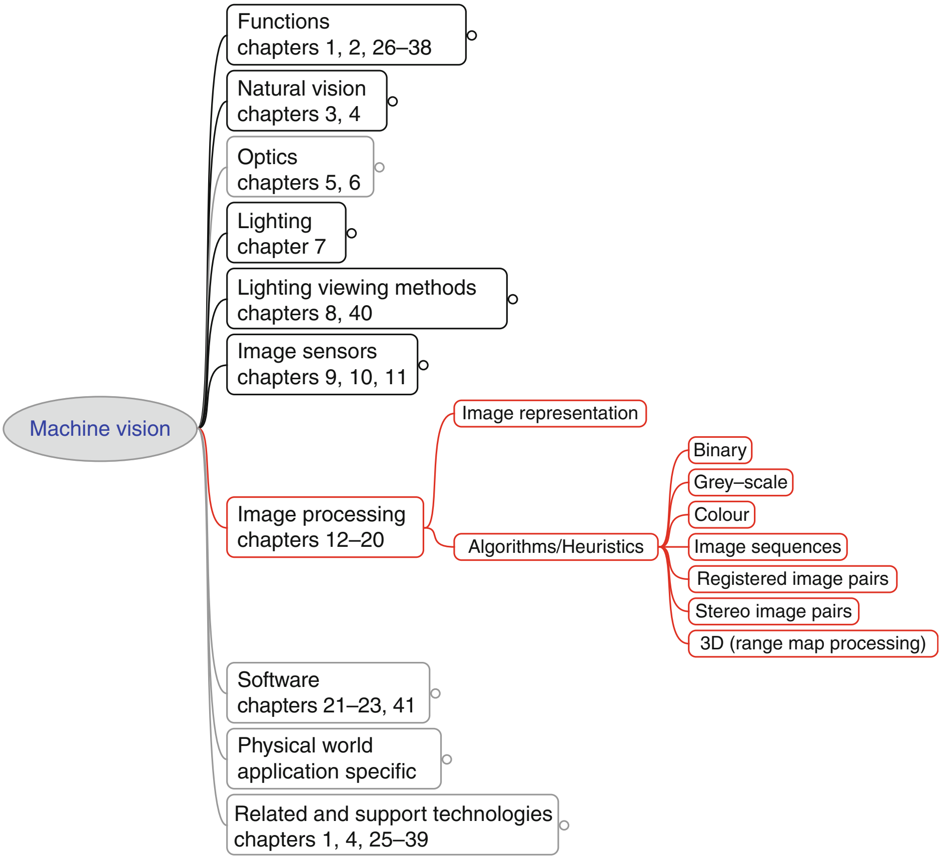


Fig. D.8

Image processing (This branch includes just a small sample of the techniques available.)

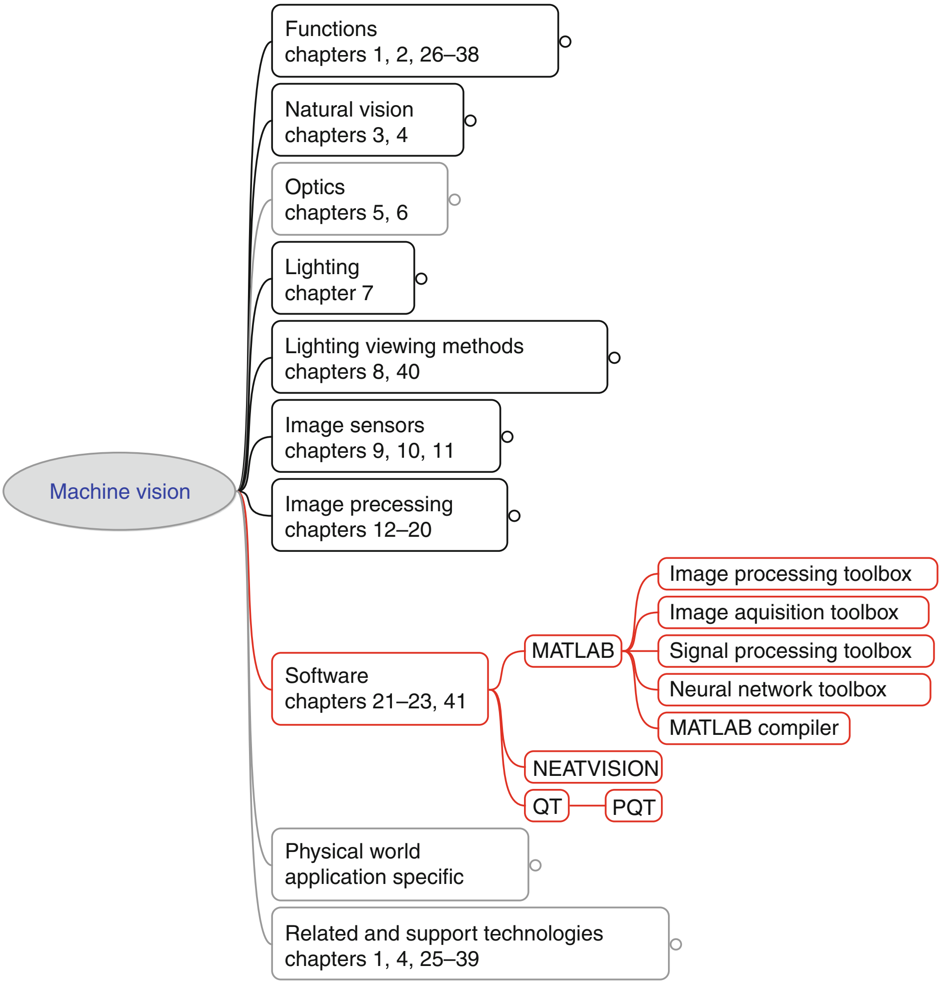


Fig. D.9

Software (Only the three software packages described in this book are listed. Others are available. See Appendix H)

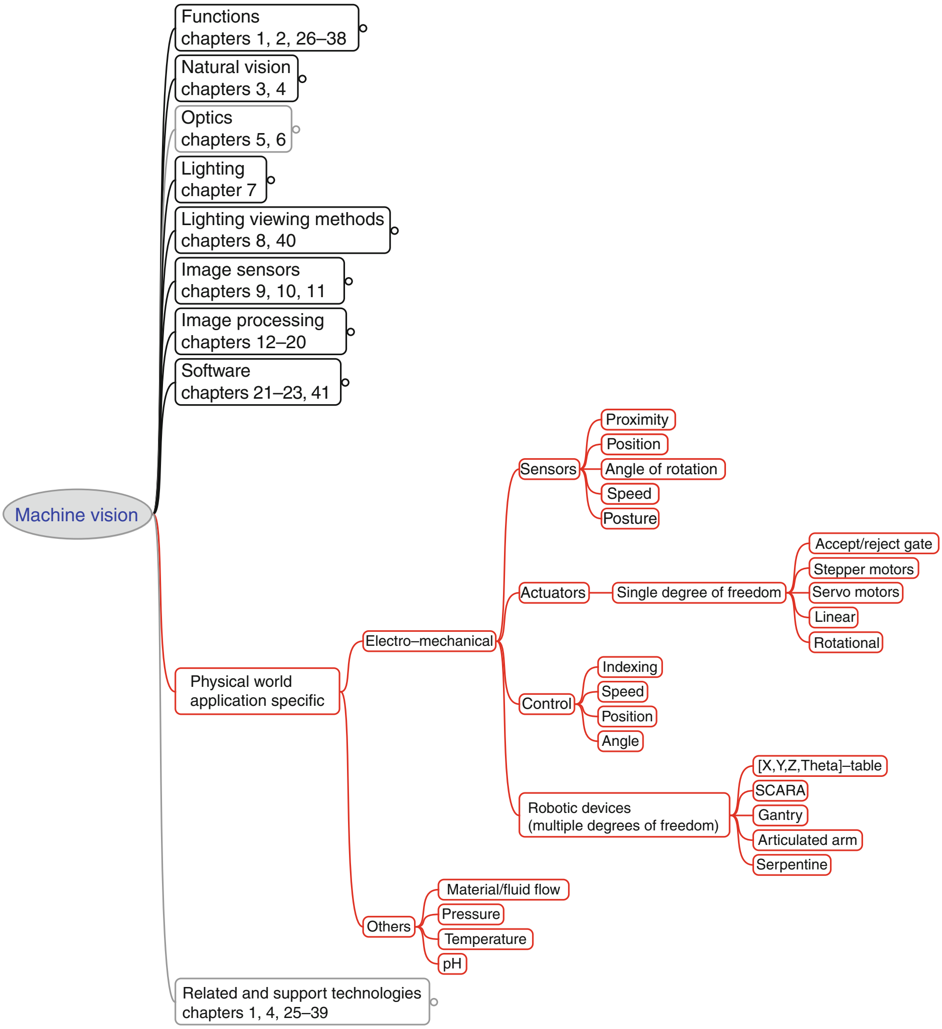


Fig. D.10

Physical world. Many vision systems interact with the physical world by controlling a widget's position. The "Others" branch allows the possibility of integrating a vision system with other real-world data sensors (sensor fusion)

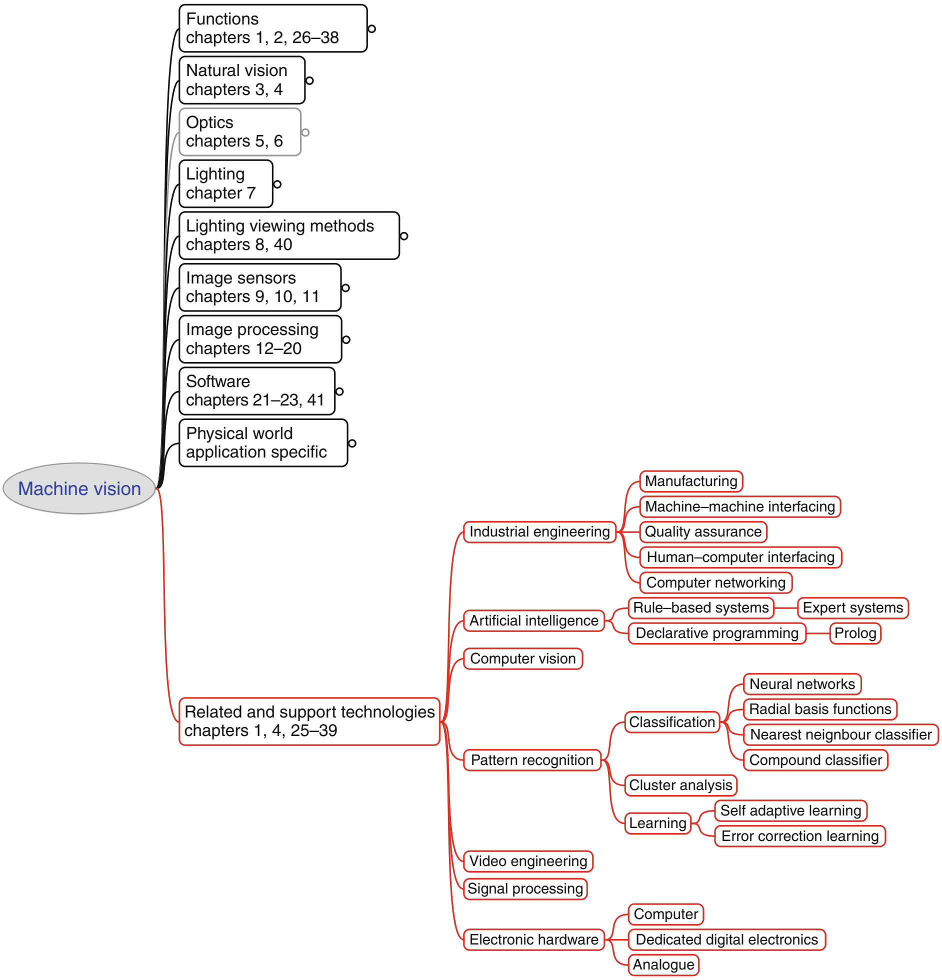


Fig. D.11

Machine Vision sometimes ''borrows'' Computer Vision algorithms but has a distinct ethos. As academic subjects, Artificial Intelligence and Pattern Recognition have separate identities from Machine Vision, although some vision systems incorporate AI and PR techniques. To be successful, a vision system must be integrated into the factory environment and manufacturing-inspection cycle. Most vision systems are specialised electronic calculators, with a video input

Appendix E: Robot Vision: Calibration

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_46,

E.1 Physical Layout

Like most image processors, PQT (Chap. 23) is able to control a range of electromechanical devices, such as the robot vision cell described in *Fig. E.1*. When such a system is first is built, the cameras and robot are not aligned exactly and it is unlikely and the relationship between their coordinate systems is known precisely. Finding the transformation matrices that enable either of the coordinate systems to be mapped into the other is known as *system calibration*. An experimental procedure for calibrating a system comprising an (X,Y,θ)-table and a pick-and-place arm, with an overhead camera is described in this short appendix.

E.2 Why Calibration is Needed

In nearly all visually guided robots, the image processor and manipulator arm rely upon totally different coordinate systems. It is important therefore that, given the address of a point in vision-system space, the system should be transformed into the different set of coordinates used by the robot. In the same way, the robot may have placed, or discovered, an object in a position that is known in terms of its own coordinates. In this case, the vision system needs to be able to interpret that data so that it can properly interpret images.

When a visually guided robot is first set up, it is necessary to calibrate it, so that the two parts of the system can communicate together effectively. In other words, it is necessary to provide some mechanism so that vision system coordinates can be mapped into robot system coordinates and *vice versa*. To begin, we need to calibrate the system and the predicate calibrate_robot is a procedure for calibrating an (X,Y,θ)-table, which is viewed using an overhead camera.

```
calibrate_robot:-
    message(["Place a white disc at, or near, the centre of the (X,Y,Theta)-table"]),
    /* ________________________________________________________________
    The user must respond to this message before the program continues.
    This disc is called the target.
    __________________________________________________________________*/
    home,                           % Put table in the "home" position
    ctm,                            % Display real-time video from the camera
    offset(Rx1,Ry1),                % Consult database for initial offset parameters
    move_table(Rx1,Ry1),            % Move table to [Rx1,Ry1]
    locate_target(X11,Y11),         % Use QT to locate the target
    turntable(180),                 % Rotate table by 180°
    locate_target(X12,Y12),         % Locate the target again, using QT
    turntable(0),                   % Reset the table to 0° position
    X1 is 0.5*(X11+X12),            % Now find centre of the table – X coordinate
    Y1 is 0.5*(Y11+Y12),            % Now find centre of the table – Y coordinate
    get_scales(Sx,Sy),              % Calculate scale factors
    Rx2 is Rx1+Sx*(256 – X1),       % Calculate movement of table
    Ry2 is Ry1+Sy*(256 – Y1),
    integer(Rx2,Rx3),
        integer(Ry2,Ry3),
        move_table(Rx3,Ry3),                   % Shift table by known amount
        locate_target(X2,Y2),                  % Locate the target again
        nl,
        write(“New offset parameters are: “),
        write([Rx3,Ry3]),
        nl,
        write(“The centroid of the target is now at”),
        write([X2,Y2]),
        nl,
        assert(offset(Rx3,Ry3)),               % Revise the offset parameters . . .
        retract(offset(Rx1,Ry1)),              % . . . and forget the old values
        grb,                                   % Digitise an image
        vpl(256,1,256,511,255),                % Plot vertical white line
        vpl(1,256,511,256,255),                % Plot horizontal white line
        define_robot_limits.                   % Draw box to show limits of travel of the table

get_scales(Sx,Sy):-
        calibrate_incs(Xinc,Yinc),             % Consult database
        locate_target(X1,Y1),                  % Find where target is in terms of vision coords.
        delta_x(Xinc),                         % Move table . . .
        delta_y(Yinc),                         % . . . by fixed amount
        locate_target(X2,Y2),                  % Locate target again
        Sx is Xinc/(X2 – X1),                  % Calculate the . . .
        Sy is Yinc/(Y2 – Y1),                  % . . . scale factors
        nl,                                    % Information for the user
        write(“Scale parameters:”),
        write([Sx,Sy]),
        scale_multipliers(S1,S2),
        assert(scale_multipliers(Sx,Sy)),
        retract(scale_multipliers(S1,S2)),
        nl.

% Assume that the target is a white disc on a dark background

locate_target(X,Y):-
        grb,                                   % Digitise an image.
        thr(64),                               % Arbitrary threshold parameter
        ndo,                                   % Shade objects by size
        thr(255,255),                          % Select the biggest to isolate target
        blf,                                   % Make sure target appears as a solid figure
        cgr(X,Y).                              % Calculate centroid of the blob

% Initial values of calibration constants. These are typical values

calibrate_incs(500,500).                       % Constants for additive term of conversion
                                               formula
scale_multipliers(22.727, −20).                % Scale factors for axis conversion
```

```
offset(5,134, −1,223).                            % Offset constants (i.e. The movement
                                                  needed to shift the table
                                                  % from pick_up point to centre of camera’s
                                                  field of view

/* The first clause is merely a convenience for the user. It draws a rectangle on the image display so
that the range of possible movements of the centre of the table can be visualised */

define_robot_limits:-
    define_robot_limits(_,_,_,_).

define_robot_limits(X3,Y3,X4,Y4):-
    axis_transformation(−7,500,−5,000,X1,Y1),     % Limits of table motion
    axis_transformation(7,500,5,000,X2,Y2),       % Limits of table motion
    image_limits(X1,Y1,X3,Y3),                    % Check valid vision system coordinates
    image_limits(X2,Y2,X4,Y4),                    % Check valid vision system coordinates
    nl,                                           % Information for the user
    write(“Robot movement limits:”),
    write([[X3,Y3],[X4,Y4]]),
    nl,
    vpl(X3,Y3,X4,Y3,255),                         % Draw a rectangle
    vpl(X3,Y3,X3,Y4,255),
    vpl(X4,Y3,X4,Y4,255),
    vpl(X3,Y4,X4,Y4,255).

% What to do if the point is outside the picture

image_limits(X1,Y1,X2,Y2):-
    size_limits(X1,X2),                           % Check X variable
    size_limits(Y1,Y2).                           % Check Y variable

size_limits(A,0):- A<0.                           % Hard limiting at edge of the image

size_limits(A,511):- A>511.                       % Hard limiting at edge of the image

size_limits(A,A).                                 % Address variable is O.K so keep it
```

E.3 Converting Vision System Coordinates to Robot Coordinates

The predicate *axis_transformation/4* uses the parameters calculated by *calibrate_robot/0* to transform vision system coordinates to robot coordinates, or vice versa.

```
% Convert Vision System Coordinates to Robot Coordinates

axis_transformation(Xr,Yr,Xv,Yv):-
    not(var(Xv)),                                 % Vision system coordinates are known
    not(var(Yv)),                                 % Vision system coordinates are known
    var(Xr),                                      % Robot coordinates are not known
    var(Yr),                                      % Robot coordinates are not known
    offset(Dx,Dy),                          % Consult database
    scale_multipliers(Sx,Sy),           % Consult database
    Xr1 is Dx+Sx*(Xv –256),             % Calculate robot’s X coordinate, floating point
    Yr1 is Dy+Sy*(Yv –256),             % Calculate robot’s Y coordinate, floating point
    integer(Xr1,Xr),                    % Convert to integer
    integer(Yr1,Yr),                    % Convert to integer
    Xr ≤ 7,500,                         % Check range of answer; can robot reach point?
    Xr ≥ −7,500,                        % Check range of answer; can robot reach point?
    Yr ≤ 5,000,                         % Check range of answer; can robot reach point?
    Yr ≥ −5,000.                        % Check range of answer; can robot reach point?

% The robot cannot reach the point specified

axis_transformation(Xr,Yr,Xv,Yv):-
    not(var(Xv)),                        % Vision system coordinates are known
    not(var(Yv)),                        % Vision system coordinates are known
    var(Xr),                             % Robot coordinates are not known
    var(Yr),                             % Robot coordinates are not known
    message([“Out of range of the robot”]),  % Tell user about the problem
    !,
    fail.                                % Force failure if we get this far

% Convert Robot Coordinates to Vision System Coordinates

axis_transformation(Xr,Yr,Xv,Yv):-
    not(var(Xr)),                        % Robot coordinates are known
    not(var(Yr)),                        % Robot coordinates are known
    var(Xv),                             % Vision system coordinates are not known
    var(Yv),                             % Vision system coordinates are not known
    offset(Dx,Dy),                       % Consult database
    scale_multipliers(Sx,Sy),            % Consult database
    Xv1 is 256.5+(Xr–Dx) / Sx,           % Calculate vision system X coordinate
    Yv1 is 256.5+(Yr – Dy) / Sy,         % Calculate vision system Y coordinate
    integer(Xv1,Xv),                     % Convert to integer
    integer(Yv1,Yv).                     % Convert to integer


axis_transformation(_,_,_,_):-
    message([“Error in the way you have specified the parameters”]),
    !,
    fail.                                % Force failure if we get this far
```

E.4 Moving the Robot

Let us consider now how *axis_transformation/4* can be used. Normally, of course, the vision system will calculate the position of some feature in an image before the robot is moved. (The

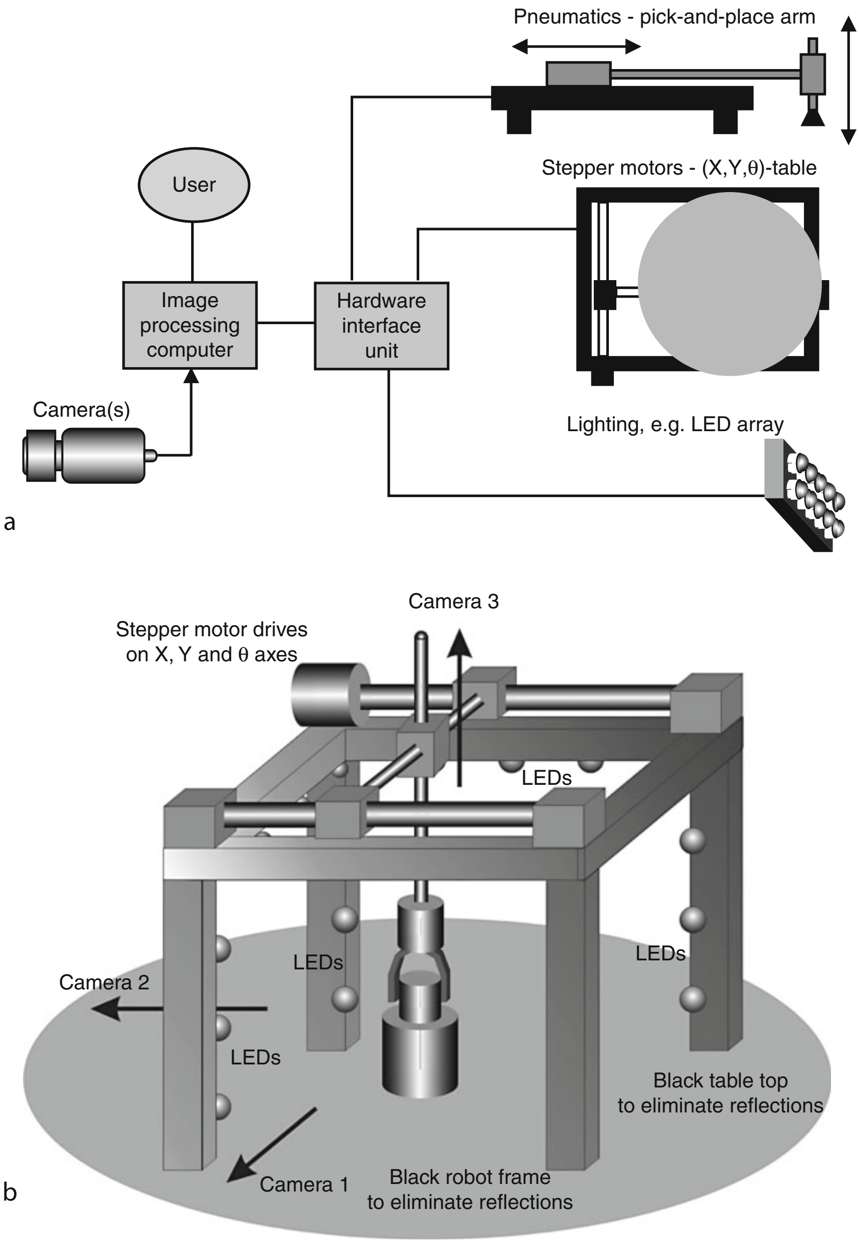

Pneumatics - pick-and-place arm
Stepper motors - (X,Y,θ)-table
User
Image processing computer
Hardware interface unit
Camera(s)
Lighting, e.g. LED array
a
Camera 3
Stepper motor drives on X, Y and θ axes
LEDs
LEDs
LEDs
LEDs
Camera 2
Black table top to eliminate reflections
Camera 1
Black robot frame to eliminate reflections
b

Fig. E.1

Robot vision cell for parts inspection. (a) Device control. (b) Flexible Inspection Cell. This is able to inspect objects from many different directions, possibly under changing lighting conditions. This particular layout, using a gantry robot, has 4 degrees of freedom. If the vertical movement is limited to just two positions (*up/down*),it has 3$^1/_2$ degrees of freedom, which is the same as an (X,Y,θ)-table with a pick-and-place arm. (➲ Chap. 40, LVM: Flexible Inspection Cell) This appendix relates to the coordination of the robot with Camera 3

objective might be to pick up an object, drill a hole, insert a screw, weld a work-piece, or perform some similar task.) This is how it can be coded in PQT.

```
locate_feature(X,Y),                        % Use vision system to locate a feature
axis_transformation(Xrobot,Yrobot,X,Y)      % Find where feature is in robot space
```

The predicate *move_robot_to/2* allows the programmer to work with the vision system coordinates without ever bothering about the fact that the robot uses a different set of coordinates.

```
move_robot_to(X,Y):-
    axis_transformation(Xr,Yr,X,Y),         % Convert to robot system coordinates
    move_table(Xr,Yr).                      % Move the table
```

Notice that we have defined the software in such a way that *move_robot_to(256,256)* locates the table so that its centre is at the centre of the field of view of the camera. (We assume here that the image resolution is 512*512 pixels.)

We are now in a position to be able to control the robot in a very simple way:

```
locate_feature(X,Y),                        % Use vision system to locate a feature
move_robot_to(X,Y),                         % Move robot to feature found above
pick_up,                                    % Grasp object pick it up. Not defined here
```

In some situations, the robot coordinates may be known. For example, the robot may might encounter an obstacle so that it cannot that it cannot move further in a given direction. Assuming that the robot knows where it is in space, it would then be helpful if the vision system coordinates could be calculated, so that the obstacle might be identified in an image. This task could be achieved in the following way:

```
obstacle(X,Y),                              % Robot locates an obstacle
axis_transformation(X,Y,Xvision,Yvision),   % Transform the axes
interpet_obstacle(Xvision,Yvision)          % Interpret the scene visually
```

We have ignored an important point here: the robot may have no way of knowing where it is in space. However, PQT can keep a track of its movements. The use of *properties* is generally to be preferred to that of *assert* and *retract*, since they can be adjusted more easily and are faster. Here is a revised definition of *move_robot_to/2*:

```
move_robot_to(X,Y):-
    axis_transformation(Xr,Yr,X,Y),         % Convert to robot system coordinates
    move_table(Xr,Yr),                      % Move the table
    set_prop(robot_vision,[Xr,Yr]).         % Store [Xr,Yr] in variable robot_vision
```

set_prop/2 and *get_prop/2* (used below) are not standard Prolog features. In order to find where the robot is, in terms of its own coordinate system, we may use the following predicate:

```
locate_robot(X,Y):-
```

```
    get_prop(robot_position,[X,Y]).           % Recall [X,Y] from stored variable robot_vision
```

whereas, the predicate *where_is_robot* locates the robot in terms of the vision system coordinates

```
where_is_robot(X,Y):-
    get_prop(robot_position,[Xr,Yr]),
    axis_transformation(Xr,Yr,X,Y).
```

The final topic in this section is that of moving the robot, in such a way that an object which has been dropped onto the (X,Y,θ)-table in random position is normalised. Suppose that the vision system, using an overhead camera, has been able to determine that the centred of an object is at [X,Y] (using vision system coordinates) and at orientation A. The normalisation process moves the table so that the centroid is located at the middle of the camera's field of view and rotates the object to an orientation of 0°.

```
normalise(X,Y,A):-
    B is -A,                                   % Calculate negative of A
    turntable(B),                              % Rotate table by angle B
    cosine(A,Cos),                             % Calculate cosine, A is in degrees
    sine(A,Sin),                               % Calculate sine, A is in degrees
    scale_multipliers(Sx,Sy),                  % Consult database for scale multipliers
    X1 is -Sx*((X – 256)*Cos+(Y – 256)*Sin),
    Y1 is -Sy*(–(X – 256)*Sin+(Y – 256)*Cos),
    integer(X1,X2),
    integer(Y1,Y2),
    delta_x(X2),                               % Increment X axis of table
    delta_y(Y2),                               % Increment Y axis of table
    grb,                                       % Digitise another image
    vpl(1,256,511,256,255),                    % Draw cross-lines . . .
    vpl(256,1,256,511,255),                    % . . . at centre of the image
    draw_robot_jaws.                           % See below
```

Of course, it would be helpful to know where the gripper will be placed when the robot tries to pick up an object. No gripper is of insignificant size, so a mis-aimed move might put the object in such a position that it and/or the gripper and robot could be damaged. Let us assume first that the robot gripper has two parallel jaws, which open and shut in a simple pincer movement. It is possible to draw rectangles on the image which represent the gripper jaws in the open/shut state (see ➲ Chap. 38). The user can of course, easily verify visually that the gripper will not collide with the object that is to be picked up. However, a little more processing allows this decision to be made automatically.

```
safe_to_lift(X,Y):-
    move_robot_to(X,Y),                        % Move the robot as described above
    grb,                                       % Digitise another image
    convert_to_binary,                         /* Convert image to binary. Object is white
                                               & background is black.
    swi,                                       % Switch images
```

```
         draw_robot_jaws(open),                      % Draw the jaws
         mni,                          % AND the two images together
         cwp(0),                       % There is no overlap. So it is safe to lift object
         lift_object.                  % Operate pick and place arm. Not specified

safe_to_lift(_,_):-
         message([“Warning: The robot cannot safely lift the object”]),
         !,
         fail.                         % Force failure
```

Appendix F: Object Location and Orientation

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_47,

F.1 The Problem

In this appendix, we present a range of methods that can be used to guide a robot to pick up an object (widget) lying in an unknown position and orientation on a plain-topped table. We shall introduce certain other simplifications:

1. The camera is vertically above the widget.
2. The widget is entirely within the camera's field of view and there is no occlusion by any other object.
3. If there is more than one object in view, no widget touches any other one.
4. There are no artefacts due to poor lighting; there are no significant shadows and there is no glinting from either the widget or table-top.
5. There is sufficient contrast to ensure that features that might be used to locate and orientate the widgets are clearly visible.

The layout of the robot vision cell is outlined in Appendix E. We shall assume that the robot has already been calibrated. (Appendix E)

Let us begin with a simple challenge: the child's toy shown in *Fig. F.1*. Despite the fact that a child of about 2 years of age can place the pieces into the appropriate slots, this presents certain (but not insurmountable) difficulties for a vision system. The problem is that that many of the techniques for locating an object and determining its orientation are unable to cope with all six shapes.

Figure F.2 presents another multi-shape application. There are four objects lying on an (X, Y, θ)-table. The geometric centroid, usually referred to simply as the *centroid*, is able to provide a reasonably good way to find the position of an object, although in the case of the Y-shaped component, the centroid lies very close to the edge. The orientation for three of the objects shown in *Fig. F.2* can be found reliably using the principal axis but not for the 3-pointed "star."

F.1.1 Mechanical Alignment

In Machine Vision, we should not over-rely on image processing; sometimes a simple mechanical handling arrangement will help to fix the orientation within quite narrow limits. *Figure F.3* shows how simple guide rails, placed over a conveyor belt, are able to fix object

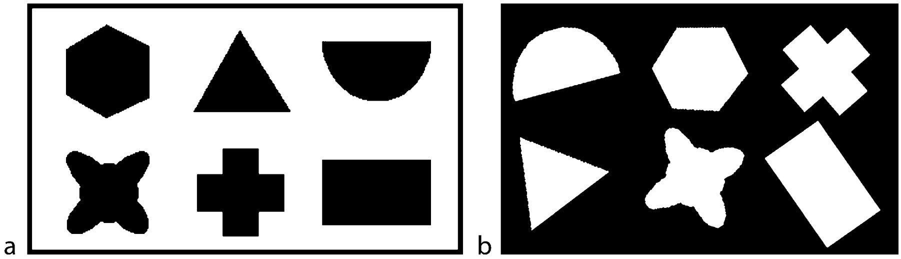


Fig. F.1

Child's toy, represented diagrammatically. (a) A plastic template with holes of six different shapes. (b) Plastic pieces that are to be placed in the slots on the template

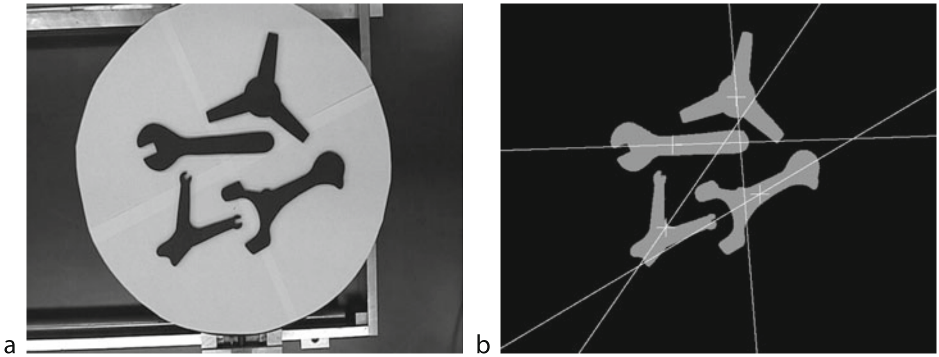


Fig. F.2
Objects on an (X, Y, θ)-table. (a) What the camera "sees." (b) After noise reduction and thresholding a "clean" binary image was obtained. The centroid position (QT function cgr) and the orientation of the principal axis (lmi) were found. The principal axis for each component has been drawn (dpa). This provides a satisfactory way to determine the orientation for three components but not for the three-pointed "star." (This is due to the three-fold symmetry of this object.)

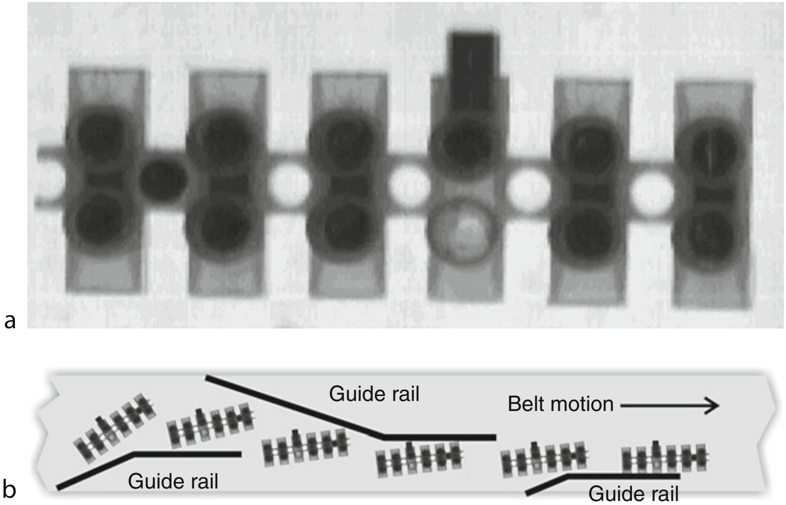


Fig. F.3
Using guide rails over a conveyor belt to limit the angular deviation of objects in front of the camera. (a) A suitable object: an electrical connection block (faulty). (b) Mechanical arrangement

orientation to within a few degrees. In some applications, this may make it possible to improve the speed of vision-based measurement of object position and the orientation. For example, the Radon Transform (QT function *rdn*) can be made much faster if we know that the objects to be inspected will always arrive, within a small angular deviation from the norm. (e.g., ±5%).

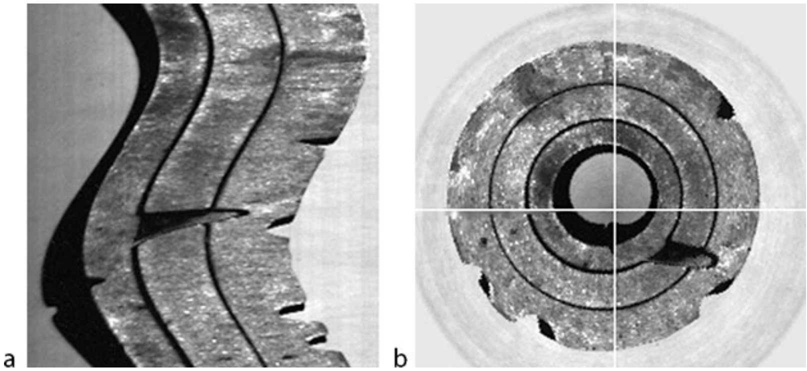


Fig. F.4

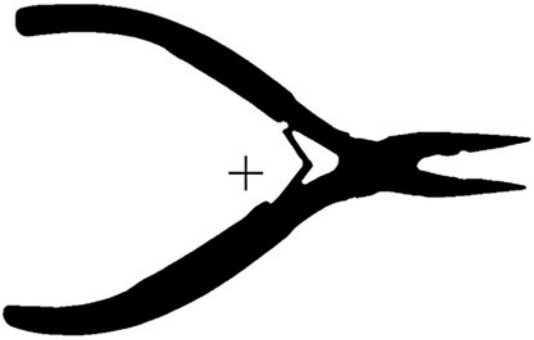
Circular component viewed on a turn-table using a line-scan camera. (a) The image obtained from the camera. Horizontal axis: radial distance from the centre of the turntable. Vertical axis: angular position. (b) The image in (a) was transformed using the QT command ptc

Sometimes, objects are presented for inspection on a turn-table. Using a line-scan camera viewing the light pattern along a turn-table radius, it is possible to obtain a digital image with polar-coordinate axes: angular position and radius (➲ *Fig. F.4*). In the target system that was eventually built for this application, the circular widget was placed centrally on the turn-table, using a robot with a self-centring gripper. The vision system was required to measure the orientation to within ±0.5°. This was achieved by using the image processor to locate shadows cast by low-angle illumination. This emphasises the important point made repeatedly in this book: by optimising the lighting-viewing arrangement, it is often possible to simplify the image processing.

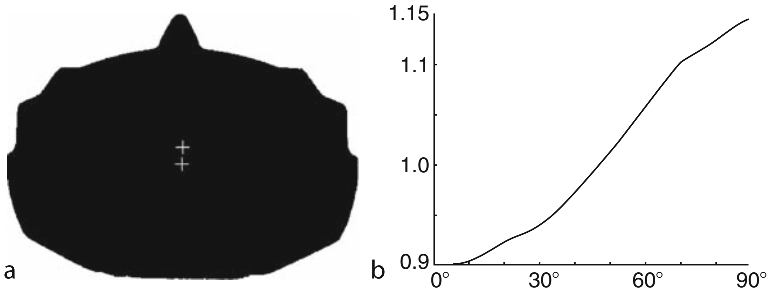
F.2 Finding Object Position

Even the seemingly straightforward task of measuring position is not as straightforward as it first seems. Every object has a finite size; your feet are on the ground but your head is in the air! As a result, there is no unique answer to the question *Where is object XXX*? To achieve safe lifting, we usually pick up an object by grasping it close to its centre of gravity but we pick up a sharp knife by holding its handle. We cannot pick up a horse-shoe by grasping it at its centre of gravity but we may well use the geometric centroid to decide where we should place our hand (➲ *Fig. F.5*). In many application, there are good reasons for trying to pick up an object by placing the robot's gripper close to its centre of gravity, as this minimises the torque exerted by the weight of the widget on the gripper. Unless we know the three-dimensional structure and material composition, we do not know exactly where the centre of gravity is. A reasonable guess, unless we know otherwise, is to assume that it is close to the geometric centroid but these two points may be far apart. Of course, using vision alone, we can only find where the centroid is. *Every application must be analysed individually.*

Centroid The most popular way to obtain position information is by locating the geometric centroid. Numerous examples are given here and elsewhere in this book (See for example, ➲ *Figs. F.2*, ➲ *F.12*, ➲ *F.14* and ➲ *F.16*). The centroid position can be computed rapidly from two simple formulae, which produce precise precise coordinates. Many companies use this fact to justify a claim that their equipment is able to obtain sub-pixel positional information.



■ **Fig. F.5**
The centroid (*black cross*) is not always a suitable grasping point



■ **Fig. F.6**
Minimum-area rectangle (MAR) surrounding the silhouette of an automobile brake pad. (a) The centre of the MAR is indicated by the upper cross and the centroid by the lower cross. (b) The aspect ratio, *R*(*θ*), of the MAR plotted against *θ*, angle of rotation. Within the range 0 ≤ *θ* < 90°, *R*(*θ*) is a monotonic function of *θ*, but *R*(*θ*) = *R*(*θ* + 180*n*), *n* = 1,2,...

Extreme Points/Minimum Bounding Rectangle (➲ *Fig. F.6*). Computing the extreme points in a binary image is a fast way to estimate object position. There are eight such points of interest: *top-left, top-right, bottom-left, bottom-right, left-bottom, right-top, left-bottom, right-bottom*. (QT has functions to measure each one separately, while *mar* derives the coordinates of the corners of the minimum-area enclosing rectangle, MAR). In certain applications, such as handling ill-formed dollops of semi-fluid material (e.g., uncooked dough) the extreme points are actually more useful than a single parameter, such as the centroid, since they limit object size as well. Although these measurements are sensitive to object orientation, they may be useful for determining both object position and orientation. In ➲ *Fig. F.6*, the centre of the MAR is close to the geometric centroid. Such a small offset may not be important. For certain shapes and for a limited range of angles, the aspect ratio of the MAR is a monotonic function of the angle of orientation. Hence, by measuring the MAR aspect ratio and using a small look-up table, the object orientation may be estimated quickly. For shapes that are almost circular, such as a gear, the variation of the position and aspect ratio of the MAR with orientation may not matter at all. Also see ➲ *Fig. F.9*, where calculating the extreme points forms just part of an intelligent heuristic procedure for model fitting.

Bounding Circle Sometimes, it may be advantageous to fit a circle to the object silhouette, then use the centre of that circle as an indicator of position. This is appropriate, for example, when the widget is nearly circular. The QT command *mbc* computes what is erroneously called the *minimum bounding circle*. (➲ *Fig. F.7*) This attempts to fit a circle tightly around as many of the pixels on the outer edge as possible. However, points that are too far inside the circle are ignored. This is, of course, a recursive definition, so *mbc* does not have a fixed execution time. Other ways to fit circles are discussed below.

Largest Internal Circle Robotic grippers do not always close to grasp an object; sometimes they can open instead. (➲ Chap. 38). A three-finger gripper of this type can be inserted into a circular hole to provide a self-centring mechanism for lifting objects. (This is similar to the human action of lifting a jar by inserting the thumb and two fingers into its neck.) The QT commands *lic* and *mic* calculate the maximum size of circle(s) that can be expanded to fit within a given space. The centre of the largest such circle (*lic*) might be used as an indicator of object position (➲ *Fig. F.8*).

Model fitting In some situations, there are no precisely defined visual features, such as holes or sharp corners, available to measure position accurately. Consider an ideal cube: precise positional information can be obtained from its corners. If the corners are rounded, as in standard gaming dice, estimates of position based on corners will be imprecise. However, it is still possible to calculate the position accurately, by observing the edges of the cube. Then, we might conveniently fit straight lines to the sides and observe where they intersect. In some situations, this can provide precise position information. We can use many different models but will limit ourselves to describing the fitting of lines, circles and polygonal curves. On its own, model fitting may not provide a complete answer to object location but it can often be a useful step along the way to achieving that goal.

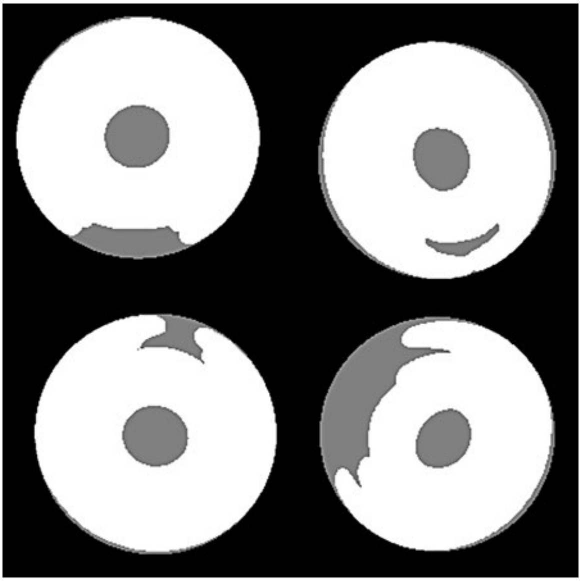


Fig. F.7

Four faulty plastic mouldings for battery tops (*white*) and the minimum bounding circle for each one

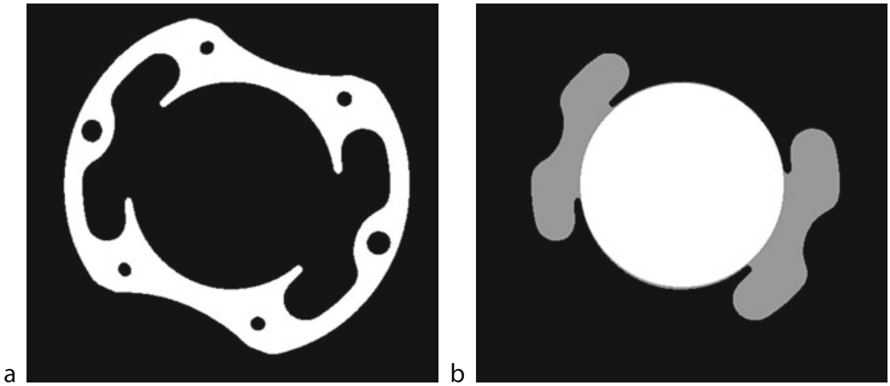


Fig. F.8

Largest internal circle. (a) Original object: metal stamping for the stator of an electric motor. (b) The largest circle (*white*) that can be placed within the hole. The centre of this circle indicates where an expanding gripper should be placed

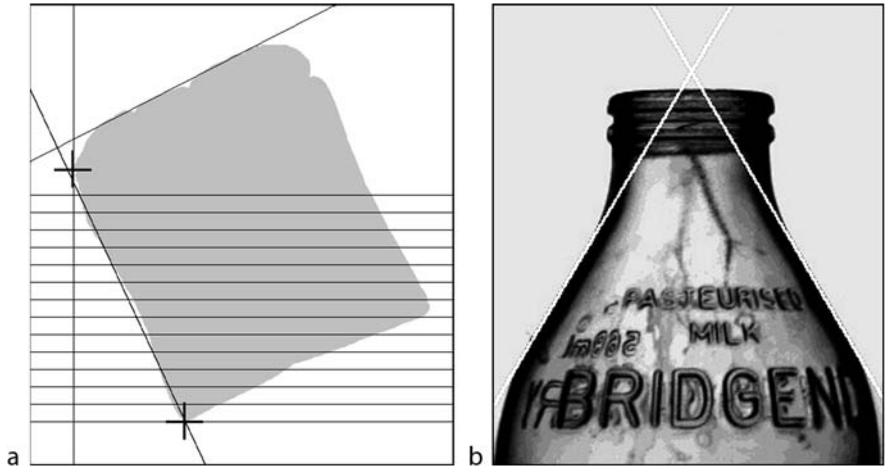
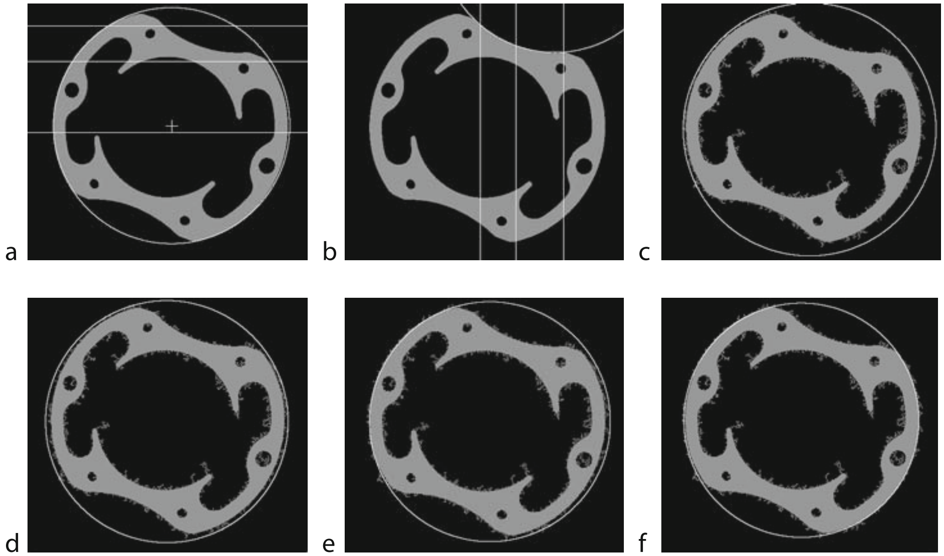


Fig. F.9

Fitting straight lines. (a) Slice of bread. Upright crosses indicate the top-left and left-bottom extreme points. These fixed the vertical positions for the horizontal scan lines, which in turn determine the data points for linear regression analysis. This enabled the line to be fitted to the left-hand side of the slice. In a similar way, a line was fitted to the top of the slice. The intersection of these two lines provides a useful datum point for further analysis. (b) Glass bottle. Two straight lines have been fitted. Again, their intersection provides a reference point for further measurement

Fitting straight lines This method is suitable for handling objects that have easily identifiable linear features. Two examples are shown in *Fig. F.9*. Since the slice of bread is nearly rectangular, the extreme points mentioned above provide useful cues for locating a set of horizontal scan lines. The points where they intersect the edge provide the data points for standard linear regression (least-squares) analysis. In the second example, two lines have been fitted to the edges of a bottle. Their intersection forms a useful datum point for inspection, location and measurement.

Fitting a circle, geometric method (*using three edge points*) In ➲ *Fig. F.7*, we illustrated one method for fitting a circle to an object silhouette using QT operator *mbc*. In that case, the whole object boundary was considered at first, although some edge pixels were subsequently "discarded." In the method demonstrated here and immediately following, only part of the edge is considered. To begin, *three* edge points are found. There are no general rules for doing this. Application knowledge should be exploited whenever possible and may make the choice of these points quite straightforward. As indicated above, simple mechanical-handling trickery is sometimes able to limit the angular deviation of the widgets from some known norm. This allows trivial edge-sampling algorithms to be used. In other cases, it will be necessary to employ a combination of smart heuristics and sophisticated image processing, to make sure that these three sample edge points are located appropriately. The position and radius of a circle that intersects these three points can then be found using three simple formulae, implemented in the QT operator *fcd*. The operator *fce* selects three edge points using horizontal scan lines



◘ Fig. F.10

Fitting a circle to the outer edge of a stamping for the stator of an electric motor. (a) Using three edge points. Three horizontal scan lines provide three intersection points on the left-hand-side of the stamping. These points are used by the QT command fcd to derive the parameters for the circle. Errors on the right-hand-side are due to quantisation noise. (b) Circle, fitted by mbc to three points found using vertical scan lines. (c) Noise added deliberately. Again fcd was used. The accuracy of fit is poor, particularly for those parts of the edge far away from the sample points. (d) Circle fitted to the noisy edge using the QT operator mbc. Since the circle encloses every white point, the fit is poor everywhere. Without the noise, mbc produces an excellent fit (not shown). (e) That part of the edge consisting of the convex arc on the left was used to fit a circle using fcl. The fit is good there, despite the noise. However, the fit on the right-hand side is poor. (f) Both convex arcs on the edge were used to fit the circle. The fit is good everywhere

placed in positions supplied to it and then applies *fcd*. ➲ *Figures F.10a* and *b* show examples of circles being fitted in this way. This method is well suited to situations where part of the edge of the widget is close to following a perfect circular arc and can be located easily. However, since it relies on only three edge pixels and is therefore sensitive to noise. Hence, even small dirt particles around the edge may cause the circle to fit badly (➲ *Fig. F.10c*). It is also susceptible to producing errors when the widget does not have perfectly circular arcs. For example, the outline of an apple or pizza would probably cause a serious misfit.

Fitting a circle: regression method (*least-squares algorithm*) A circle can be fitted to some or all edge pixels surrounding an object silhouette, using a least-squares algorithm. (QT command *fcl*) This may rely on all or just some of the edge pixels. In the latter case, the sample points may be scattered around the object perimeter or grouped together to form a circular arc. A good fit is less likely to be achieved in the latter case. Since this is a regression (least-squares) algorithm, a good fit can often be achieved in the presence of significant amounts of debris.

Fitting a polynomial curve The following MATLAB function uses a least-squares algorithm to fit a polynomial of order n to a set of m edge points, between the limits $y1 \leq y \leq y2$. Normally, we would expect $m >> n$.

```
function p = fit_polynomial(y1,y2, m, n)
% Fit a polynomial of order n (n ≥ 1) to a binary edge sampled at m pivotal points. [1/4]
global current
global alternate
original=current;                       % Save image for later
[x,y]=hzs(y1,y2, m);                    % Pivotal points
p=polyfit(y,x,n);                       % Calculate polynomial parameters
q=polyval(p,y);                         % Evaluate polynomial
zer                                     % Black image
for i=2:m
vpl(round(q(i)),round(y(i)), round(q(i-1)),round(y(i-1)),255)
                                        % Piece-wise linear plot
end
alternate = original;                   % Recover original image
subplot(1,2,1), imshow(current)         % Draw current image
subplot(1,2,2), imshow(alternate)       % Draw alternate image
```

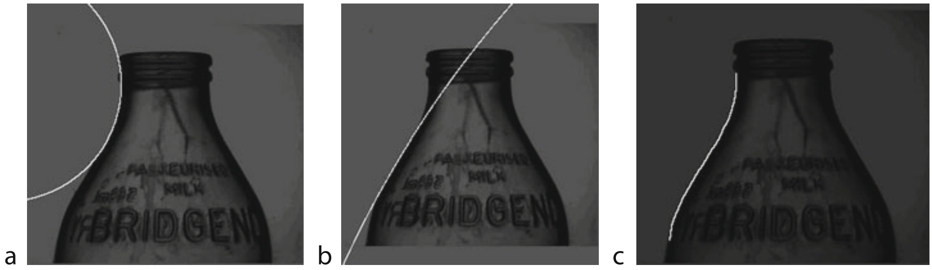


Fig. F.11

Fitting curves to the neck-shoulder region of a bottle. (a) Circle fitted to the neck. (b) Circle fitted to the shoulder. (c) 3rd order polynomial

Figure F.11 shows circles and a 3rd-order polynomial fitted to the neck-shoulder region of a glass bottle.

F.2.1 Using the Model

The *XY*-coordinates of the centre of the circle fitted by *mbc*, *fcd* or *fcl* often provide a way of defining the position of the widget. This is obviously so, where the fitted circle encloses the object, as in *Figs. F.7* and *F.10*. In those cases where the edge is locally concave (e.g., *Figs. F.10b* and *F.11a*), we may regard the fitted circle as a possible indicator for positioning one of the fingers of a robot gripper.

The polynomial found by *fit_polynomial* is single-valued and cannot describe a closed curve. Hence it cannot be used on its own to find the position of a widget. This function finds the local trend of the direction of the edge, ignoring minor deviations. Differentiating the polynomial enables us to estimate the possible slippage of a gripper that relies on friction, rather than widget geometry, to maintain a secure grasp. The polynomial might be used to calculate intersection points with other curves (lines, circles or polynomials), in a similar way to that shown in *Fig. F.9*. It is probably not a good idea to use the polynomial to extrapolate far outside the range of its input data.

F.3 Finding Object Orientation

F.3.1 From Object Silhouettes

There are numerous ways to find the orientation of an object. Here are just a few. The most appropriate method can only be chosen after studying the application requirements in detail.

1. Principal Axis One of the most popular ways to determine the orientation of an object is to use the principal axis (lmi). This is very effective if the object is long and thin, like the screwdriver in *Fig. F.12*. It is less effective if the object has no obvious "long axis," as judged by a human being. Calculating the principal axis is quite fast. It is important to note that, in some situations, it may be advantageous to calculate the principal axis on a lake or bay, rather than the object silhouette (*Fig. F.13*)
2. Bay and centroid (*Fig. F.14*)
3. Lake and centroid (*Fig. F.15*)
4. Centroid and furthest edge point (*Fig. F.16*)
5. Two bays (*Fig. F.17*)
6. Two lakes (*Fig. F.18*)
7. Lake and bay (*Fig. F.19*)
8. Two most distant edge points (*Fig. F.20*)

F.3.2 Complex Objects

Finding the orientation of a complex object has no clearly defined rules: each application is different and requires human intuition to design a robust algorithm. If there are just a few

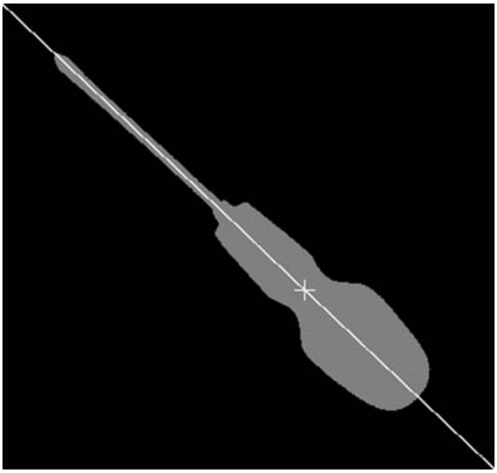


Fig. F.12
Principal axis and centroid

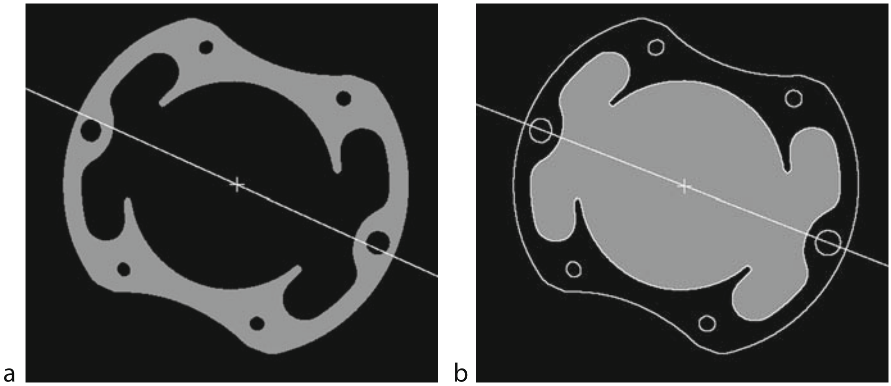


Fig. F.13
Principal axis and centroid, calculated on a lake. (a) Principal axis of the object silhouette. (b) Principal axis of the largest lake is more accurate, in this instance. [QT: enc, thr, blf, exr, big, angle = lmi]

clearly identifiable features, as in the central region of ➲ *Fig. F.21*, it may be possible to perform an initial calculation rapidly but which provides an ambiguous answer. With a little more cunning, it is often possible to resolve this easily. For example, in ➲ *Fig. F.21* simple pattern matching over a small carefully selected area will suffice. It is important to note however, that we have made two implicit assumptions:

1. The size of the object and the lens magnification are both known precisely.
2. Useful features of interest can be resolved accurately in ➲ *Fig. F.21*, if we consider only the central portion of the image. The complicated periphery of this object is simply ignored

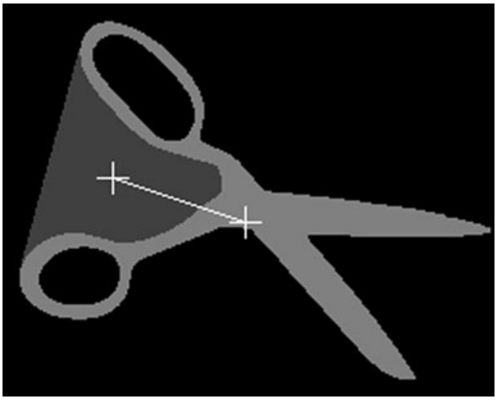


Fig. F.14

Orientation found from the line joining the object centroid and the centroid of the largest bay. QT command sequence: [[a,b] = cgr; chu, exr, big, [c,d] = cgr; angle = 180*atan2((a-c),(b-d))/pi]

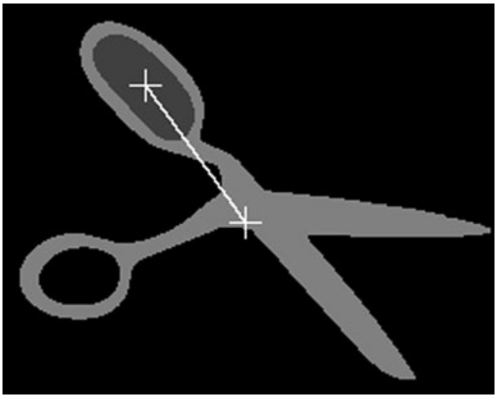


Fig. F.15

Orientation found from the line joining the object centroid and the centroid of the largest lake. QT command sequence: [[a,b] = cgr; blf, exr, big, [c,d] = cgr; angle = 180*atan2((a-c),(b-d))/pi]

during the initial stages. To resolve the ambiguity of orientation, we must perform additional tests. For example, we might devise an algorithm that detects the dark hole on the scanning circle at 2 o'clock.

If we exercise some ingenuity, there are many more "tricks" that we can perform! We cannot describe them all; our objective here is simply to prompt the reader to think of new ideas for him/herself. Consider *Fig. F.22*. The details of the processing that resulted in *Fig. F.22b* need not concern us here. The important point is that a number of significant

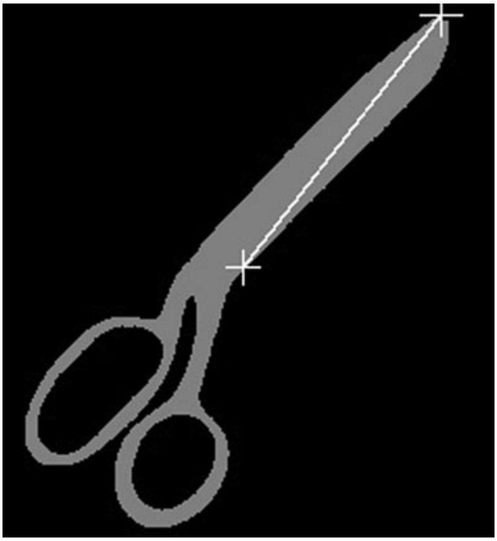


Fig. F.16
Orientation found from the line joining the object centroid and the most distant edge point. QT command sequence: [[a,b] = cgr; dfc,[c,d] = pki, 180*atan2((a-c),(b-d))/pi]

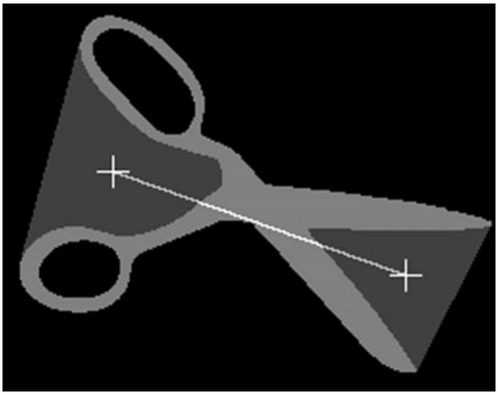


Fig. F.17
Orientation found from the line joining the centroids of two bays. QT command sequence: [chu, exr, big(3), [a,b] = cgr; swi, big(4), [c,d] = cgr; 180* atan2((a-c),(b-d))/pi]

features have been identified and measured. In this example, each feature is represented simply by its *XY*-position and area. The result is the following list:

$$L = [[122, 105, 8341], [122, 192, 818], [45, 71, 619], [45, 138, 609], [199, 139, 603],$$
$$[199, 71, 601], [46, 27, 159], [46, 179, 114], [196, 28, 111], [196, 180, 103]]$$

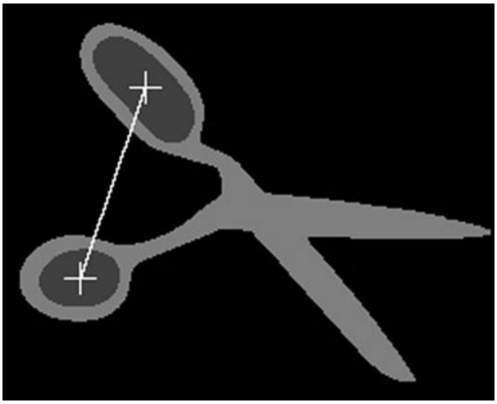


Fig. F.18
Orientation found from the line joining the centroids of two lakes. QT command sequence: [blf, exr, big,[a,b] = cgr; swi, big(2), [c,d] = cgr; 180*atan2((a-c),(b-d))/pi]

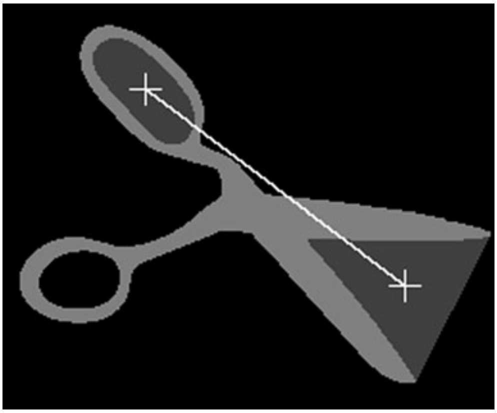


Fig. F.19
Orientation found from the line joining the centroids of one lake and one bay. QT command sequence: [wri, chu,exr, big(4),[a,b] = cgr; rei, blf, exr, big(1), [c,d] = cgr; 180*atan2((a-c),(b-d))/pi]

The format is as follows:

$$[[X1, Y1, \text{Area1}], [X2, Y2, \text{Area2}], [X3, Y2, \text{Area3}],]$$

This is an example of what we shall refer to as a *Star Map*. The position and a brief description is given for each feature, which is thereafter regarded as existing at a single point.

In some applications, the order in which the descriptions of individual feature are added to the star map list is not necessarily fixed or known; it is accidental that, for *Fig. F.22*, the large central well is described by the head of *L* (i.e., first sub-list). This can easily be identified from the list *L*, since the well is much larger than the nine peripheral holes. The fact that there is a single very distinctive feature makes the matching of pairs of lists much faster. In other

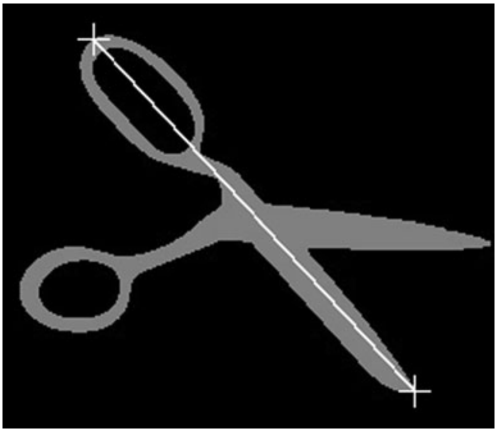


Fig. F.20
Orientation found from the line joining the two most distant edge points. QT command mdp

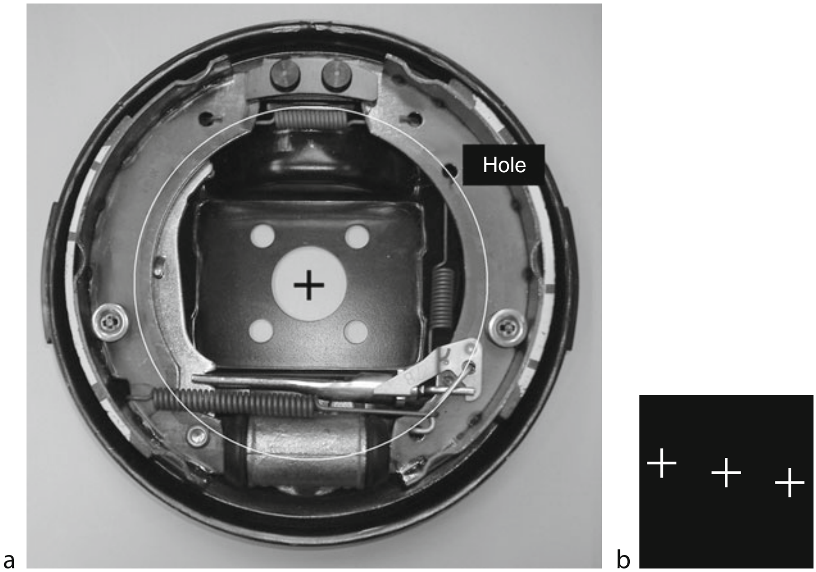


Fig. F.21
Finding the orientation of a complex object (automobile brake assembly) can sometimes be solved by the application of several simple heuristics. (a) The centroid of the central hole is shown as a black cross. The white circle is positioned at the centroid and its radius chosen to intersect the dark hole. By scanning around this circle, the orientation can be determined but with some ambiguity as the circle intersects other dark regions. (b) 3-tuple operator. This is then fitted to the original image by rotating it around the centroid of the central hole. When all three points rest over bright pixels, we signal a possible ''hit.'' Clearly, there are only two such hits as the 3-tuple rotates through 180°. Again, this will ambiguously determine the orientation but this can be resolved easily by referring to the scanning circle in (a)

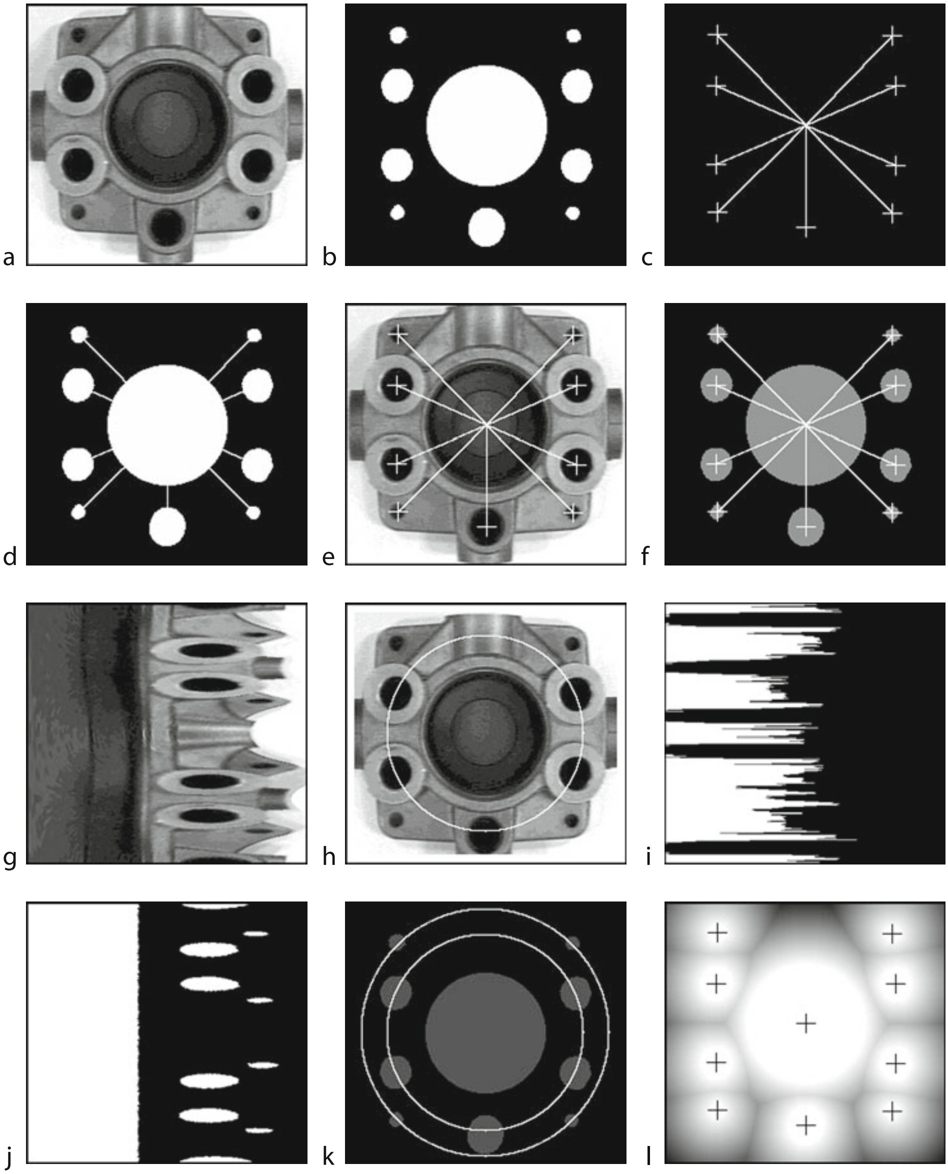


Fig. F.22

A summary of several ideas for determining the orientation of a complex object: an alloy casting for an hydraulics manifold. (a) Original image. (b) Ten features identified: central well and nine peripheral holes. (c) Centroids, joined to create a single "spider." (The crosses are included merely for clarity of illustration.) (d) Object, created by ORing images (b) and (c). It is possible to determine the orientation approximately by finding the principal axis of this object. (QT command lmi) (e) The body and feet of the spider define a 10-tuple filter, whose output is minimal when the spider is in/close to this position. (f) The feet and body of the spider define a binary erosion operator. The output is white only when the spider is in/close to this position.

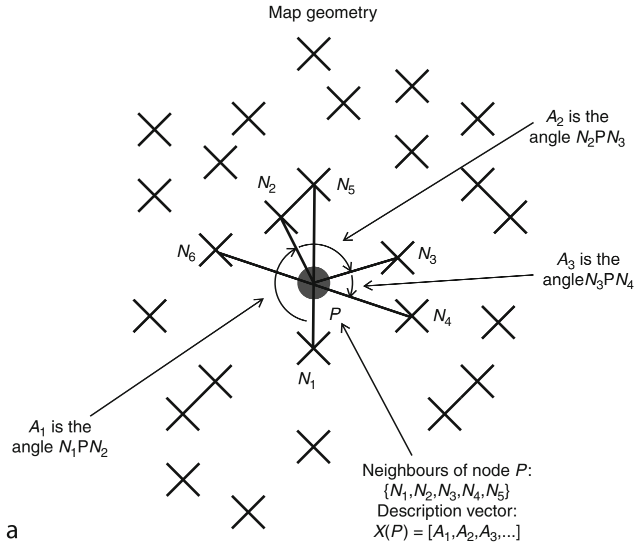
applications, different image features might be more appropriate. There is no restriction on the number of feature measurements that we include in each sub-list. A bare printed circuit board is well suited to this type of representation. The features that we use in this case might include component lead holes, pads (for attaching integrated circuits, connectors etc.), conductor track bifurcations and corners. If, as in *Fig. F.22a*, there is a single distinctive feature that can provide a well-defined reference point, the feature-measurement list might also include angular positions and distances from it. Several ideas for finding the orientation of the hydraulics manifold casting are summarised in *Fig. F.22*.

Suppose the we have two lists: L_{test} derived from a test sample and the L_{ref} from a reference "golden" sample. It is possible to match L_{test} and L_{ref}, using a variety of algorithms. Prolog is well suited for this type of task, since it involves pair-wise matching of the elements of two lists. *Figure F.23* shows how natural this type of task is in Prolog. On the other hand, matching lists is not natural for QT, or MATLAB. List matching is made easier if we can guarantee that L_{test} and L_{ref} have the same number of members; the same features are always detected. This is not an essential requirement but the computation is faster if this condition is met. Suppose that L_{test} and L_{ref} both have N members, Then, the time to perform the matching rises as $N!$. This would probably not present insurmountable problems for modest values of N (say $N \leq 10$) but would, for example, be problematical if we tried to match two large printed circuit boards by comparing lists describing hundreds of component-lead holes. We can reduce the computational work load considerably, if we sort the lists in some way first. For example, in *Fig. F.22*, there are three distinct types of features: the large central well and two different sizes of holes. We can make a significant saving in computation time if we sort L_{test} and L_{ref} in order of the areas of the blobs. Restricting the orientations of the widgets that L_{test} and L_{ref} represent to within a few degrees would also make a large difference. Even the simple observation that the central well in *Fig. F.22* is much larger than all of the others helps a lot, since we can use its centroid as a "pivot" during the matching process.

F.3.3 Radon Transform

Suppose that the object to handled by the robot has no obvious structure in its outline but has a grid-like pattern on its top surface that it is visible to an overhead camera. The orientation of the grid can be determined using either the Radon or Hough transforms. We shall demonstrate the use of the former (*Fig. F.24*). Both of these transforms generate an image in which bright spots indicate both the orientation and position of large sets of white pixels in the input image

(g) Polar-to-Cartesian axis conversion (ptc), centred at the middle of the well. (h) Circular scan, centred at the middle of the well and applied to the original image (i) Intensity profile around the circular path defined in (h). A similar result can be obtained, albeit more slowly, by following a vertical linear path in (g), just to the right of centre. The five large peripheral holes are clearly evident from this graph. (j) Polar-to-Cartesian axis conversion (ptc), applied to the binary image in (b). (k) Two circular scans, centred at the middle of the well, applied to the binary image in (b). These yield binary signatures which allow the orientation to be determined simply. (l) The grey background is the grass-fire transform (gft) of an image that contains just ten white pixels: the centroids. The 10-tuple spider will produce a maximum output when it is in the position shown here


Map geometry
A_2 is the angle N_2PN_3
N_2
N_5
N_6
N_3
A_3 is the angleN_3PN_4
P
N_4
N_1
A_1 is the angle N_1PN_2
Neighbours of node P: $\{N_1,N_2,N_3,N_4,N_5\}$
Description vector: $X(P) = [A_1,A_2,A_3,...]$
a

Notation: describing star maps

(i) S_{test} is the set of test map nodes

(ii) S_{ref} is the set of reference map nodes

(iii) $P \in S_{\text{test}}$ (P is a node in the test map.)

(iv) $Q \in S_{\text{ref}}$ (Q is a node in the reference map.)

(v) N_i is the node that is the ith nearest neighbour to P

(vi) A_i is the angle N_iPN_{i+1}. All angles are measured in the same direction (clockwise here.)

(vii) The point P is represented by $X(P) = [A_1,A_2,A_3,...]$, where A_i is the angle N_iPN_{i+1}

(viii) Each point in the test map is described in this way. The result is a set of vectors

$$\{X(P)|P \in S_{\text{test}}\} \qquad \text{... [F.1]}$$

(ix) Similarly, let $\{Y(P) = [B_1,B_2,B_3,...]|Q \in S_{\text{ref}}\}$ represent the set of reference map nodes

(x) To compare two node $P \in S_{\text{test}}$ and $Q \in S_{\text{ref}}$, compute $D_{P,Q}$ where

$$D_{P,Q} = \sum_i (A_i - B_i)^2 \qquad \text{... [F.2]}$$

(xi) To match two nodes $P \in S_{\text{test}}$ and $Q \in S_{\text{ref}}$,, an upper limit is set on $D_{P,Q}$

$$D_{P,Q} \leq \theta \qquad \text{... [F.3]}$$

b where θ(*theta* in the Prolog program) is a positive constant that we must select for each application.

Prolog program to match node pairs

```
match_star_maps(Test,Ref,L):-                          % Makes program easier to use
          match_star_maps(Test,Ref,Test,Ref,[],L).

match_star_maps(_,_,[],[],L,L).                        % Terminate recursion

match_star_maps(Test,Ref,T1,R1,L1,L2) :-
          member(P,Test),                              % Choose a node, P, from test star map, Test
          member(Q,Ref),                               % Choose a node, Q, from test star map, Ref
          get_parameter(theta,Theta),                  % Allows user to adjust theta easily
          match_nodes(P,Q,Theta),                      % Based on [F.1] and  [F.2]
          cull(P,T1,T2),                               % Match found, so remove P from T1
          cull(Q,T2,R2),                               % Match found, so remove Q from R1
          match_star_maps(Test,Ref,T2,R2,[[P,Q]\L1],L2). % Repeat with depleted lists
```

c

Fig. F.23

Coding and matching star maps with Prolog implementing a naive search algorithm

that are aligned in a linear manner. By selecting the brightest of these spots, it is possible to find the orientation of the most prominent linear feature in the input image. The white pixels contributing to the same bright spot in the transform image do not necessarily lie within a single connected blob. The QT command sequence used to generate ➋ *Fig. F.24c* was

$$\text{wri}('\text{grid}'), \text{enc}, \text{thr}, \text{ero}, \text{ero}, \text{ero}, \text{rdn}, \text{enc}, [x,y,t] = \text{pki}; \text{rei}('\text{grid}'), \text{tur}(-x)$$

F.3.4 Correlation

None of the techniques that we have described so far can calculate the parameters necessary to guide a robot to pick up all six objects in ➋ *Fig. F.1*. We cannot leave this topic without describing how this can be done. We shall describe three methods, each based on correlation.

Edge distance profile. Suppose we have already found the centroid. The orientation can be found using the radius (R) expressed as a function of the distance travelled around the perimeter of the object (➋ *Fig. F.25*). A simple approximation to this is obtained by the following formula:

$$D = \sum \left(N_{\text{odd}} + \sqrt{2} N_{\text{even}} \right)$$

Here, N_{odd} is the number of odd-numbered chain-code steps and N_{even} is the number of even-numbered chain-code steps. Notice that $R(D)$ is a periodic, single-valued function (➋ *Fig. F.25*). This graph changes very little when the object is rotated, except that the phase is shifted (Small variations occur due to quantisation effects). Hence, the corresponding curve from a reference object can be fitted using one-dimensional correlation techniques. As a result, this is a fast method for finding object orientation.

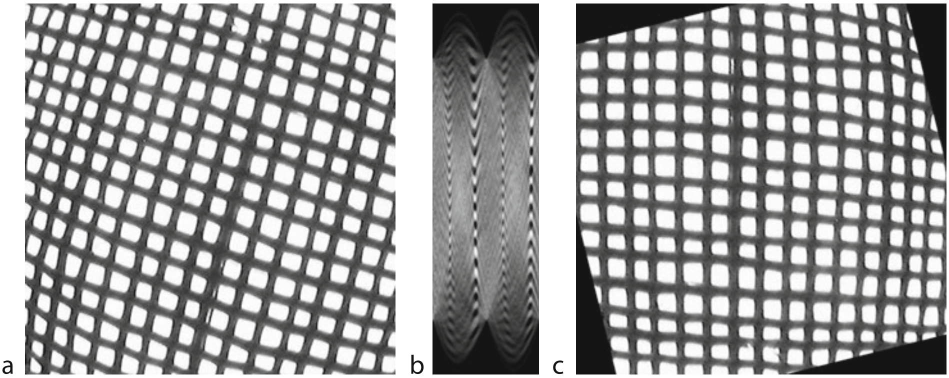


Fig. F.24

Using the Radon transform to determine the orientation of a grid. (a) Original image. (b) Radon transform image. Abscissa, angle. Ordinate, Distance from the centre of the image. (b) The brightest pixel in the RT image was found and its X-coordinate was used to determine the orientation (α) of the grid. The original image was then rotated by an amount -α. Notice that although the grid is not perfectly rectangular, the computation has resulted in an accurate orientation of the vertical wires

Correlation on a polar-coordinate plot (➜ *Fig. F.26b*). The Cartesian-to-polar coordinate-axis transformation (QT command *ctp*) is performed first. This requires a reference point. (The centroid is often most convenient but any other well-defined feature will suffice.) The result is correlated with a similar image derived from a reference object. It is clear in ➜ *Fig. F.26c* that rotating the object by even a very small amount results in a large change in the value of the auto-correlation coefficient. This indicates that this could form the basis for fixing the orientation compared to that of a reference object.

(c) *Correlation in three dimensions.* We have just seen how correlation can be used to fix orientation. We used the centroid to fix the orientation first. It is technically feasible to perform correlation in three dimensions. This could follow the route already established, or it could employ a hill-climbing process, in the hope of improving the speed of processing. However,

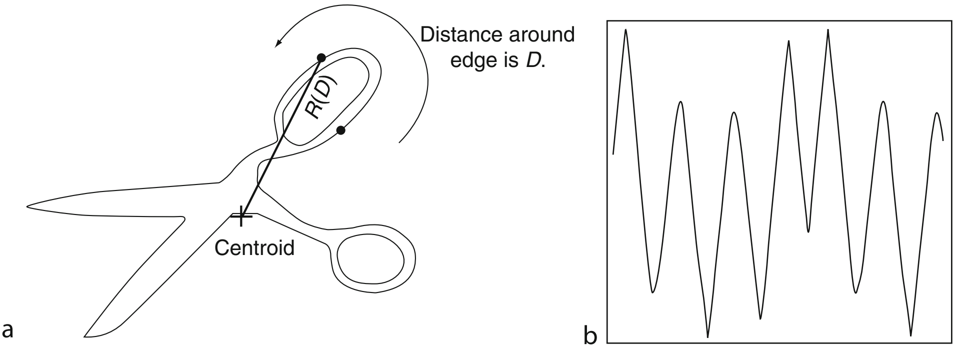


Fig. F.25

Radius plotted as a function of the estimated distance around the perimeter. (a) How the coding is performed. (QT command rdc.) (b) Results for the scissors in ➜ *Fig. F.27a*. Abscissa: D. Ordinate: R(D)

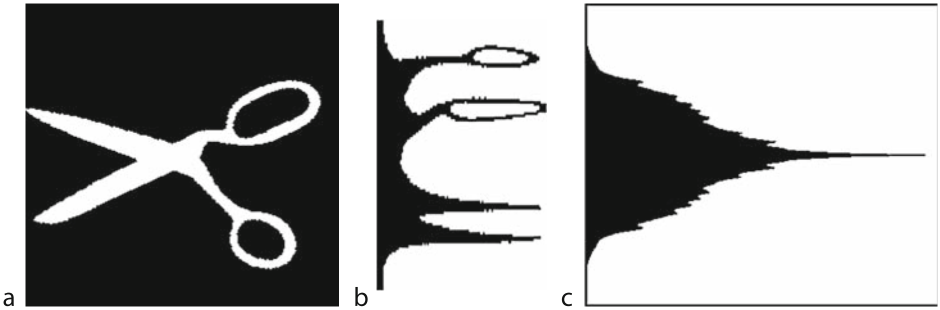


Fig. F.26

Correlation on a polar-coordinate plot. (a) Original image. (b) Cartesian-to-polar coordinate-axis transformation. (c) Two-dimensional autocorrelation function was applied to (b). The intensity profile along a vertical line through the centre of the auto-correlogram shown here has a very sharp peak. This indicates that cross-correlation between polar-coordinate plots, derived from the test and the reference objects, is able to fix the orientation precisely

correlation in three dimensions is not normally recommended, since it is unlikely to be fast enough for use in practice and the process is sensitive to minor changes in shape. For the sake of completeness, we include *Fig. F.27*, which shows that, if the orientation is fixed, it is a straightforward matter to use correlation to fix the *XY*-coordinates.

F.4 Finding Object Chirality

Consider *Fig. F.28*. However either of these objects is rotated or translated in the plane defined by this page, it is impossible to fit one onto the other. It is necessary to flip it over to do so. They are said to differ in their *chirality*. (This term borrowed from chemistry where it refers to pairs of molecules that are structurally identical but are mirror images of one another. The most infamous pair of molecules of this kind is thalidomide.) Many measurements are the same for these two objects, so how can we distinguish them? The numbers in *Fig. F.28* indicate the relative sizes (areas) of the concavities (This includes both lakes and bays.) Now, starting with the concavity with the largest area, we list the order in which we encounter the others, as we travel clockwise. For the object on the left, the sequence is:

$$S_1 = [1, 6, 2, 5, 3, 4] (\text{Object on the right})$$

and for the one on the right it is

$$S_2 = [1, 4, 3, 5, 2, 6] (\text{Object on the right})$$

Notice that these lists are linear representations of cyclic sequences. Hence, S_1 is the reverse of S_2. However these two objects are rotated or translated within the same plane, the lists S_1 and S_2 will remain the same. Hence, we can correlate the list generated from a test object with that from a reference object, to determine the chirality.

This approach is not totally satisfactory, since pairs of concavities may have very similar areas. For example, let us suppose for the moment that the finger holes in the scissors are identical. We might then find that S_1 *is* either [1,6,2,5,3,4] or [1,5,2,6,3,4], while S_2 is [1,4,3,5,2,6] or [1,4,3,6,2,5]. Determining the chirality is still possible but it just takes a little longer as two reference lists have to be matched to that from the test object. Problems are worse if there are two concavities with equal areas that are bigger than all others. Another difficult situation occurs when the concavities all have equal areas. It is then impossible to use this approach to determine chirality. Notice that this condition does not imply that concavities with equal areas have the same shapes; it might still be possible to modify the chirality test to compare meta-concavities, or even meta-meta-concavities. There are many other chirality tests that we can devise. For example, we might compare only bays, or only lakes, to increase processing speed. Alternatively, we might compare such features as the number of meta-bays, aspect ratio, or shape factor (Area/Perimeter^2). It is easiest to devise chirality tests around a single numeric metric. Deciding which one to use can only be done after careful consideration of the application.

F.5 Challenges

In common business parlance, the word "*challenge*" is a euphemism for "*very difficult*" but this is not the intended meaning here. We have described many different ways to determine position and orientation. However, there are numerous other methods that we could employ. *Figure F.29* illustrates a variety of applications, some of which require methods not yet covered

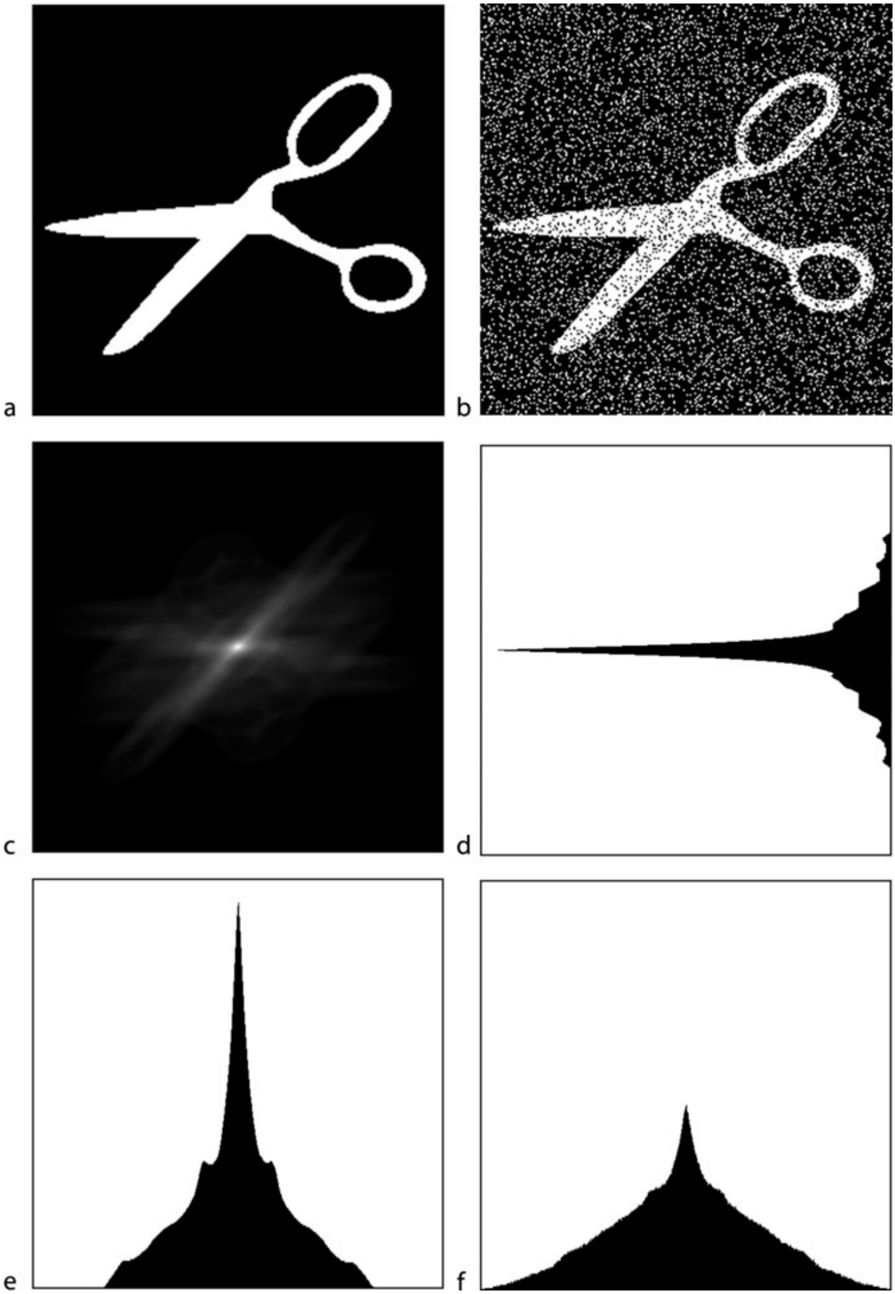


Fig. F.27

Correlation in two dimensions: *XY*-translation, with the orientation correctly fixed. (a) original image. (b) Salt-and-pepper noise added. QT: spn(0.25) (c) two-dimensional auto-correlogram of (a). (d) Intensity profile (vertical section) through the centre of (c). (e) Intensity profile (horizontal section) through the centre of (c). (f) As (e) but computed on the noisy image, (b)

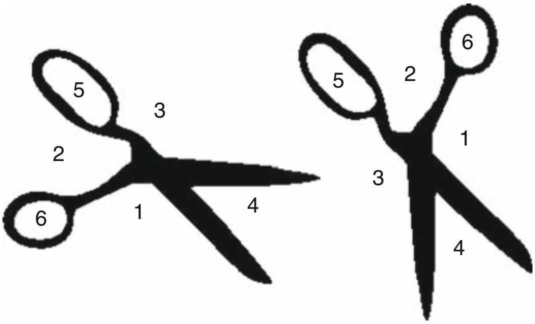


Fig. F.28

Two objects that differ in their chirality. However either of them is moved or rotated, it is impossible to fit it onto the other. It is necessary to turn it over, to do so. The numbers indicate the relative sizes (areas) of the concavities (lake and bays)

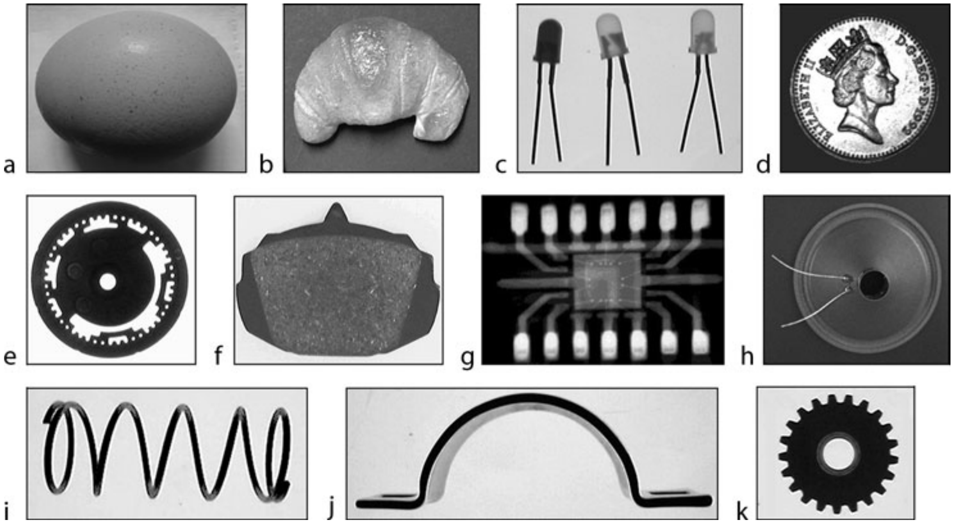


Fig. F.29

Challenges. (a) Egg (b) Croissant (c) LEDs (d) Coin (e) Cam (f) Brake pad (g) IC x-ray (h) Wires (i) Spring (j) Semi-toroidal mirror (viewed edge on) (k) Gear

in this appendix. They are all soluble using quite simple QT command sequences, although there is no prescription that we can offer that will solve them all. There is one common factor that all of these applications and those illustrated earlier share: they all require that magic ingredient: *human intelligence* (*Fig. F.1*).

Appendix G: Image Catalogue

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_48,

G.1 Images Accessible Via QT

QT is supplied with a range of images, stored in the folder *BGB_MatLab_Images*. Some of these images form the basis for illustrations found elsewhere in this book. ➜ *Table G.1* describes a selection of these images and gives their corresponding file numbers. (Other images are provided but are not catalogued below, to conserve space.) In order to read the monochrome *Coffee beans* image, use the QT command *rni(4)*. To read the colour *Apples* image, type *rci(294)*. Notice that the folder *BGB_MatLab_Images* must be placed on MATLAB's search path first. A pictorial index is provided below.

Most of the images have direct industrial relevance. However others are included, simply because they model important industrial vision applications. For example, the three *Scissors* images *(no. 48– 50)* exemplifies articulated objects that have a single hinge. *Child's building blocks* (no. 306) resembles printed packing and is a useful model for studying colour recognition. In some applications, the requirements may be surprisingly simple; in images 307, 460, 462, 762, for example, monitoring the relative proportions of mixtures of colours is all that is needed. (How many times have you been frustrated to find that your favourite colour is not available in a packet of Liquorice Allsorts?) In such cases, any suitable analogous task that presents the same important properties as a real industrial application will suffice for learning purposes. For an author, of course, it circumvents copyright restrictions.

Using QT and the images supplied on the DVD, you will be able to repeat many of the experiments reported in this book. Additional test images can be drawn, by hand, generated using a graphics program (e.g. *Adobe Illustrator*™, *Photoshop*™), or using QT's limited set of image-generation functions. (*cir, dic, vpl, wgx*, etc.) For educational and training purposes, images can be captured using a standard digital camera and simple lighting, such as an LED torch. In this case, it is important to realise that, even if the whole image is not appropriately lit, parts of it may suffice for an exploratory study. (Image 28, *Creased fabric*, is an example.) Using ordinary everyday objects as subjects for learning QT and the principles of lighting is both rewarding and fun. Go to the pantry, medicine cabinet, tool-box, children's toy box, etc. for inspiration. Then, follow four simple steps to choose the lighting and viewing conditions:

1. Experiment
2. Observe
3. Adjust
4. Go to step 1

Table G.1

Some of the images accessible from QT

File Number	Object/scene	Lighting-viewing Method	Typical inspection task (e.g. detect/locate/inspect/measure)
0	QT Logo	N/A	N/A
2	Metal mesh	Back	Orientation of mesh
3	Grains of rice	Front	Length and width
4	Coffee beans	Front	Analyse texture to determine type
17	Electrical connector	Back	Faulty assembly
18	Loudspeaker cone with loose wires	Front diffuse	Ends of wires and orientation
19	Coil with loose wires	Back	Ends of wires and find orientation
20	Internal screw thread	Conical mirror	Integrity of thread
21 – 24	Coin (UK, 20 pence)	Diffuse front (21 and 23)/360° grazing (22 and24)	Lettering and monarch’s head. Flat surface for pits and scratches.
28	Creased fabric	Grazing illumination	Susceptibility to creasing
31	Glass bottle with crack	Back	Detect cracks, measure profile
32	Glass vial (brown)	Dark-field	Profile and dimensions
33	Glass bottle	Crossed linear polarisers	Stress in side walls
34	Embossing on glass bottle	Back	Complete and properly formed
37	Hydraulics manifold	Front	Overall inspection
	(plan view)		Holes for debris
38	Hydraulics manifold	Front	Complete inspection of each feature
	(side view)		including checking holes for debris
40	Brake pads (two samples)	Front	Check integrity of assembly.
41	Ferrous automobile with crack	Dye penetrant UV fluoroscopy (Low-level VIS lighting reduces contrast)	Detect cracks
43	Screwdriver	Poor backlighting	Find position and orientation
44	Formal table place setting	N/A – binary image	Object recognition and layout
45	Slice of bread	N/A – binary image	Sides parallel.
			Overspill.
			Rounded top.
			Overall symmetry
46	Four piece parts	N/A – binary image	Guide robot for pick and place
			Position, orientation, chirality
			Sufficient space for gripper
47	Conrod	N/A – binary image	Position, orientation, chirality

Table G.1 (continued)

File Number	Object/scene	Lighting-viewing Method	Typical inspection task (e.g. detect/locate/inspect/measure)
48 – 50	Scissors	N/A – binary image	Gripping points
			Position, orientation, chirality
			Angle of opening.
52	Leather glove parts	N/A – binary image	Sort. Check chirality
53	Plant	Back lighting	Dissect plant for micropropagation
54	Cake decoration pattern	N/A – binary image	Aesthetic appearance (broad scale)
55	Cake decoration pattern	N/A – binary image	Aesthetic appearance (broad scale)
57	Outline of Britain and Ireland	N/A – binary	Test image – binary image processing
73	Peppers	Front	Test image – colour recognition
74	Parrots	Front	Test image – colour recognition
91	Template for child’s toy	N/A – binary image	Simple model of robot assembly
			Locate shapes and template slots and parts
92	Ivy leaves (composite)	Front	Test image – shape analysis
93	Loudspeaker cone	Back	Length of end tube
99	Object in image 763	Line-scan camera Front	Orientation
108	Jar of baby food with contaminants	X-ray image	Foreign bodies
109	Biscuit (cookie)	Poor front lighting	Circularity and size.
			Count (visible) chocolate drops
111	Aerosol spray assembly	X-ray	Completeness of assembly
112	Power plug (13 A, UK standard)	X-ray	Safe wiring
113	DIL integrated circuit	X-ray	Bonding wires
114	Tooth brush	X-ray	Metal clips holding bristles in place
135	Printed circuit board	X-ray	Check conductors
136	Micro-circuit	Not known	Material deposition
142 and	Clock (two images, same	Exactly registered image	Telling time (Model of reading needle
143	camera position)	Diffuse front	instruments)
148	Clock	Diffuse front	Telling time
158	Metal grid	Back	Check holes not blocked
			Find orientation
159 and 160	Hand drawn lines	Diffuse front	Test images for Hough and Radon transforms
162	Shaft coding cam	Back	Find orientation

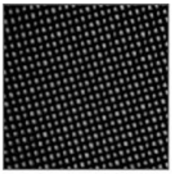
Table G.1 (continued)

File Number	Object/scene	Lighting-viewing Method	Typical inspection task (e.g. detect/locate/inspect/measure)
163	Rattan cane webbing	Diffuse front	Check weave
168	Coloured bricks	Diffuse front	Test image – colour recognition
169	Cam	Back	Orientation
171	Gear	Back	Shape
			Orientation
173	Zinc die casting	Back	Flash (extra material, moulding fault)
174	Metal stamping (stator for small electric motor)	Back	Shape
			Position and orientation
175	Plastic moulding	Back	Shape
179	Resistors	Diffuse front	Colour bands
208	Four objects on [XYθ]-table	Diffuse front	Robot vision: pick and place
210	Sprocket wheel	Diffuse front	Orientation and orientation
223	Croissants	Diffuse front	Colour
			Shape
225	Cookies	Diffuse front	Model of multi-component packing
239	Cane webbing	Diffuse front	Weave and find orientation
243	Car drum brake	Diffuse front	Detailed inspection of assembly
249	Dice	Diffuse front	Pick and place parts
251	Keyboard	Diffuse front	Completeness
			Correct key labels
264	Three girls	Daylight	Scene segmentation
266	Sweets	Diffuse front	Test image – colour recognition
270	Pencils	Diffuse front	Colour mix
294	Apples	Diffuse front	Grading
302	Coloured fabric	Diffuse front	Printing
303	Plastic container (simulated)	Diffuse front	Printing
305	Three shades of yellow	Diffuse front	Test image – colour recognition
306	Child’s building bricks	Diffuse front	Test image – colour recognition
307	Rubber mouldings, pencil erasers	Diffuse front	Colour mix
308	Component in plastic bag	Diffuse front	Read red printing
327	Peaches on tree	Diffuse front	Test image – colour recognition
344	Pizza	Front diffuse	Colour mix
354	Wool rug	Front diffuse	Test image – colour recognition

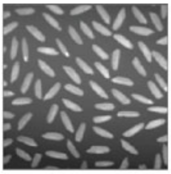
Table G.1 (continued)

File Number	Object/scene	Lighting-viewing Method	Typical inspection task (e.g. detect/locate/inspect/measure)
432	Electronic circuit	Microscope	Check integrity
433	Electronic circuit	Microscope	Check integrity
441	Hydraulics manifold	Omni-directional	Detailed inspection of all features
446	Quiche	Front diffuse	Colour mix
460	Sweets	Front diffuse	Test image – colour recognition
462	More sweets	Front diffuse	Test image – colour recognition
483	Coil with loose wires	Back	Robot to grasp wires
492	Uncut sapphires	Back	Size and shape
497	Plastic moulding	Dark field	Check moulding integrity
499	Printed circuit board	Front diffuse	Locate drill holes
500	Metal label	Front diffuse	Embossed numerals
526	Fluorescent printing	UV fluorescence	Authenticity
533	Plant	Back	Micropropagation
562	Printed text	Front diffuse	Optical character recognition
570	RFID sticker	Front diffuse	Check integrity
582	Hologram	6 different angles	Verify authenticity
583	Crown bottle top	Front diffuse	Integrity of seal
589	Tooth-brush	X-ray	Check integrity
591	Aerosol spray nozzle	X-ray	Check completeness of assembly
627	Loaf	Multiple light stripes	3D shape inspection
628	Aerosol spray cone	Dark field	Spray cone density analysis
629	Loaf	Multiple light stripes	3D shape inspection
740	Glass jar: Spike	Back	Critical defect
741 – 3	Bottle: Birdswing	Back	Critical defect
745 – 7	Bottle	Back	Shape
749	Bottle: Birdswing	Back	Critical defect
750	Glass vial	Back	Filling level
756	Glass jar: Spike	Dark field	Critical defect
757	Bottle: Birdswing	Dark field	Critical defect
758	Glass tumbler, filled	Striped background	Filling level
759	Glass jar, filled	Striped background	filling level
760	Glass jar, filled	Striped background	Filling level
762	Liquorice allsorts	Front diffuse	Colour mix
763	Alloy casting	Front	See images 99 and 763.
764	Catalytic converter	Front diffuse	Blocked holes
766	Dirty mirror	Grazing	Fingerprints, scrathes, pits, dirt
794	Object in image 763	Depth map – structured lighting	3D shape

0
QT logo

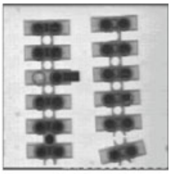


2
Metal grid

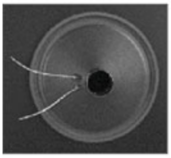


3
Rice grains

4
Coffee beans

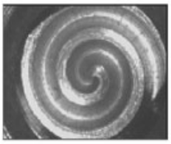


17
Electrical connector



18
Loose wires

19
Loose wires



20
Internal screw thread

21
Coin

22
Coin

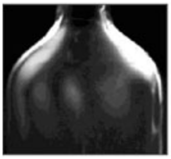
23
Coin

24
Coin

28
Creased fabric

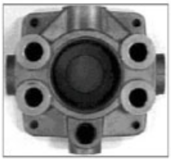
31
Cracked bottle

32
Glass vial

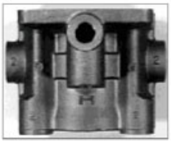


33
Stress in glass

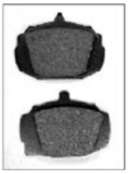
34
Embossed glass



37
Hydraulics manifold



38
Hydraulics manifold

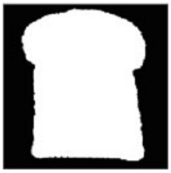


40
Brake pads

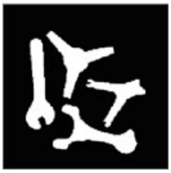
41
Cracked ferrous part

43
Screwdriver

44
Table place setting



45
Slice of bread



46
4 parts

Fig. G.1.1 (Continued)

47
Conrod

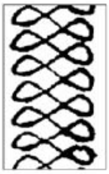
48
Scissors

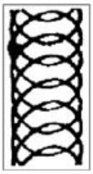
49
Scissors

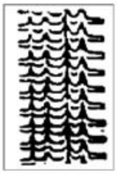
50
Scissors

52
Glove components

53
Plant


54
Cake decoration


55
Cake decoration


56
Cake decoration

57
Map of UK & Ireland

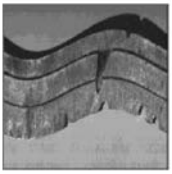
73
Peapers

74
Birds

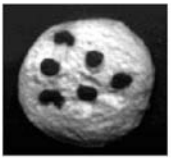
91
Child's toy

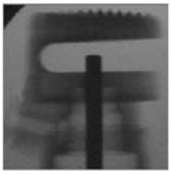
92
Ivy leaves

93
Loudspeaker cone

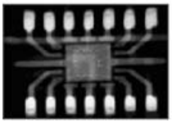

99
Line scan image

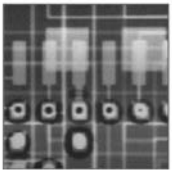
108
Jar of baby food

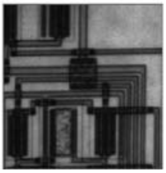

109
Cookie

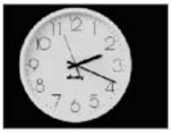

111
Aerosol spray

112
Power plug


113
DIL chip


135
PCB


136
Microcircuit


142
Clock

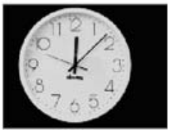
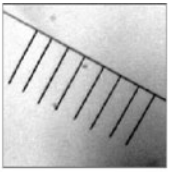

143
Clock

Fig. G.1.2 (Continued)

148
Clock

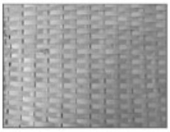
158
Metal grid



159
Hand-drawn lines

160
Hand-drawn lines

162
Timing cam



163
Rattan cane web

168
Child's building blocks

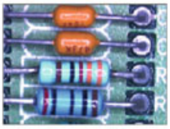
169
Cam

171
Gear

173
Die casting

174
Metal stamping

175
Plastic moildings

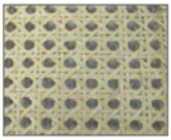


179
Resistors

208
[XYtheta] table

210
Sprocket

225
Yummy things to eat



239
Rattan cane web

243
Card drum brake

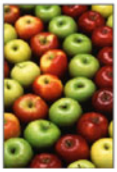
249
Dice

251
Keyboard

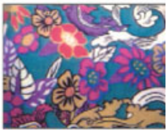
264
3 grils

266
Sweets

270
Colour pencils

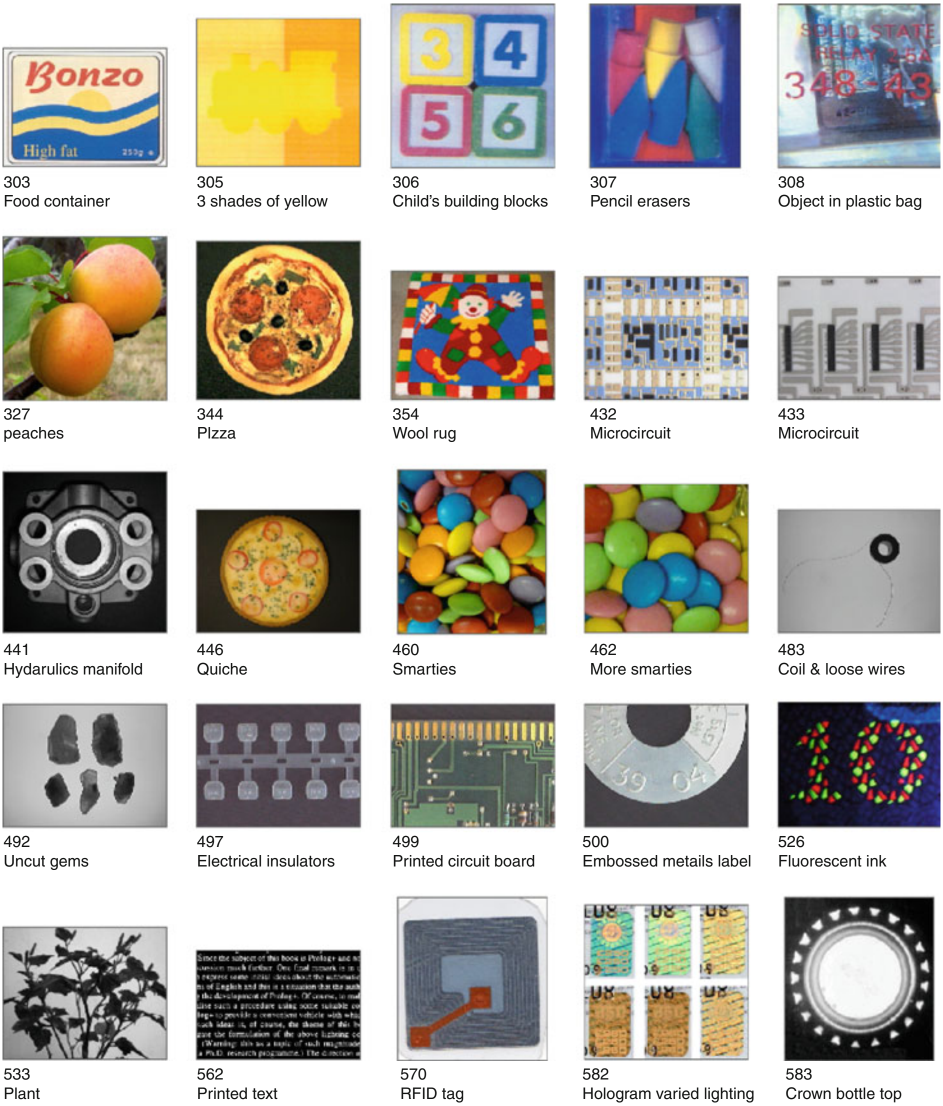


294
Apples



302
Dress fabric

Fig. G.1.3 (Continued)



303
Food container

305
3 shades of yellow

306
Child's building blocks

307
Pencil erasers

308
Object in plastic bag

327
peaches

344
Plzza

354
Wool rug

432
Microcircuit

433
Microcircuit

441
Hydarulics manifold

446
Quiche

460
Smarties

462
More smarties

483
Coil & loose wires

492
Uncut gems

497
Electrical insulators

499
Printed circuit board

500
Embossed metails label

526
Fluorescent ink

533
Plant

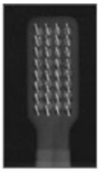
562
Printed text

570
RFID tag

582
Hologram varied lighting

583
Crown bottle top

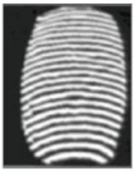
Fig. G.1.4 (Continued)


589
X-ray toothbrush

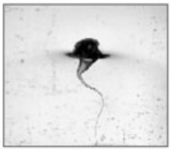
591
Aerosol spray

627
Light stripes on loaf

628
Aerosol spray


629
Light stripes on loaf

730
Road sign varied lighting


740
Spike

741
Birdswing

742
Birdswing

743
Birdswing

745
Glass bottle

747
Glass bottle

749
Birdswing

750
Filled vial

756
Spike

757
Birdswing

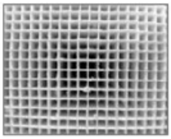
758
Filled tumbler

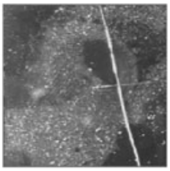
759
Filled jar

760
Filled jar

762
Liquorice allsorts

763
Alloy casting


764
Catalytic converter


766
Mirror

794
Depth map

Fig. G.1.5

Pictorial Guide to QT Images

G.2 Sample Infra-Red Images

The spectral response of a silicon photo-detector extends well into the infra-red waveband (➋ *Fig. G.2.1*), suggesting that a standard digital camera can be modified to detect NIR images, by simply removing the built-in IR-blocking filter. (Several "hobbyist" photography web sites show how this can be done.) Some cameras have a mechanical switch to move the filter out of the optical path. Several of the images shown below were obtained using a *Sony Cyber-shot DSC-H9* camera in "*Nightshot*" mode. This disables its internal IR-blocking filter. The author then fitted a VIS-blocking filter with a cut-of wavelength of 720 nm. (He purchased a filter of diameter 72 mm and held it in place with a 74–72 mm step-down converter ring, attached to the lens hood. It is not possible to fit the filter directly to the zoom lens.) This produces a camera that is sensitive only to NIR wavelengths: approximately 720–1,100 nm. Normally in "*Nightshot*" mode (i.e. without the additional filter), the camera senses both VIS and IR, which is also useful on occasions. Nearly all of the functions of the *Sony Cyber-shot DSC-H9* camera are available with this set-up. In particular, the focus and light meter still work as normal.

Of course, before experimentation can begin, an NIR light source is also needed. There are several low-cost options:

- Sunlight (It is difficult to find a lamp much cheaper than this!)
- Incandescent filament lamp
- Bathroom "heat lamp"
- IR LEDs. (The author bought a kit with 36 IR LEDs mounted on a PCB, for under €20. The power supply is a 12 V/0.5 A source.)
- IR flood-light, intended for surveillance CCTV systems

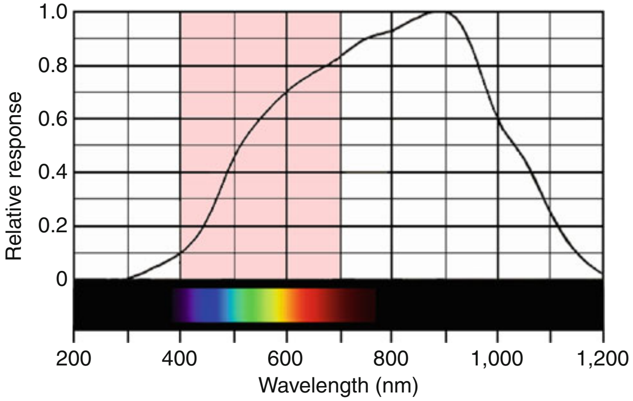


Fig. G.2.1

Typical spectral response of a silicon photo-detector. Notice that the device is more sensitive to IR than it is to VIS (*pink*). This is inconvenient for normal use, so a filter is usually placed in the optical path to attenuate IR. If this filter is removed, a Si-based camera can be made sensitive to both VIS and NIR. To block VIS light, a long-pass filter with a cut-off wavelength just over 700 nm is put in front of the sensor chip

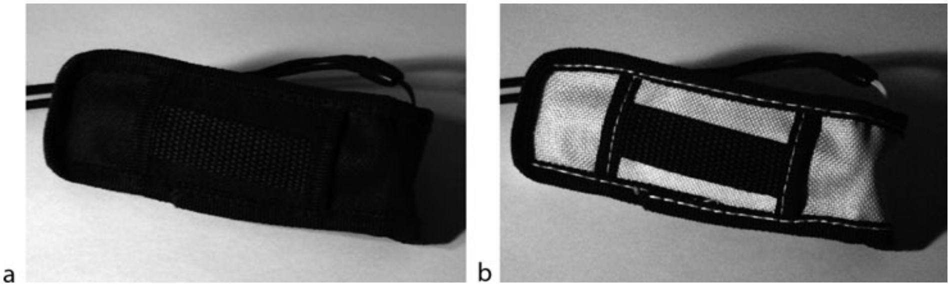


Fig. G.2.2

IR sometimes produces unexpected and spectacular results! This is a canvas case, about 120 mm long. (a) VIS image. (b) NIR image. Waveband approximately 720–1,100 nm (There has been no image processing to enhance the contrast)

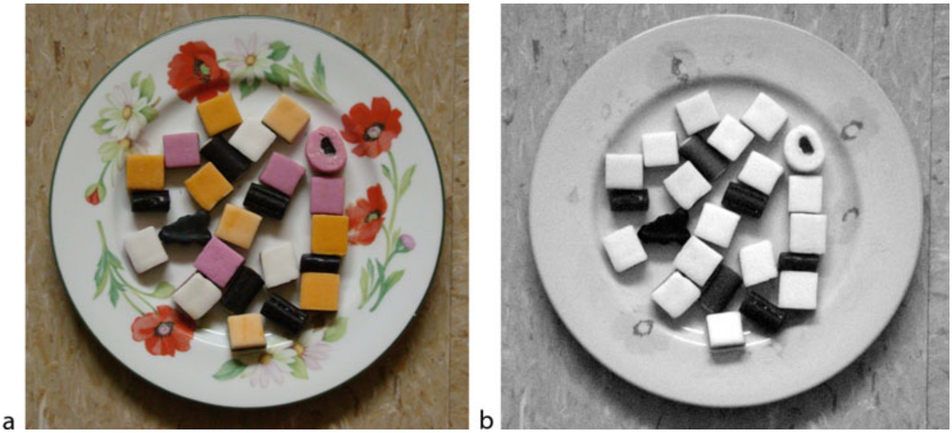


Fig. G.2.3

IR is often insensitive to colouring. Liquorice Allsorts (Yummy!) (a) VIS image. (b) NIR image. Waveband approximately 720–1,100 nm. Notice that the colouring has little effect on the NIR image and that the printed pattern on the plate almost disappears

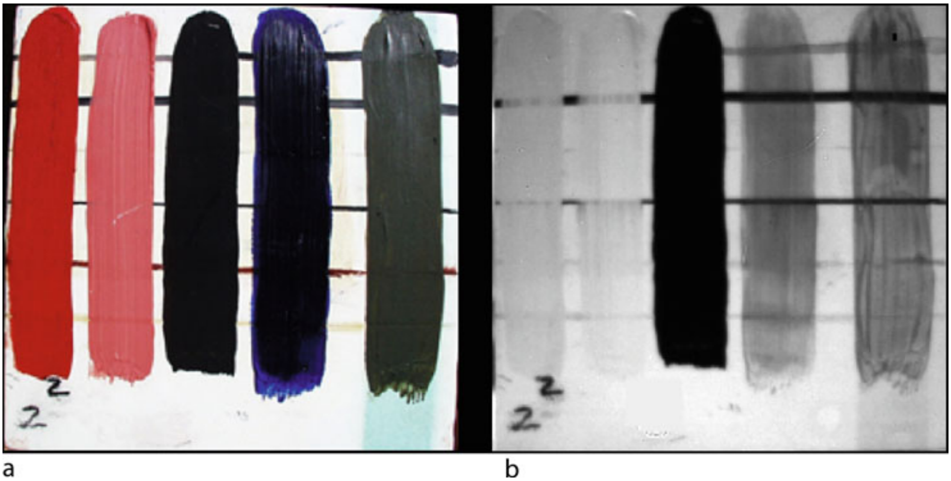


Fig. G.2.4
IR can sometimes see through paint. The ability to do so depends on the material content of the coating. (**a**) VIS image. (**b**) IR image. Waveband approximately 900–1,680 nm. Notice that the "black" (central stripe is nearly opaque and brush marks are enhanced in the right-most stripe). This image appeared in "*Alien Vision*" (From [5]. Reproduced with permission.)

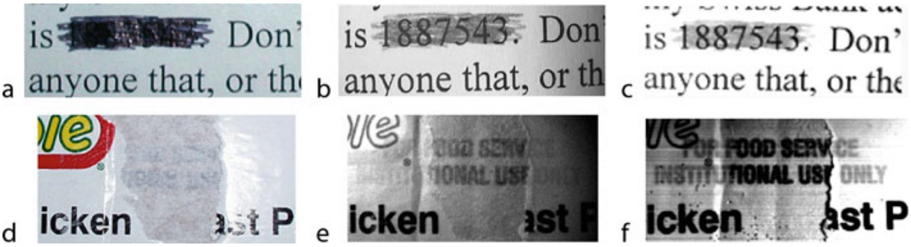


Fig. G.2.5
Another example of seeing hidden details using IR. (**a**) VIS image. Important detail is obscured by black ink from a ball-point pen. (**b**) NIR image. (**c**) SWIR image. (**d**) Another sample. VIS image. Part of the printing is obscured by a thin paper label. (**e**) NIR image. (**f**) SWIR image. (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/ Reproduced with permission)

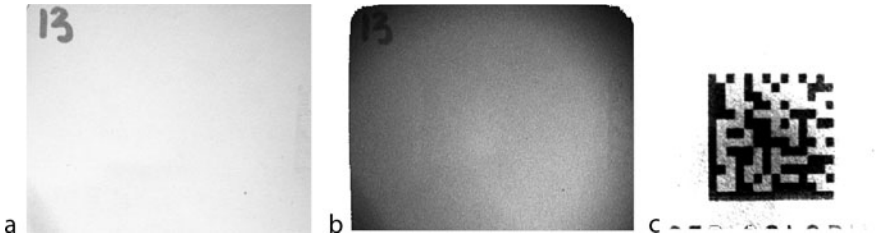


Fig. G.2.6

Another example. The ability to see through paint varies with wavelength. (a) VIS image. (b) Waveband: 840–889 nm. (c) Waveband: 900–1,680 nm. (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/ Reproduced with permission)

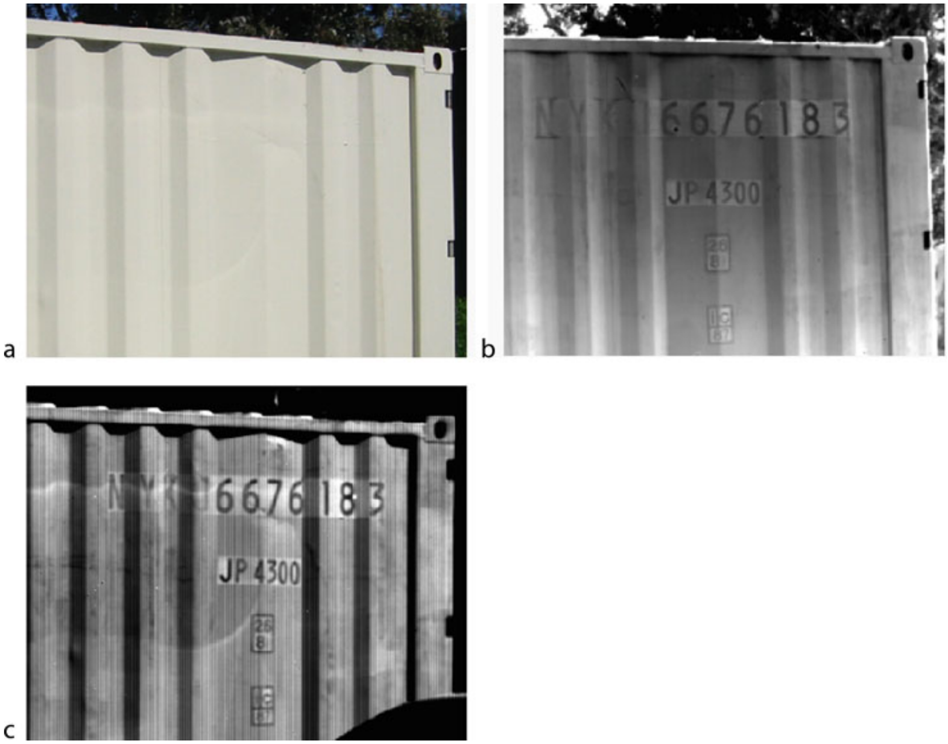
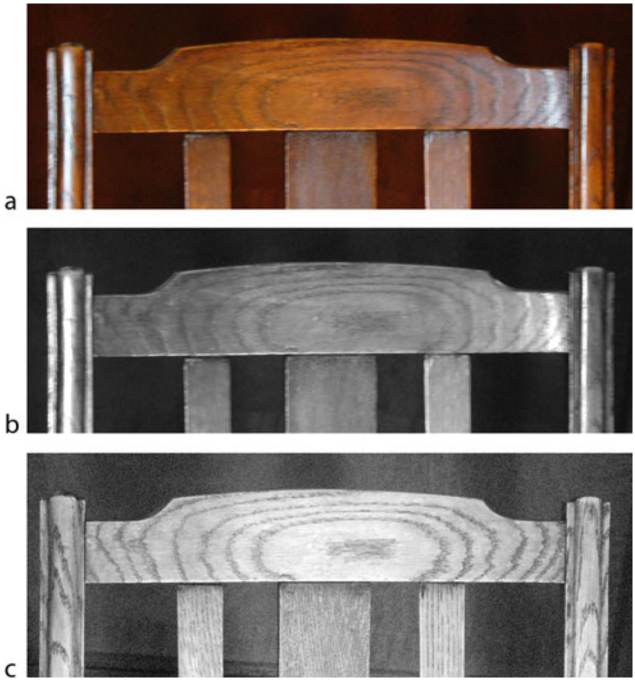


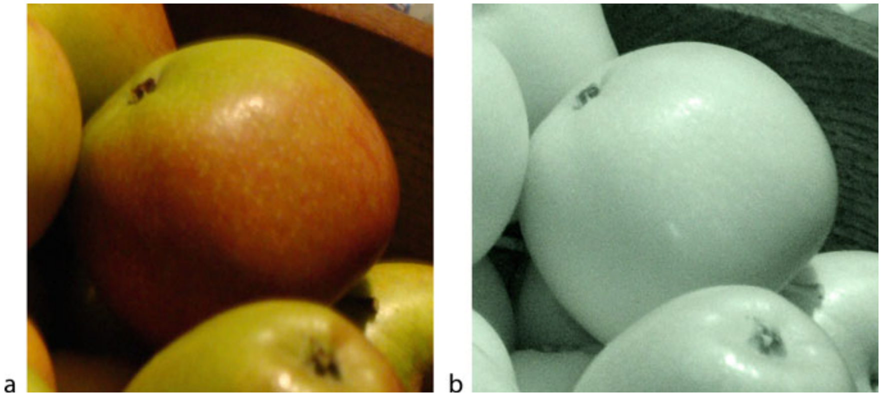
Fig. G.2.7

Another example showing that the visibility through paint improves with longer wavelength. Shipping container (a) VIS image. (b) IR image. Waveband: 1,300–1,700 nm. (c) IR image. Waveband: 1,500–2,500 nm. (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/ Reproduced with permission)



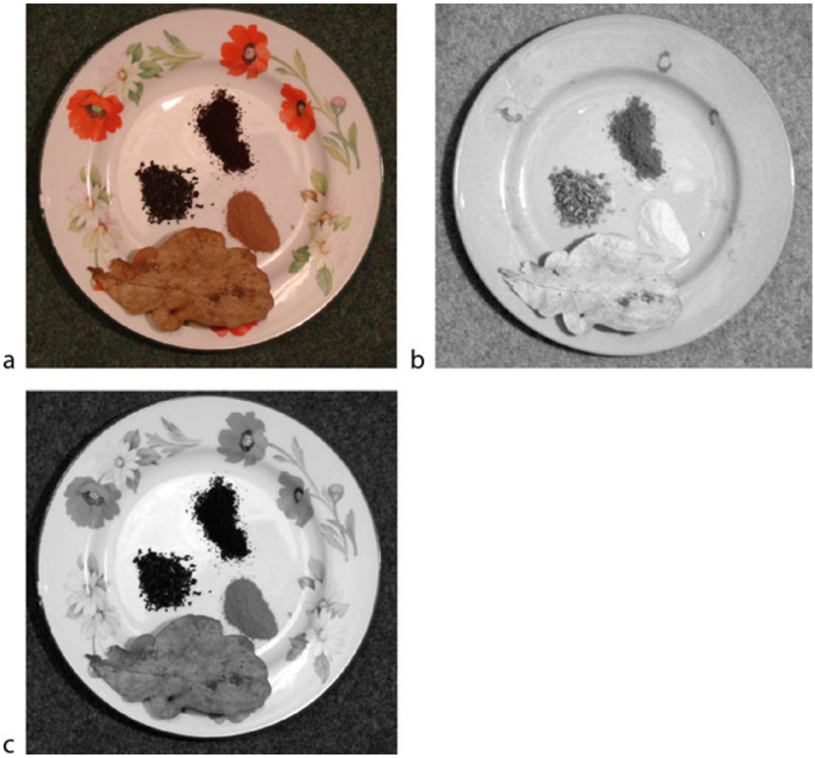
■ **Fig. G.2.8**

Seeing through glossy lacquer is improved by viewing in infra-red. Wood grain. (a) VIS image. (b) Intensity component of the VIS image. (c) IR image. Waveband: approximately 720–1,100 nm. (Image obtained using the Cyber-shot DSC-H9 camera, modified as explained above)

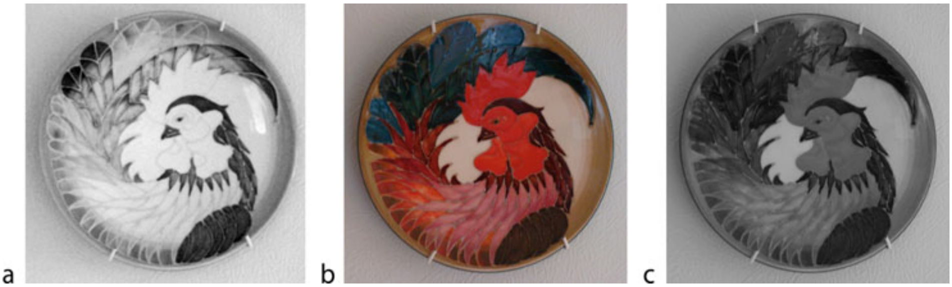


■ **Fig. G.2.9**

Colouration of fruit is often invisible in infra-red. Apples viewed in diffuse natural light. (a) VIS image. (b) Infra-red. Camera response: approximately 720–1,100 nm (Image obtained using the modified Cyber-shot DSC-H9 camera)



◘ Fig. G.2.10
IR response is inpredictable from everyday experience. (a) VIS image. Tea, coffee, chocolate and a dead oak leaf. (b) NIR image, obtained using the modified Cyber-shot DSC-H9 camera. (c) Intensity component, derived from the VIS image



◘ Fig. G.2.11
Decorative ceramic wall plate. (a) NIR image, obtained using the modified Cyber-shot DSC-H9 camera. (b) VIS image (c) Intensity component of the VIS image

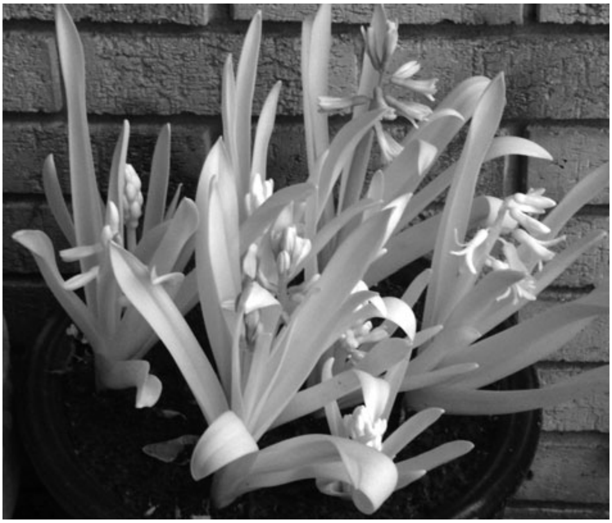


Fig. G.2.12

Foliage and flowers (hyacinths) in NIR (Image obtained using the modified Cyber-shot DSC-H9 camera) Foliage appears bright at these wavelengths. Recall that even dead leaves can do so

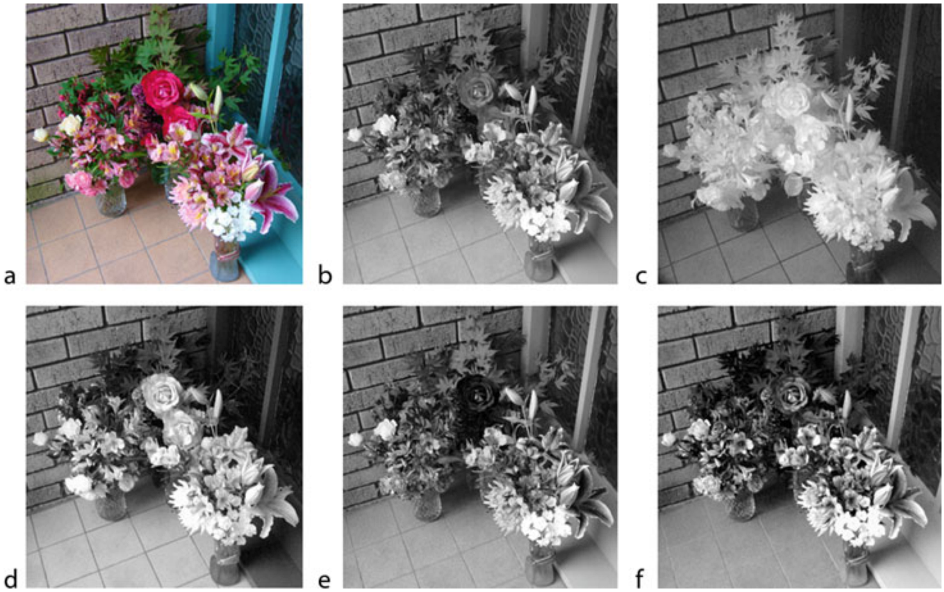
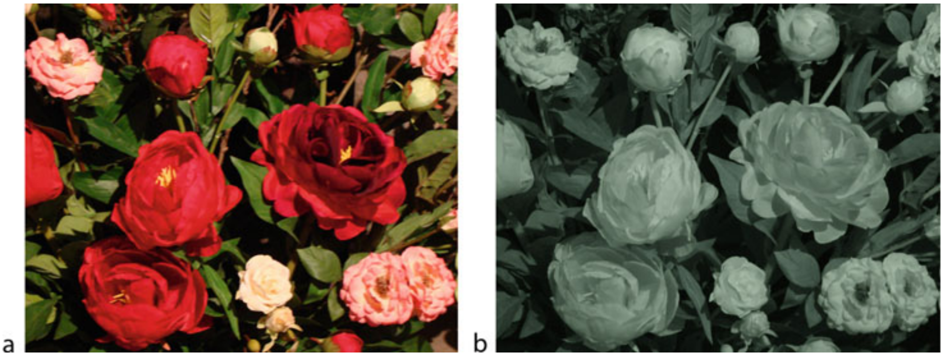


Fig. G.2.13

Flowers and green vegetation normally reflects NIR. (a) VIS image. (b) Intensity component. At each pixel, the intensity here is equal to the sum of the RGB values in the colour image. (c) NIR image. Waveband approximately 720–1,100 nm. (d) R-channel. (e) G-channel. (f) B-channel



■ **Fig. G.2.14**

Artificial flowers look natural in VIS but are easily identified as being synthetic using infra-red. (a) VIS image. (b) NIR image (720–110 nm) The flowers are bright as they would be for growing plants. However, the foliage is dark, which is quite different from the bright leaves in ➜ *Figs. G.2.12* and ➜ *G.2.13*

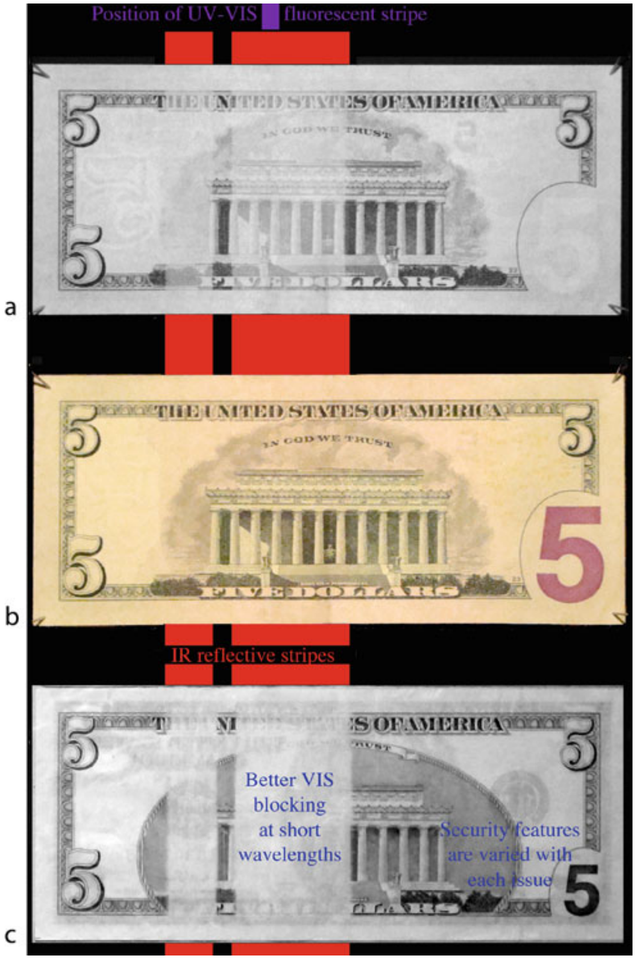
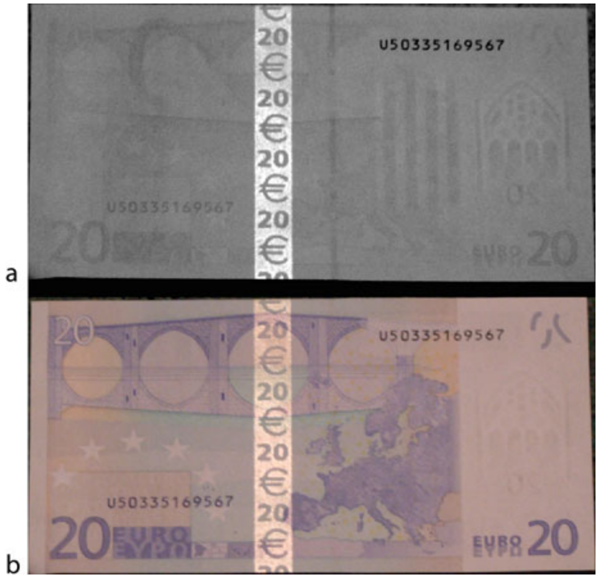

Position of UV-VIS fluorescent stripe
a
b
IR reflective stripes
Better VIS blocking at short wavelengths
Security features are varied with each issue
c

Fig. G.2.15
IR reflective ink enhances security on United States currency notes. (The UV–VIS fluorescent stripe is not visible in either VIS or IR.) (a) NIR image (waveband approximately 720–1,100 nm). Notice that the large numeral ''5'' has low contrast and that there are two broad vertical stripes where the picture of the Lincoln Memorial has reduced contrast. (b) VIS image. (c) IR image. The IR reflective stripes are more distinct at longer wavelengths



■ **Fig. G.2.16**

Security feature on a €20 currency note. (a) NIR image, exploiting specular reflection. (b) VIS image again arranged for specular reflection. The note also contains UV–VIS fluorescent security features but these are not visible here

G.2.1 Sample Thermal IR Images

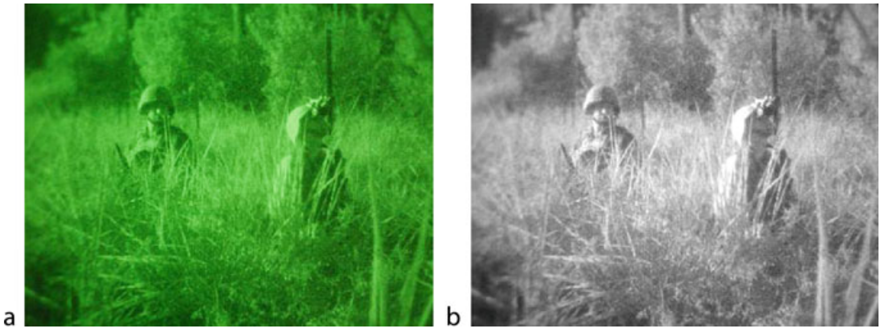


Fig. G.2.17
Night Vision camera. (a) Two soldiers seen through an image intensifier. The *green tint* is a product of the display and has no importance for Machine Vision. Source http://en.wikipedia.org/wiki/File:Nightvision.jpg (b) Intensity image. The grey-scale format is more relevant for industrial applications

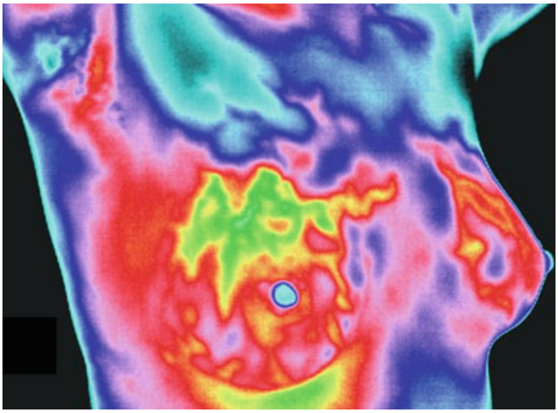


Fig. G.2.18
Thermography provides a fast, low-cost, non-intrusive diagnostic test for cancer. Here, the *green area* (hotter than the surrounding tissue) raises suspicions of cancer in the right breast. Pseudo-colour is often helpful where qualitative judgements are made by a person but has no real value for automated analysis in Machine Vision

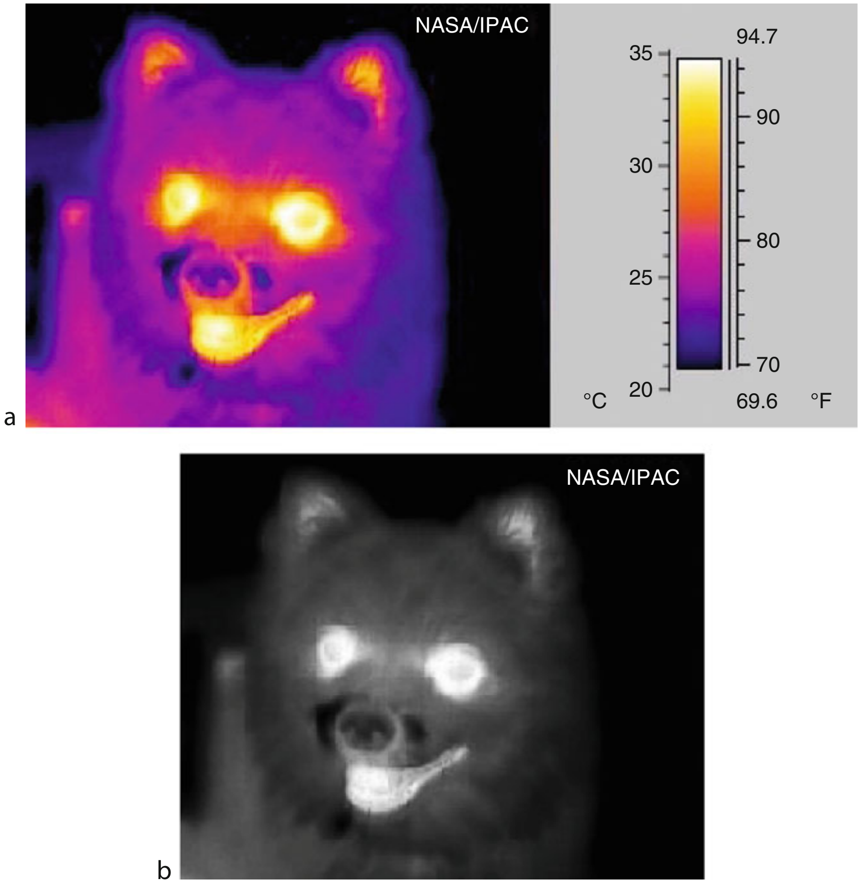


Fig. G.2.19

Thermal Infrared image (Thermograph). (a) Pseudo-colour image (Source NASA, Wikpedia. URL http://en.wikipedia.org/wiki/Thermographic_camera, accessed 10th Novemeber 2011) (b) The pseudo-colour mapping relates temperature to colour and does a good job at preserving a linear relationship between intensity and temperature. Hence, the mapping can be reversed, to yield an image that is more relevant to Machine Vision (When this reverse mapping is applied to the calibration scale, the result is a linear intensity "wedge")

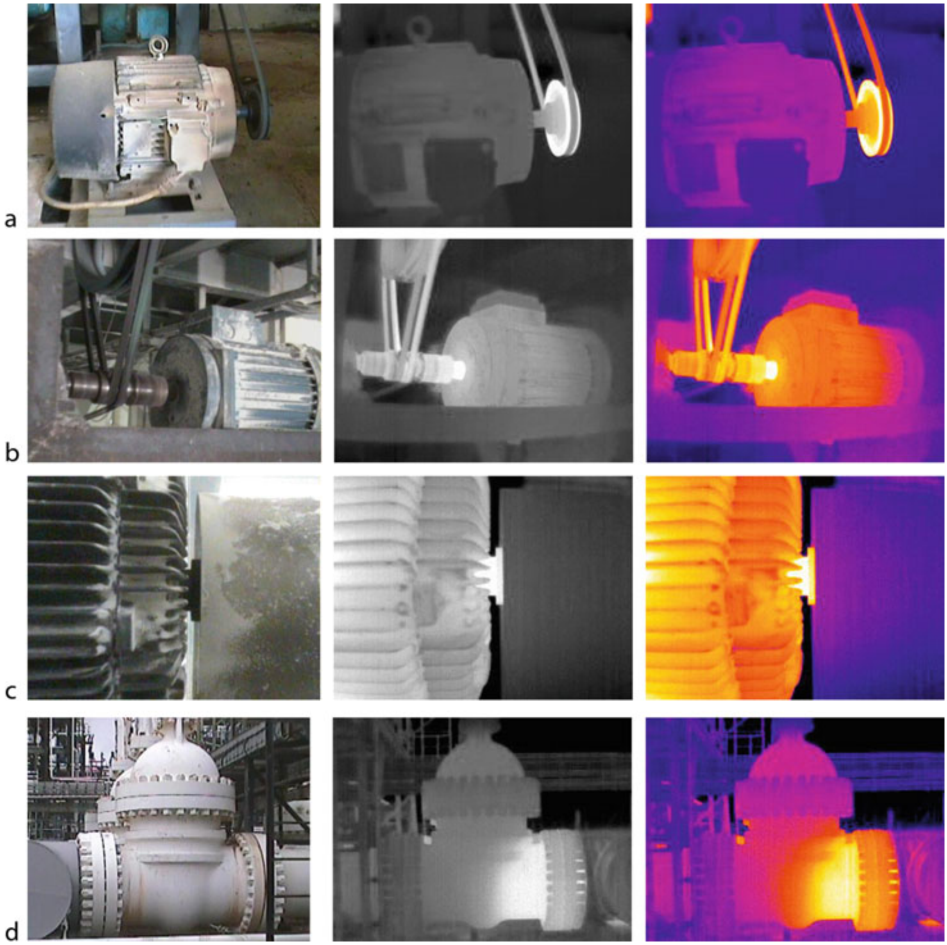


Fig. G.2.20

Thermal image. Showing, the VIS image (*left*), grey-scale IR image (*centre*) and pseudo-colour image (*right*). The grey-scale images were reconstructed by the author, by reversing the pseudo-colour mapping. (a) Belt and pulley. (b) Electric motor with a pulley and belt. It appears that the belt is slipping and causing heating by friction. (c) Motor shaft. This suggests that there is a faulty bearing. (d) Pipe and valve (shut). The VIS and pseudo-colour images were obtained by David Joseph (Infrared Vision, URL http://www.infraredvision.co.uk) using a ThermaCAM™ IR camera (FLIR Systems), URL www.flirthermography.com was used. Reproduced with permission

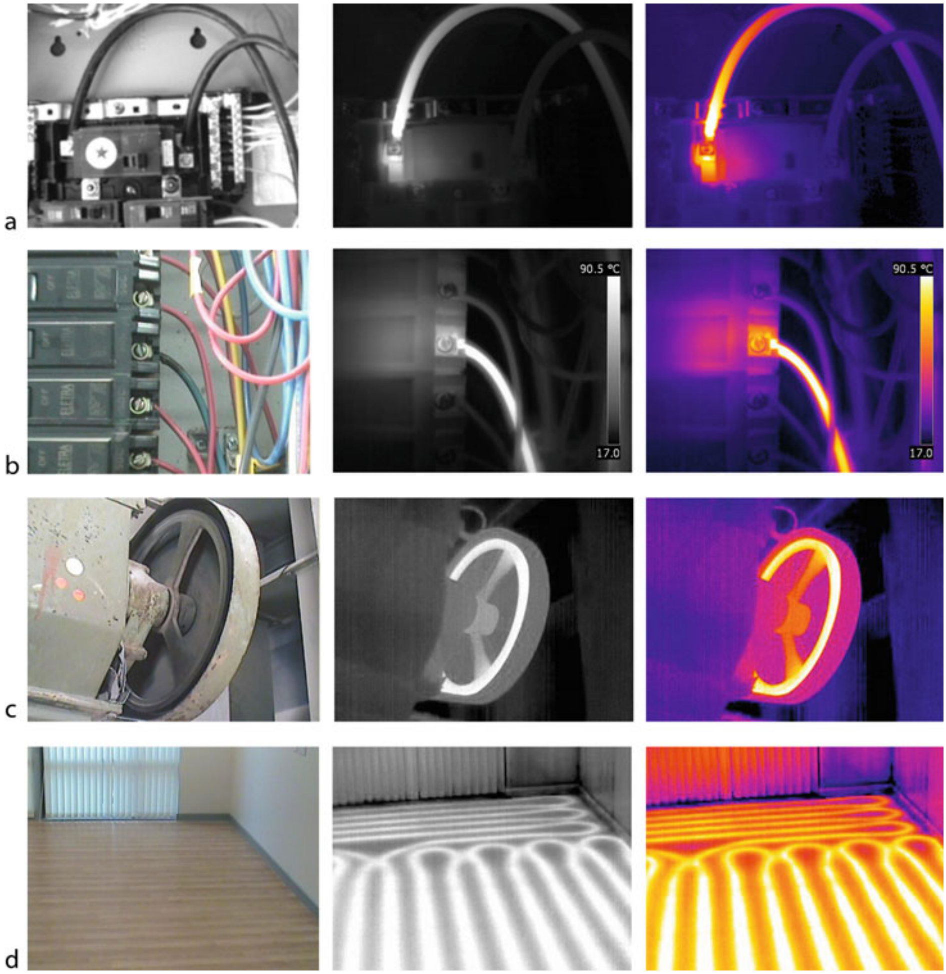


Fig. G.2.21

Thermal images. (a) Electrical wiring. (b) Electrical wiring, another example. Notice that the calibration scale on the right shows a ''wedge'' in which the intensity increases linearly along the vertical axis. This demonstrates the validity of the reverse mapping from pseudo-colour to grey-scale. (c) Pulley overheating as a result of a faulty belt. (d) Under-floor electrical heating. Images supplied by David Joseph (Infrared Vision, URL http://www.infraredvision.co.uk, Reproduced with permission)

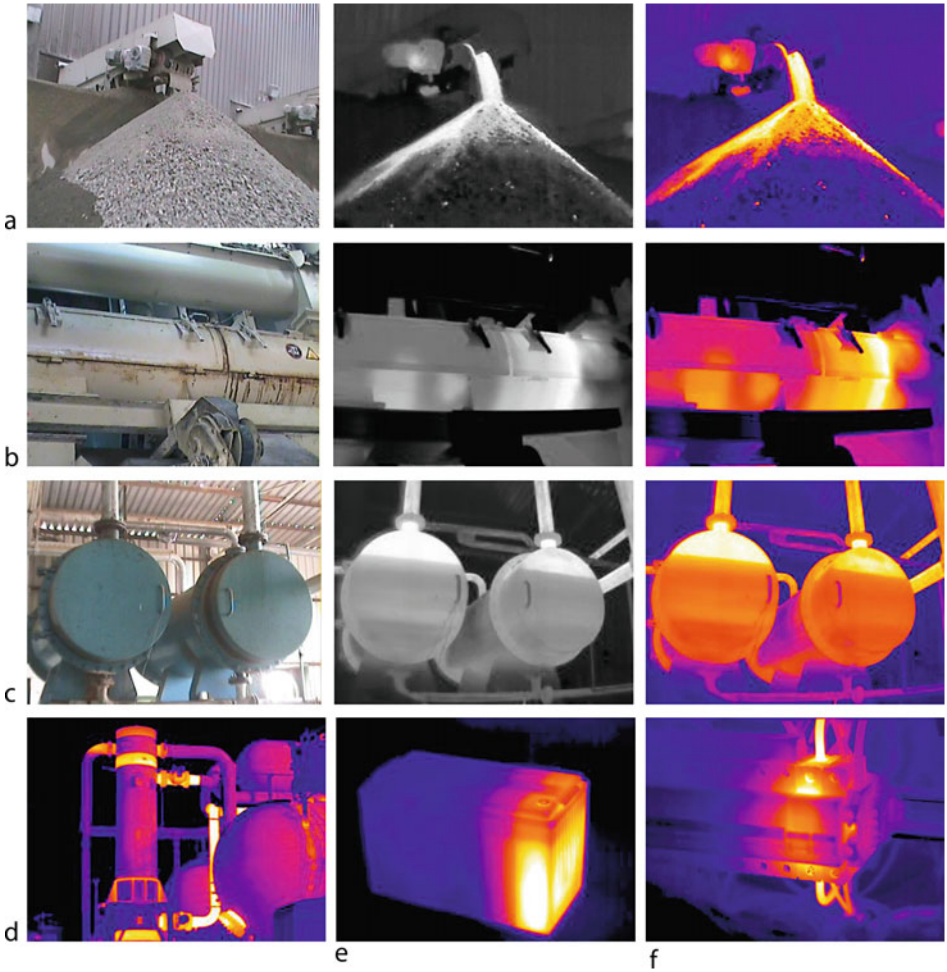
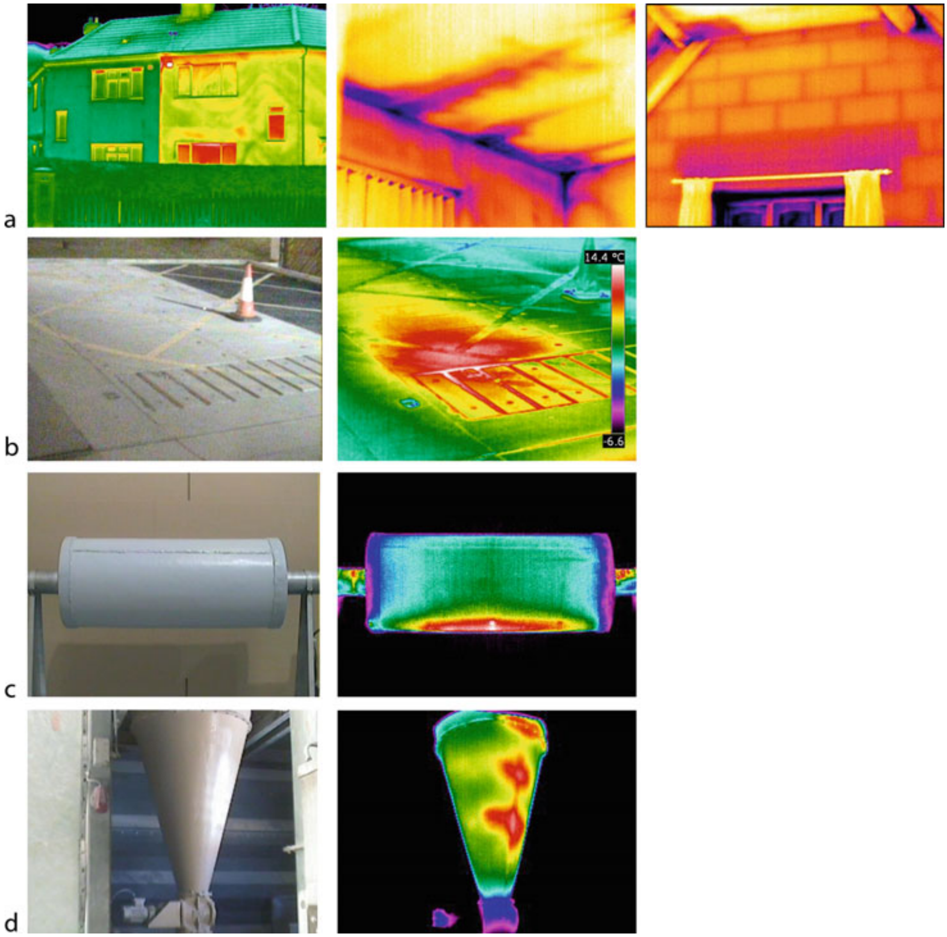


Fig. G.2.22

Thermal images. (a) Slag heap. (b) Rotary kiln. (c) Settling tanks. The horizontal striations record the filling history (''stepped'' rate of filling). (d) Petro-chemical plant. (e) Lead-acid accumulator whose end cells broke down while charging. (f) Electrical circuit breaker with a high-resistance connection. Images supplied by David Joseph (Infrared Vision, URL http://www.infraredvision.co.uk. Reproduced with permission)



■ **Fig. G.2.23**

Thermal IR images. (a) Locating sites of heat loss in buildings. This is a common area of application for IR cameras. (b) Underground steam escape. Notice the calibration scale. This pseudo-color mapping is not reversible. (c) Roller. (d) Hopper. Images supplied by David Joseph (infrared Vision, URL http://www.infraredvision.co.uk. Reproduced with permission)

G.3 Sample UV Images

The images in this section were all supplied by Dr. Austin Richards and are reproduced here with his permission. It should be noted that these images were obtained using UV lighting and sensor; they are not the result of fluorescence.

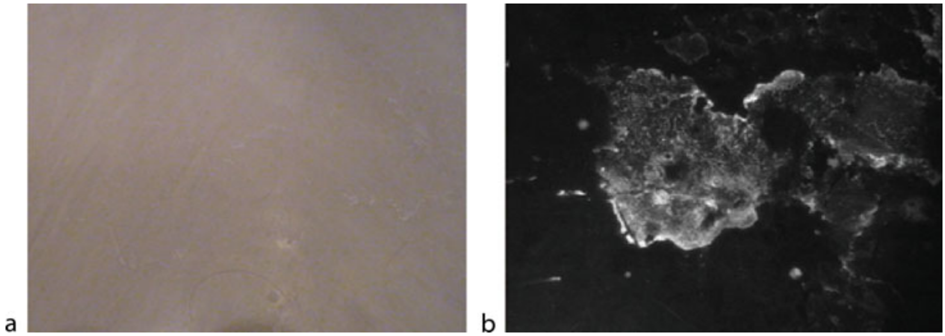


Fig. G.3.1

Toothpaste on a plastic counter-top. (a) VIS image showing no sign of the contamination. (b) UV image. Filter cut-off: 396 nm (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

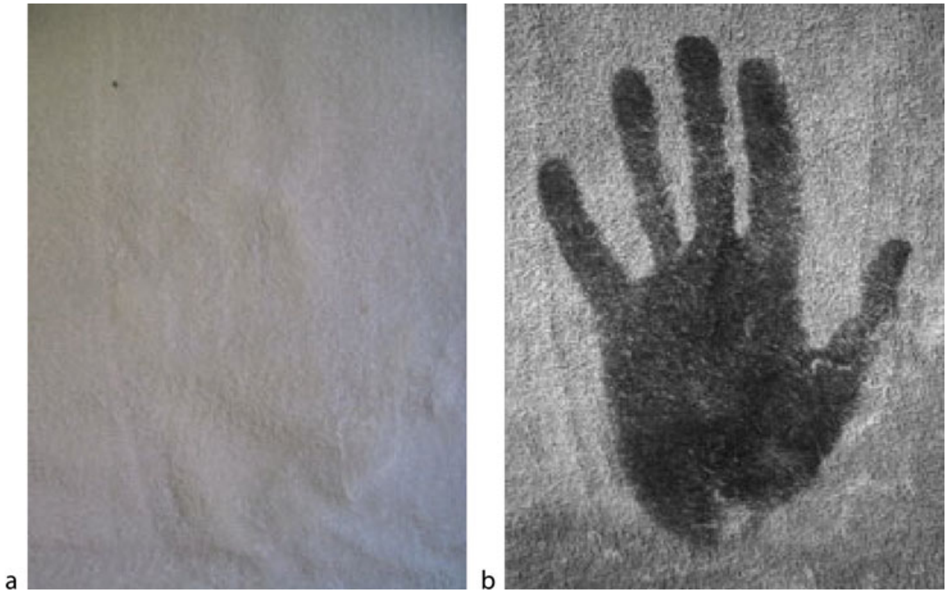


Fig. G.3.2

Sunblock hand-print on a white towel (Sunblock contains an effective UV absorber, such as avobenzone.) (a) VIS image. (b) UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

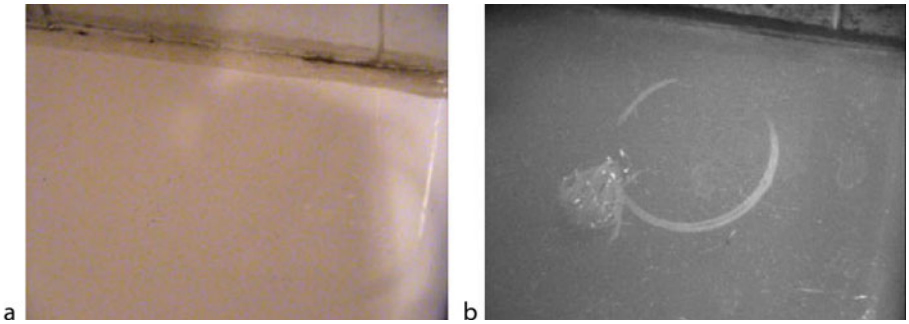


Fig. G.3.3
Traces of candle-wax and hair-dye on a bathtub. (**a**) VIS image. (**b**) UV image, wavelength 365 nm (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

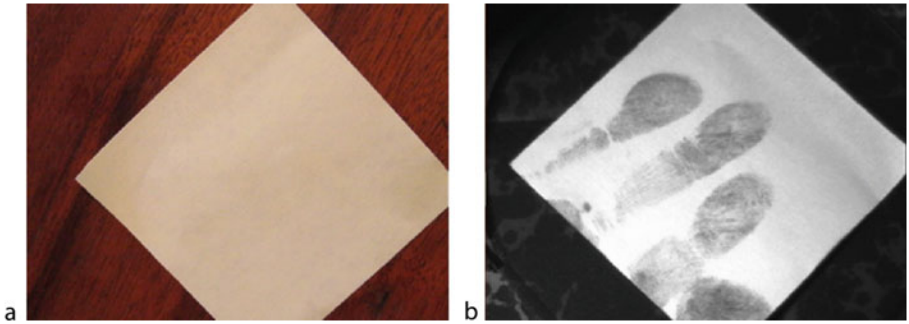


Fig. G.3.4
Moisturizer containing some sunblock. (**a**) VIS image. (**b**) UV image (These finger-marks can also be detected using UV–VIS fluorescence) (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

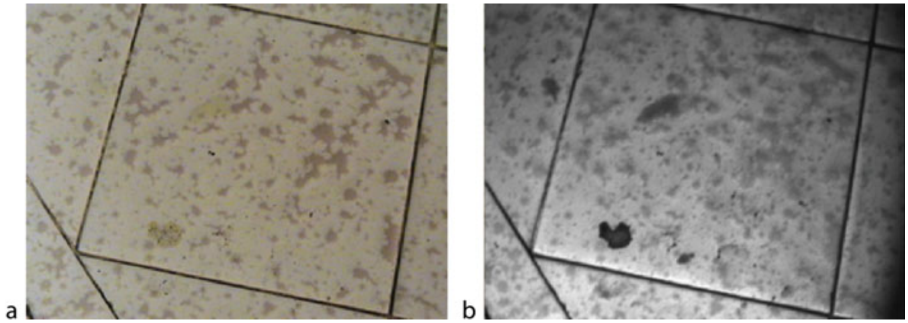
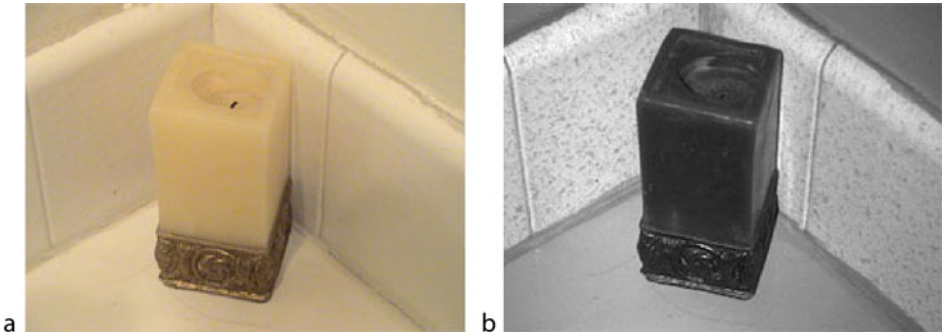
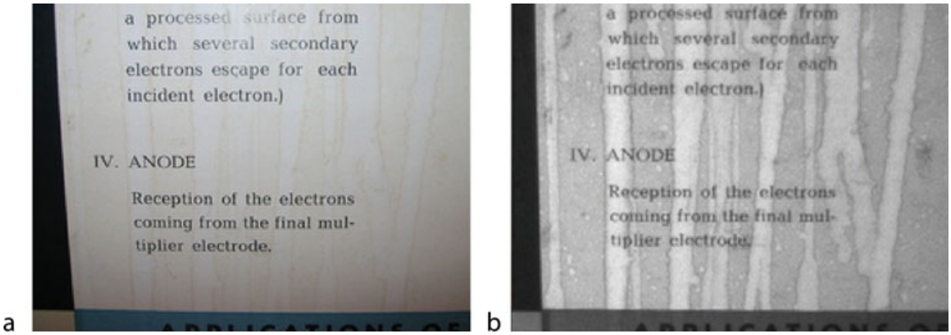


Fig. G.3.5
Kitchen tiles with a hard ceramic glaze. (**a**) VIS image. (**b**) UV image (Grease, as many other long-chain hydrocarbons absorbs UV very strongly) (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)



■ **Fig. G.3.6**

Candle against ceramic tiles. (a) VIS image. (b) UV image (The freckle-like pattern in the tile contains yellow flecks that absorb UV strongly) (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)



a processed surface from which several secondary electrons escape for each incident electron.)

IV. ANODE

Reception of the electrons coming from the final multiplier electrode.

APPLICATIONS OF

■ **Fig. G.3.7**

Water stains on paper. (a) VIS image. (b) UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

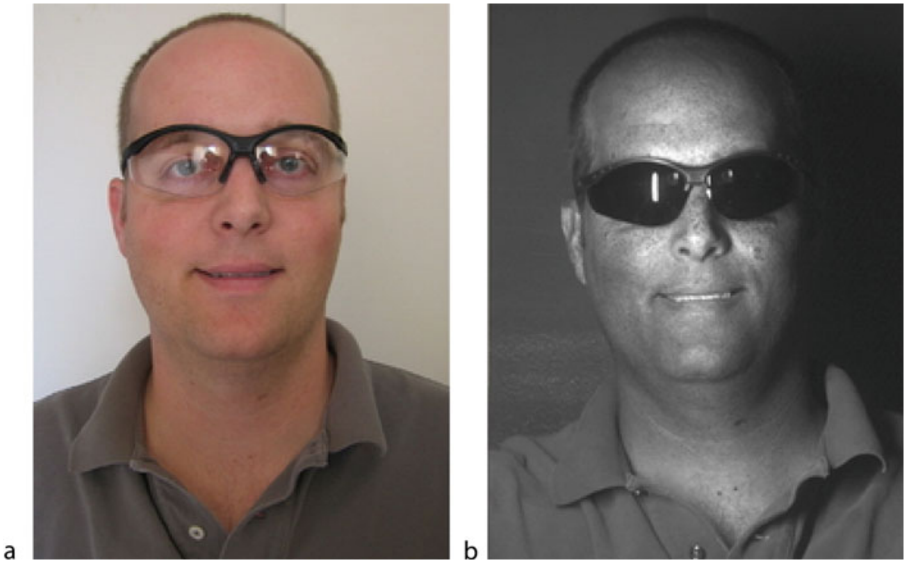


Fig. G.3.8

Plastic safety glasses with a UV absorbing coating. (a) VIS image. (b) UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

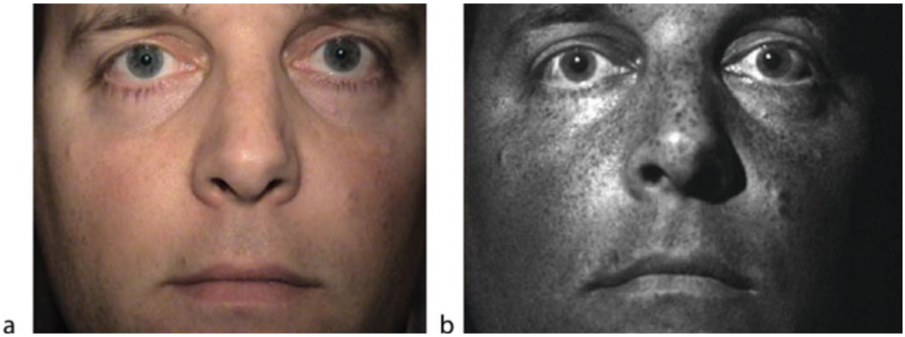
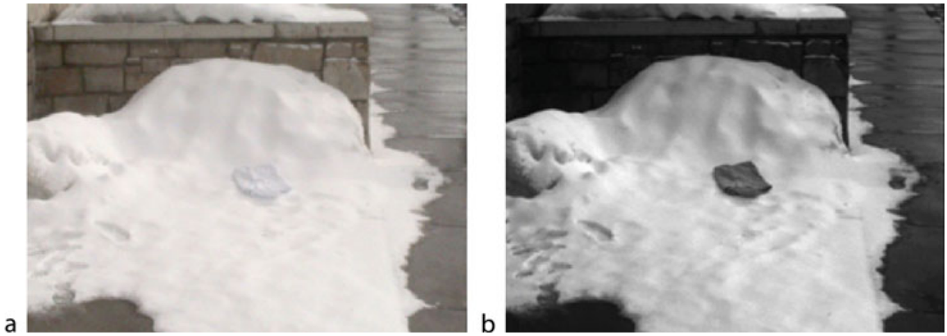


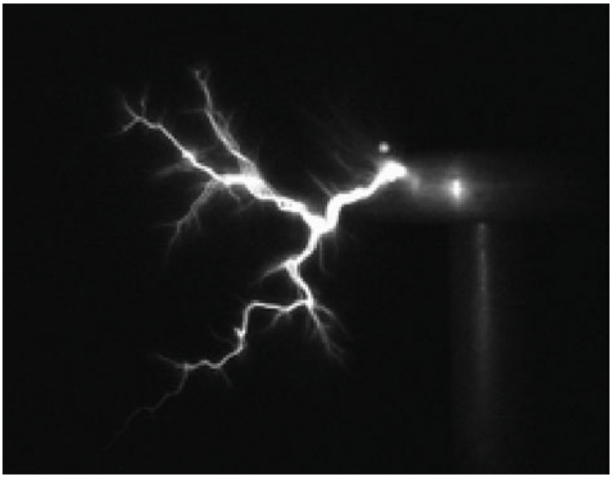
Fig. G.3.9

Facial skin with mild sun damage (Sun damage on skin manifests itself is as dark patches of excess melanin produced in response to UV exposure. This is much easier to visualize in the UV than in the visible band. Burns, bruising, bite marks, avulsive wounds and injection sites can also be seen in the UV, sometimes months after they have faded to the eye. Forensic odontologists often attempt match bite marks to the teeth of a suspected assailant using UV photography) (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)



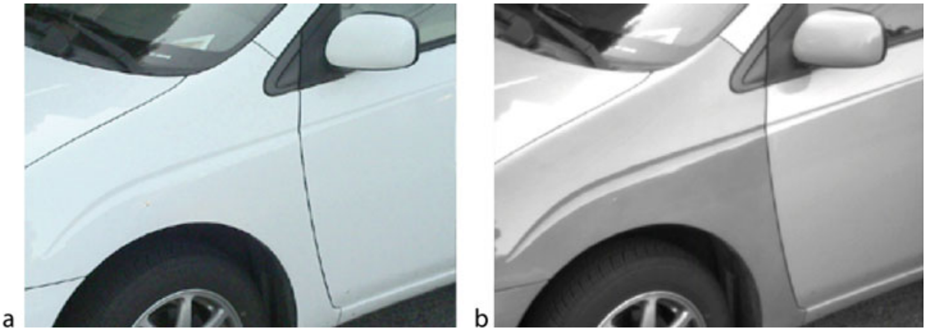
■ **Fig. G.3.10**

Cotton cloth on snow. (a) VIS image. (b) UV image (Many white fabrics used for snow camouflage are clearly visible in UV) (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)



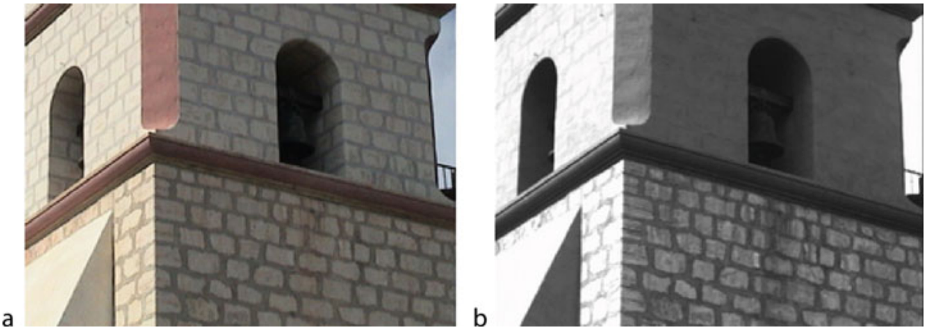
■ **Fig. G.3.11**

Imaging Electrical Discharges in UV (High voltage discharges produce excitation of atmospheric nitrogen, leading to emission of a weak deep-purple light that also has a strong UV component)



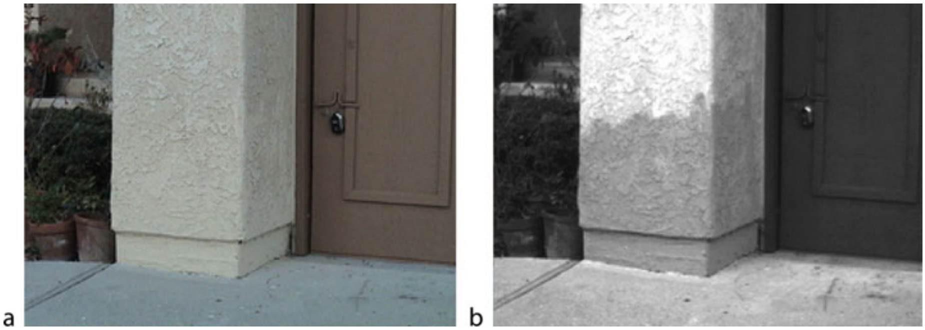
■ Fig. G.3.12

Toyota Prius car with repainted wing panel. (a) VIS image. (b) UV image (The contrast is impossible to detect visually) (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)



■ Fig. G.3.13

Masonry tower with a superstructure that has artificial stone-and-mortar patterns painted on. (a) VIS image. (b) UV image. Notice the water stain (*centre-bottom*) in the UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)



■ Fig. G.3.14

Repainted exterior stucco. (a) VIS image. (b) UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

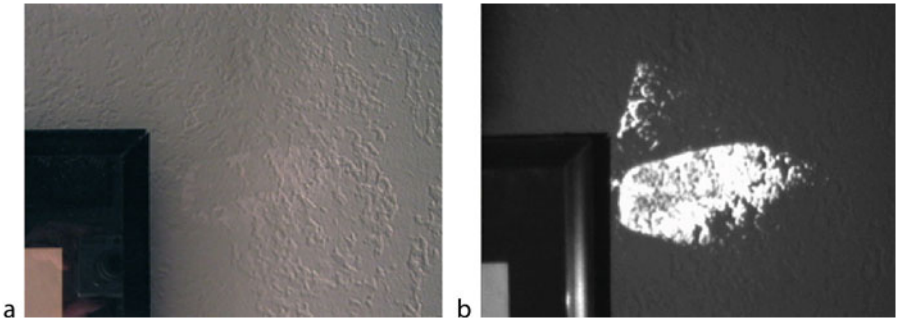


Fig. G.3.15

Painted plaster wall with fresh Spackle™ repairs. (a) VIS image. (b) UV image (Spackle™ is a surface filler used by decorators. Spackle™ is a registered trademark of the Muralo Company) (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

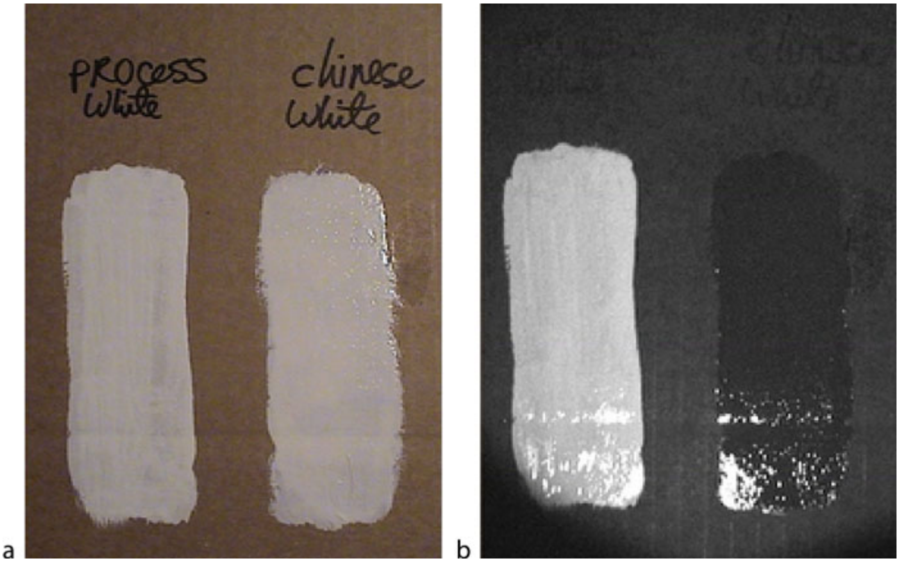


Fig. G.3.16

Two types of white paint. (a) VIS image. Process White (*left*) is a watercolour that accurately reproduces in photographs and is very reflective to UV light at 365 nm. Chinese White (*right*) is not nearly so reflective of UV (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

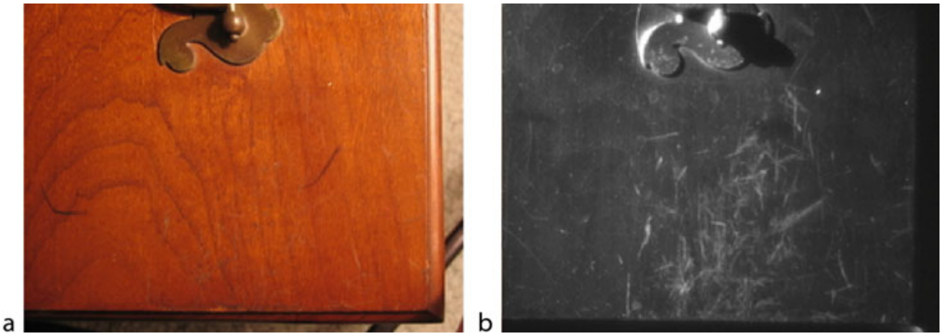


Fig. G.3.17
Scratches on varnished wooden surface. (**a**) VIS image (Scratches are very hard to see because the varnish is transparent to visible light) (**b**) UV image. Illumination wavelength: 365 nm (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

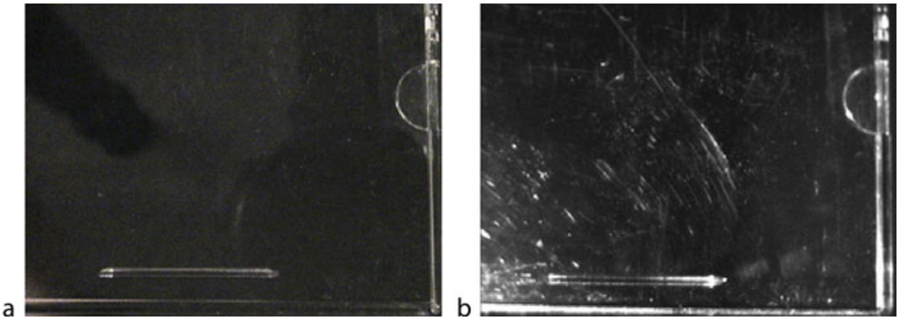


Fig. G.3.18
Scratched CD jewel case. (**a**) VIS image. (**b**) UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

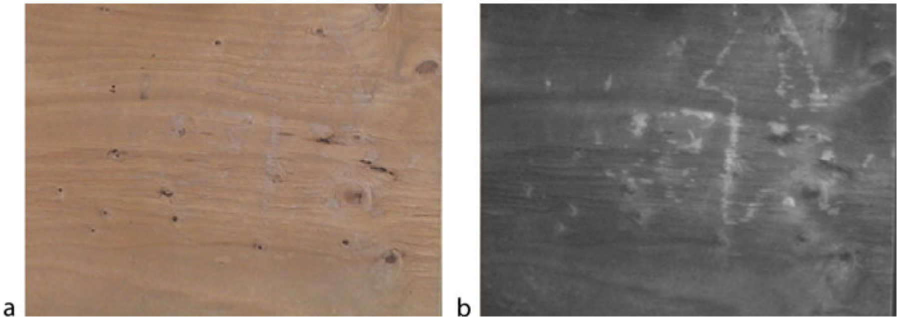


Fig. G.3.19
Dirt on plywood. (**a**) VIS image. (**b**) UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

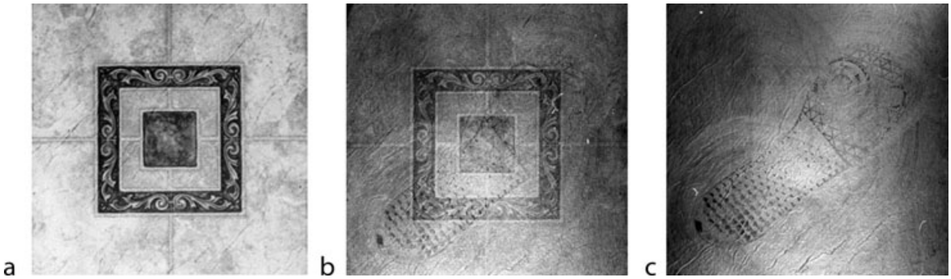


Fig. G.3.20

Footprints impressed in floor wax on a linoleum floor tile. (**a**) VIS image. (**b**) UV image. Wavelength: 365 nm band. (**c**) UV image. Wavelength 310 nm (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

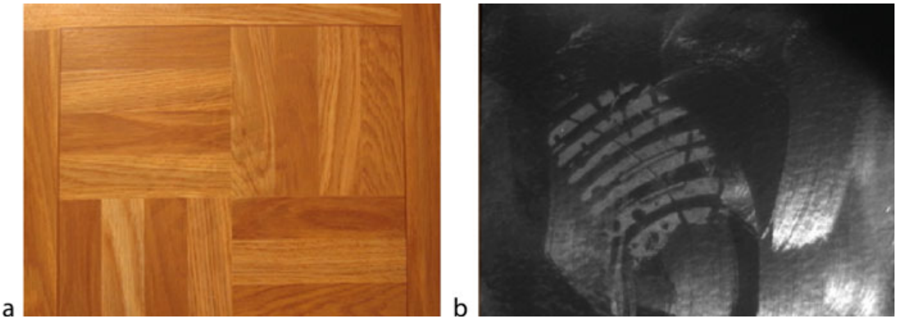


Fig. G.3.21

Shoe imprint on a waxed vinyl floor tile. (**a**) VIS image. (**b**) UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

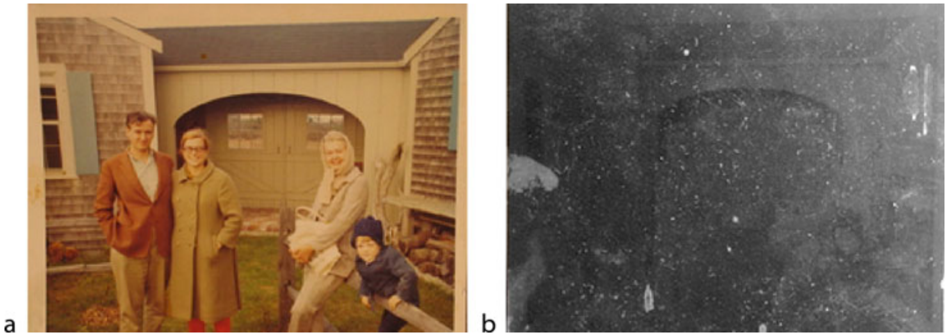


Fig. G.3.22

An old photograph. (**a**) VIS image. (**b**) UV image (The scratched and stained clear coating absorbs UV, which therefore does not reach the photographic image) (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

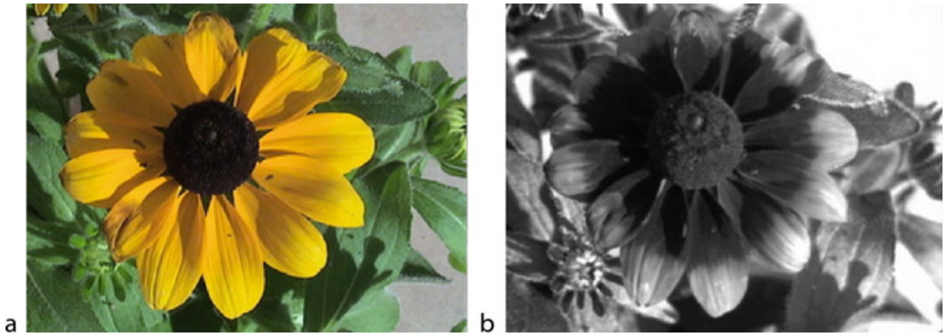


Fig. G.3.23
Flower, Black-eyed Susan (a) VIS image. (b) UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

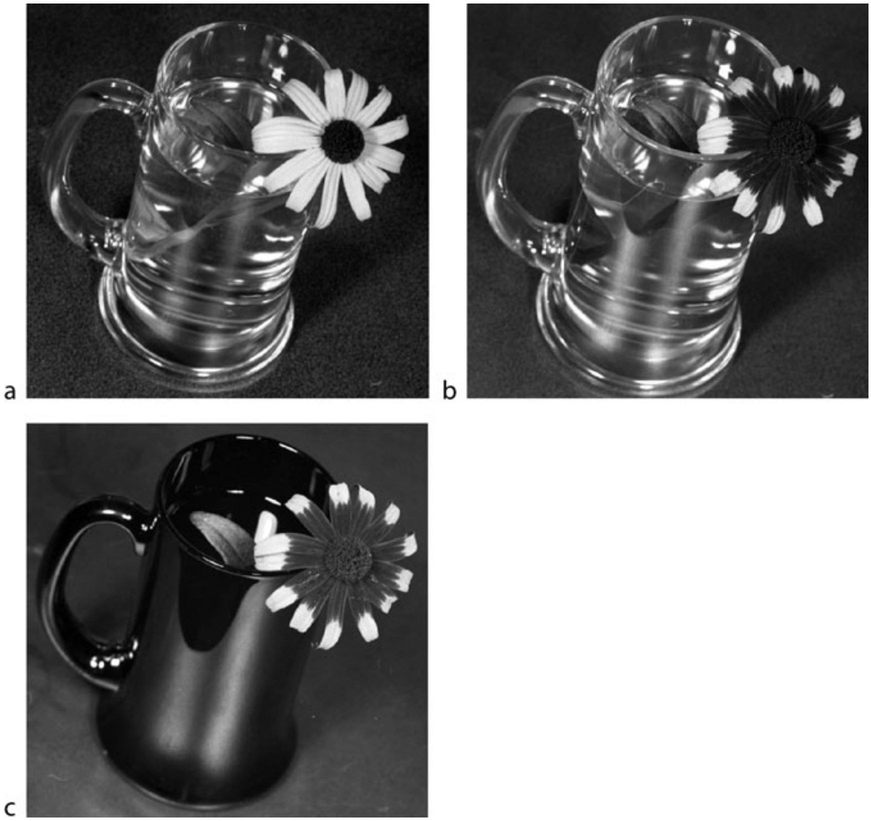


Fig. G.3.24
Glass, water, flower and stippled table. (a) VIS image. (b) UV image. Wavelength 310 nm. (c) UV image. Wavelength 360 nm (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

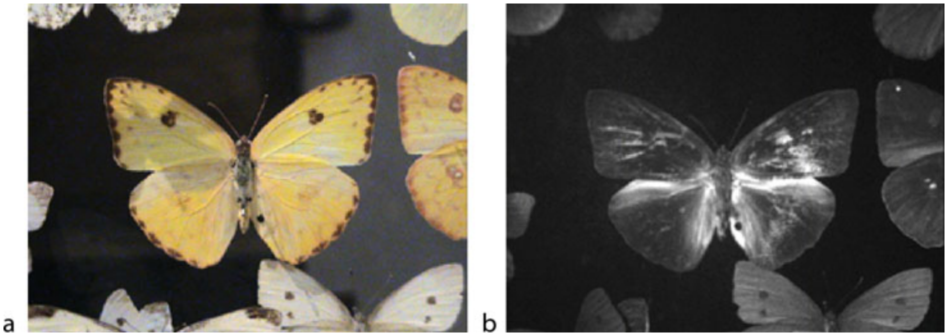


Fig. G.3.25
Butterflies. (a) VIS image. (b) UV image (Images supplied by Dr. Austin Richards, URL http://www.austinrichards.com/. Reproduced with permission)

References

1. Infra-red characteristics (2010) http://photo.net/learn/optics/edscott/ir000010.htm. Accessed 18 March 2010
2. IR and UV Photography (2010) http://www.mat.uc.pt/~rps/photos/FAQ_IR.html. Accessed 18 March 2010
3. Spectral Selectivity (2010) http://photo.net/learn/optics/edscott/spectsel.htm. Accessed 18 March 2010
4. Vision, URL (2010) http://hyperphysics.phy-astr.gsu.edu/hbase/vision/visioncon.html#c1. Accessed 18 March 2010
5. Richards A (2001) Alien Vision. SPIE, Bellingham, ISBN 0-8194-4142-2
6. Richards A http://www.austinrichards.com/
7. Indigo Systems (2010) http://www.flir.com/. Accessed 23 March 2010
8. Pixel Thermographics (2010) http://www.pixelthermographics.co.uk/. Accessed 26 March 2010
9. Infrared Vision (2010) http://www.infraredvision.co.uk/. Accessed 30 March 2010
10. Reflected UV Photogrphy (2010) Williams R, Williams G, http://msp.rmit.edu.au/Article_01/13.html. Accessed 19 May 2010
11. Enter the Unreal World of UV Color Photogrpahy, Bjørn Rørslett (2010) http://www.naturfotograf.com/uvstart.html. Accessed 19 May 2010

Appendix H: Connecting QT to External Devices

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_49,

The purpose of this appendix is to describe how QT can be set up with low-cost hardware, to build an experimental tool kit. The author has set up such a system in his own home, demonstrating that beginning in Machine Vision is possible without incurring great expense. A low-cost camera and interface board allows QT to capture digital images and control simple external hardware devices. This arrangement is certainly adequate for gaining familiarity with the technology and developing concept-level solutions for a wide range of non-demanding applications.

H.1 Acquiring Images

Digital images can be captured for subsequent processing by QT in several ways:

(a) Flat-bed scanner (A standard "domestic" scanner is able to provide good quality images in JPEG format.)
(b) Digital camera (A standard still digital camera can provide good image quality. However, this option does not provide easy machine-user interaction.)
(c) MATLAB Image Acquisition Toolbox [http://www.mathworks.com]. This offers a wide variety of options for image capture

A fourth option exists: images can be captured, via the web, using QT command *gwi*. A web-enabled camera with an Ethernet connection (for example, AXIS 210), can be connected directly to the computer's network port, using a cross-over cable. Of course, this disables its web connection. Set up the camera to operate as a simple web server, storing the most recently digitised image at a fixed-address. Then, use *gwi* to download that image, just like any other web file.

Another option for image capture is described later in this section.

H.1.1 Image Format

For convenience, images should be stored in JPEG or TIFF format. However, MATLAB can read other formats, using the command *imread*, including

• BMP	Windows Bitmap
• JPEG	Joint Photographic Experts Group
• PNG	Portable Network Graphics
• JPEG 2000	Joint Photographic Experts Group 2000
• GIF	Graphics Interchange Format
• PBM	Portable Bitmap
• TIFF	Tagged Image File

See [http://www.mathworks.com]

H.1.2 Location

For ease of use, image files should be placed in either of the two following folders:

- *BGB_MatLab_work_files*. Files can have alphabetic, alphanumeric, or numeric names and are read using *rei* or *rea*. The folder *BGB_MatLab_work_files* is intended for storing temporary files.
- *BGB_MatLab_Images*. Files should be in JPEG or TIFF format and have numeric names. They are read using *rni* (grey-scale, or binary), or *rci* (colour). The folder *BGB_MatLab_Images* is intended for storing image files permanently. The command *uia* adds the *Current* image in the lowest-numbered empty slot.

  To use use other folders, simply adjust the contents of the file qt_paths.m

H.1.3 USB Camera

Another option exists for image capture, providing the user to with a dynamic image display. This is particularly useful when adjusting the camera focus. This uses a USB camera, such as one of the following

- *Dino-Eye* USB camera [http://www.dino-lite.eu] The AM423C camera in the *Dino-Eye* range provides a C-mount lens fitting and can therefore be attached to a range of optical devices, such as microscopes, endoscopes, wide-angle/zoom lens, etc.
- *iSight* USB camera [http://www.apple.com/support/isight/] This has a motorised drive, enabling software control of focus, gain, compensation of colour temperature, etc.

These cameras can be operated using a software driver such as EvoCam. [http://www.evological.com/] The software should be set to capture images periodically (e.g., 1 image/s), downloading the digitised image into a file with a specified name. The QT commands *isight* and *frz* have been programmed to read that file.

H.2 Controlling the Velleman K8055 Interface Board Via QT

The *K8055 USB Experiment Interface Board*, pictured below, is a low-cost micro-controller device and is well suited for use in education, training and hobbyist applications, as well as providing "quick-fix" engineering solutions. It allows a computer to control the following I/O channels via a single USB port:

- Five digital input channels
- Eight digital output channels
- Two analogue inputs
- Two analogue outputs

Power is supplied by the host computer via its USB port. (70 mA load) Further details of the board can be found at:

http://www.velleman.eu (accessed 13th July 2010)

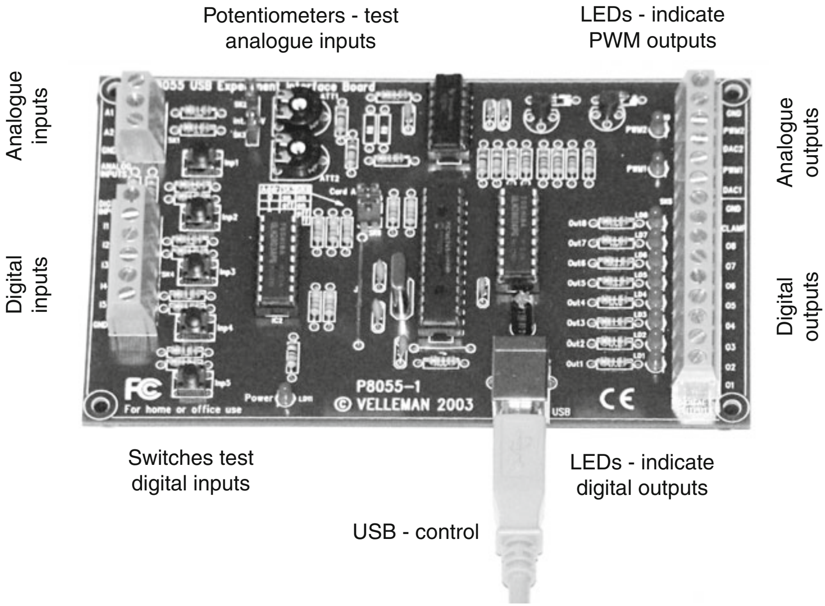

Potentiometers - test analogue inputs
LEDs - indicate PWM outputs
Analogue inputs
Digital inputs
Analogue outputs
Digital outputs
P8055–1
© VELLEMAN 2003
Switches test digital inputs
LEDs - indicate digital outputs
USB - control

Fig. H.1
The Velleman K8055 USB Experiment Interface Board

The K8055 Board can be controlled from software written in Delphi, Visual Basic, C++ or any other 32-bit Windows application development tool that supports calls to a Dynamic Link Library (DLL). Up to a maximum of four boards can be connected together, thereby increasing the number of I/O channels available.

A similar, more powerful module, Velleman Extended USB Interface Board, model no. K8061, is also available from the same manufacturer. This provides more digital and analogue I/O channels, with higher resolution on the latter.

Another range of devices with roughly similar capabilities is available from Arduino (URL: http://arduino.cc/, accessed 6th January 2010)

H.2.1 Software

The MATLAB code listed below was developed and operated on a Macintosh computer running the following software:

OS X, v 10.6.2
Parallels DeskTop 4.0 for Mac (URL: https://www.parallels.com/uk)
Windows XP (Note: The K8055 module is controlled via Windows, not OS X.)
MATLAB R2009b (URL: http://www.mathworks.co.uk)
The QT functions are based on software provided at the following web site:
http://hackhole.blogspot.com/2007/11/interface-velleman-k8055-usb-board-with.html

H.2.2 Connecting to the Board

First, open the USB connection between the computer and K8055 board. Under the environment described above, this is achieved by selecting the following menu item in Parallels:

```
Devices->USB->USB K
```

It is then necessary to run *vel_start* before any other commands are sent to the board

```
function vel_start(a)
% Load K8055 DLL & H files. Then, connect to specified board address
disp('Go to menu item Devices->USB->USB K8055 first')
if nargin == 0
   ChannelAdress=0;
else
   ChannelAdress=input(' Channel (0-3): ');
end
fprintf (' Status:')
addpath('C:\Program Files\MATLAB\R2009b\');
loadlibrary('K8055D.dll','K8055D.h');
tmp=calllib('K8055D', 'OpenDevice',ChannelAdress);
if (tmp == -1)
   fprintf ('\tBoard%d not connected\n\n',ChannelAdress)
else
   fprintf ('\tConnected to board%d\n',ChannelAdress)
libfunctionsview('K8055D');
end
vel_test
```

Closing the USB Connection

```
function vel_close
calllib('K8055D', 'CloseDevice')
```

Setting one/all digital output channels

```
function vel_set_dig_out(a,b)
global vel_digital_output_channel
if nargin == 1
   calllib('K8055D','SetDigitalChannel',a);
   vel_digital_output_channel(a)=1;
else
   if nargin == 0
      for i=1:8
      calllib('K8055D','SetDigitalChannel',i);
      vel_digital_output_channel(i)=1;
      end
   else
```

```
    if b == 0
      calllib('K8055D','ClearDigitalChannel',a);
      vel_digital_output_channel(a)=0;
     else
      calllib('K8055D','SetDigitalChannel',a);
      vel_digital_output_channel(a)=1;
     end
    end
end
vel_status_dig_out;
```

Clearing One/All Digital Output Channels

```
function vel_clear_dig_out(a)
global vel_digital_output_channel
if nargin == 0
  for i=1:8
  calllib('K8055D','ClearDigitalChannel',i);
  vel_digital_output_channel(i)=0;
  end
else
  calllib('K8055D','ClearDigitalChannel',a);
  vel_digital_output_channel(a)=0;
end
vel_status_dig_out
```

Read One of the Digital Input Channels

```
function b=vel_read_digital(a)
b=calllib('K8055D','ReadDigitalChannel',a);
```

Set One of the Analogue Output Channels

```
function vel_set_analog_out(a,b)
global vel_digital_output_channel
global vel_analog_output_channel
calllib('K8055D','ClearAnalogChannel',a)
libpointer=b; calllib('K8055D','SetAnalogChannel',a);
```

Read One of the Analogue Input Channels

```
function b=vel_read_analog(a)
b=calllib('K8055D','ReadAnalogChannel',a);
```

Test the Connection

```
function vel_test(a)
if nargin == 0
  a=10;
end
for j=1:a
  vel_set_dig_out
```

```
    vel_clear_dig_out
end
for j=1:a
    for i=1:8
        vel_set_dig_out(9-i,1);
        vel_set_dig_out(9-i,0);
    end
end
```

Wait for Signal on a Given Digital Input Channel

```
function b=vel_wait_dig_input(a)
tic;
while vel_read_digital(a) == 0;
    b=toc; disp(['Waiting for input ', int2str(a), ' Delay: ',int2str
    (b), ' seconds'])
end
```

Wait for Signal on Any Digital Input Channel

```
function [b,c]=vel_wait_any_dig_input
tic;
while 1
    x=toc;
    disp(['Waiting for any digital input Delay: ',int2str(x),
    ' seconds'])
    for i=1:5
        if vel_read_digital(i)== 1
            b=i;
            c=toc;
            return
        end
    end
end
```

Wait for a High Voltage on Any Analogue Input Channel

```
function [c,d]=vel_wait_analog_input(a,b)
tic;
c=vel_read_analog(a);
d=toc;
while vel_read_analog(a)<b
            x=toc;
    c=vel_read_analog(a);
    disp(['Waiting for analogue input. Current voltage=',num2str
    (c),' volts....
            Delay: ',int2str(x), ' seconds'])
    d=toc;
end
c=vel_read_analog(a);
```

Appendix I: Information Sources

Bruce G. Batchelor
Cardiff University, Cardiff, Wales, UK

Bruce G. Batchelor (ed.), *Machine Vision Handbook*, DOI 10.1007/978-1-84996-169-1_50,

I.1 Measuring the Level of Interest

Table I.1 shows the number of "hits" found when searching the technical literature in May 2010.

The general search profile "*machine vision*" showed that there was a growth of activity during the previous 5 years:

393 in 2005
377 in 2006
500 in 2007
577 in 2008
560 in 2009

Table I.1

Number of "hits" found during automated searching of the technical literature on Machine Vision. (24th May 2010) All searches except the ones marked with an asterisk (*) were based on a profile with the general form ["machine vision" AND xxx OR. . .OR zzz]. The numbers indicate the approximate level of interest in Machine Vision within the research community and should not be interpreted as defining the complete picture of the body of published work in this area

Category	Profile (Approximate Wording)	No. of "Hits"
General	Computer vision*	39,426
	Machine vision	6,499
	Applications	2,529
	Optical inspection*	1,095
	Agriculture	9
Natural products/scenes	Animals	44
	Apples	63
	Cow/pig/sheep	11
	Fish	37
	Grain/cereal/rice	85
	Insects	25
	Nuts	7
	Vegetables	26
	Weeds	60
	Aircraft	73
	Automotive/automobile	254
Industrial objects/scenes	Bottles/glass/glassware	105
	Electronics/PCBs	326
	Inspection	1,589
	Piece parts	851
	Robotics	1,390
	Steel	82
	Surface	937
	Textile/knitware	72

Table I.1 (Continued)

Category	Profile (Approximate Wording)	No. of "Hits"
	Tile	19
	Cameras	1,740
	Fibre optics	733
Sensing medium	Fluorescence	47
	Infrared	32
	LED lighting	184
	Lighting	109
	Optics/filter/lens etc.	418
	Ultraviolet	1,631
	X-ray	20
	Colour	70

I.2 Web Resources: Information, Services and Products

Organisation	URL
Adobe	http://www.adobe.com/
Advanced Illumination	http://www.advancedillumination.com
Alacron	http://www.alacron.com/
APG	http://www.apgvision.com/
Arvoo	http://www.arvoo.com/
Automated Vision Systems	http://autovis.com/web2/index.php
Bitec	http://www.bitec.ltd.uk/
Bouldeeer Imaging	http://www.boulderimaging.com/index.html
Brainyech	http://www.braintech.com/
Carl Zeiss	http://www.zeiss.de
Carnegie Mellon Univ. Lists research groups, image databases, vision software, companies etc.	http://www.cs.cmu.edu/~cil/vision.html
CCDDirect	http://www.ccddirect.com/
Conduant	http://www.conduant.com/
Control Vision	http://www.controlvision.com/
CVI Melles Griot	http://optics.mellesgriot.com
CyberOptics Semiconductor	http://www.imagenation.com/
Daitron	http://www.daitron.com/
Dalsa	http://www.dalsa.com/
Dalsa	http://www.dalsa.com/mv/
Data Translation	http://www.datx.com/
DataCube	http://www.datacube.com/

Organisation	URL
Delft Univ. of Technology	http://www.ph.tn.tudelft.nl/~bokhove/imap/
DiectedPerception	http://www.DPerception.com/
DVC	http://www.dvcco.com/index.html
Edmund Optics	http://www.edmundoptics.com
Eltec Electronik	http://www.eltec.de/
Epigem	http://www.epigem.co.uk/
EPIX	http://www.epixinc.com/
European Optical Society	http://www.myeos.org
Eyetronics	http://www.eyetronics.com/
FDS Research	http://www.fdsresearch.com/index_frame_en.htm
Fiberoptics Technology	http://www.fiberoptix.com
General Electric Lighting	http://www.gelighting.com
Genex Technolgies	http://www.genextech.com/
GlobalSpec	http://process-equipment.globalspec.com/
Graftek Imaging	http://www.graftek.com/
Halcon	http://atlantek.com.au/
High Intensity Discharge	http://www.sylvania.com
i2S Linescan Imaging	http://www.i2S-linescan.com/
Illumination Eng. Society	http://www.iesna.org/
Image Labs International	http://www.vision1.com/
Imaging Source	http://www.theimagingsource.com/
Industrial Machine Vision	http://www.industrialmachinevision.com/
Ino	http://www.ino.ca/en/
inSpeck	http://www.inspeck.com/
IO Industries	http://www.ioindustries.com/
IQin Vision	http://www.IQeye.com/
Isra Vision	http://www.isravision.com/
JAI	http://www.jai.com/EN/Pages/home.aspx
Keyence	http://world.keyence.com/
Leutron	http://www.leutron.com/
Light Works	http:www.LW4U.com
Lord Imaging	http://www.lord-ing.com/
Lumitex	http://www.lumitex.com
Machine Vision Online	http://www.machinevisiononline.org/
Matrox Imaging	http://www.matrox.com/imaging/
MEMS Optical	http://www.memsoptical.com/
Metris	http://us.metris.com/home.php
Montrose Technologies	http://www.montrose-tech.com/index.html
Mueller	http://www.muellerr.ch
Multipix Imaging	http://www.multipix.com/

Organisation	URL
MuTech	http://www.mutech.com/
National Instof Health (USA)	http://rsb.info.nih.gov/nih-image/
National Instruments	http://www.natinst.com/imaq
Nerlite	http://www.nerlite.com
Newport Instruments	http://www.newport.com
Newton Research Labs	http://www.newtonlabs.com/
NextEngine	https://www.nextengine.com/indexSecure.htm
NFM	http://www.nfminc.com/html/em2.htm
Optasia Systems	http://www.singaporegateway.com/optasia/
Optel Vision	http://www.optelvision.com/
OSA – Home Page	http://www.osa.org
Pelco	http://www.pelco.com/
Perkin Elmer	http://optoelectronics.perkinelmer.com
Philips Lighting	http://www.lighting.philips.com
Pinnacle Vision	http://www.pinnaclevision.co.uk/bandit.htm
Pixelsmart	http://www.pixelsmart.com/
Point Grey Research	http://www.ptgrey.com/
PPT Vision	http://www.pptvision.com/
QImaging	http://www.qimaging.com/
RoboCams	http://www.robocams.com/
RoboRealm	http://www.roborealm.com/
Rocky Mountain Instrum.	http://rmico.com
Schott	http://www.us.schott.com
Seeing Machines	http://www.seeingmachines.com/
SenTech	http://www.sentechamerica.com/
Sick	http://www.sick.com/
SkySoft	http://www.skysoft-imaging.de/en/
Spectronix	http://www.spectronix.net/
SPIE	http://spie.org
Stocker Yale	http://www.stockeryale.com
Subtechnique	http://www.subtechnique.com/
Sundance	http://www.sundance.com/
Surveyor	http://www.surveyorcorp.com/
Sylvania	http://www.sylvania.com
Tectivity	http://www.tectivity.com/
Time Machines	http://www.time-machines.com/
TracLabs	http://www.traclabs.com/
TVI Vision	http://www.tvivision.com/
TYZX	http://www.tyzx.com/
UKA Optics	http://www.ukaoptics.com/

Organisation	URL
UKIVA	http://www.ukiva.org/
Videometer	http://www.videometer.com/
Videre Design	http://users.rcn.com/mclaughl.dnai/support.htm
Vision Components	http://vision-components.com/en/
Vision Light Technology	http://www.visionlighttech.com
Vision Systems Int.	http://www.vision1.com/vsi/
Vista Imaging	http://www.vistaimaging.com/
Visual Inspection Systems	http://www.visualinspections.it/VISUAL_UK.htm

I.2.1 Information on Specific Topics

Topic	URL
Abbe prism	http://en.Wk.org/wiki/Abbe_prism
Abbe-Koenig prism	http://en.Wk.org/wiki/Abbe-Koenig_prism
Achromatic doublet lenses	http://www.newport.com/Achromatic-Doublet-Lens-Tutorial/141058/1033/catalog.aspx
Achromatic lenses	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=312
Amici prism	http://en.Wk.org/wiki/Amici_prism
Amici roof prism	http://en.Wk.org/wiki/Amici_roof_prism
Anti-reflection coatings	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=247
Area lighting	http://www.nerlite.com/Area.html
Aspheric lenses	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=315
Atmospheric transmittance	http://en.wikipedia.org/wiki/Optical_window
Atomic spectra	http://physics.nist.gov/cgi-bin/AtData/pt?optionslist=XXT1/
Back light	http://www.nerlite.com/Backlight.html
Backlight	http://en.wikipedia.org/wiki/Backlight
Back-light	http://www.advancedillumination.com/pList.php?cat=6a204bd89f3c8348afd5c77c717a097a
Ballasts	http://www.sylvania.com/LearnLighting/LightAndColor/Ballasts/
Beamsplitter	http://www.ericweisstein.com/research/thesis/node98.html
Beamsplitters	http://www.mellesgriot.com/pdf/P_Handbook_Beamsplitters.pdf
Beamsplitters	http://www.newport.com/Beamsplitters/141111/1033/catalog.aspx
Bioluminescence	http://en.wikipedia.org/wiki/Bioluminescence
Black body	http://en.wikipedia.org/wiki/Black_body
Black light (UV)	http://en.wikipedia.org/wiki/Black_light
Borescopes & fiberscopes	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=1732
Bremsstrahlung	http://en.wikipedia.org/wiki/Bremsstrahlung
Calibration standards	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=1729

Topic	URL
Care and cleaning optics	http://www.newport.com/Care-and-Cleaning-of-Optics/141176/1033/catalog.aspx
CcMmYK color model	http://en.wikipedia.org/wiki/CcMmYK_color_model
Ceramic discharge metal halide	http://en.wikipedia.org/wiki/Ceramic_discharge_metal_halide_lamp
Cerenkov radiation	http://en.wikipedia.org/wiki/Cherenkov_radiation
Chemiluminescence	http://en.wikipedia.org/wiki/Chemoluminescence
Choosing illumination	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=264
CIE color models	http://dba.med.sc.edu/price/irf/Ad_tg/models/cie.html
Cleaning optics	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=265
Cloudy day lighting	http://www.nerlite.com/CDI.html
CMYK color model	http://en.wikipedia.org/wiki/CMYK_color_model
Coaxial lighting and viewing	http://www.nerlite.com/DOAL.html
Cold cathode	http://en.wikipedia.org/wiki/Cold_cathode
Cold cathode	http://en.wikipedia.org/wiki/Cold_cathode
Cold cathode	http://www.stockeryale.com/i/fluo/products/panels/p_ml_cold_cathode.htm
Collimated on-axis lighting	http://www.nerlite.com/DOAL.html
Color science	http://mcsl.rit.edu/
Color theory	http://dba.med.sc.edu/price/irf/Ad_tg/color/main.html
Color theory	http://en.wikipedia.org/wiki/Color_theory
Color wheel	http://en.wikipedia.org/wiki/Color_wheel
Colour and light	http://www.sylvania.com/LearnLighting/LightAndColor/ColorandLightRelationship/
Colour characteristics	http://www.sylvania.com/LearnLighting/LightAndColor/LampColorCharacteristics/
Colour concepts	http://www.sylvania.com/LearnLighting/LightAndColor/LightColorCharacteristicf/
Colour measurement	http://www.zeiss.de/C12567BB00549F37?Open
Colour models	http://dba.med.sc.edu/price/irf/Ad_tg/models/main.html
Colour models	http://www.visionbib.com/bibliography/compute97.html
Colour perception	http://dba.med.sc.edu/price/irf/Ad_tg/color/variables.html
Colour perception	http://dba.med.sc.edu/price/irf/Adobe_tg/color/variables.html
Colour recognition	http://home.zcu.cz/~holota5/publ/ae01.pdf
Colour recognition	http://www.experts-exchange.com/Programming/Languages/CPP/Q_23612169.html
Colour recognition	http://www.freepatentsonline.com/EP0033493.html
Colour recognition	http://www.freepatentsonline.com/EP0114515.html
Colour recognition	http://www.highbeam.com/doc/1G1-10911121.html
Colour recognition	http://www.highbeam.com/doc/1G1-115837730.html
Colour recognition	http://www.highbeam.com/doc/1P3-1557469171.html
Colour recognition	http://www.manufacturingtalk.com/news/lrc/lrc101.html

Topic	URL
Colour recognition	http://www.ncbi.nlm.nih.gov/pubmed/9851017?dopt=Abstract
Colour space conversion	http://www.cambridgeincolour.com/tutorials/color-space-conversion.htm
Colourfulness	http://en.wikipedia.org/wiki/Saturation_(color_theory)
Compact fluorescent	http://en.wikipedia.org/wiki/Compact_fluorescent_lamp
Compact fluorescent	http://www.gelighting.com
Complex sources	http://www.lumitex.com/mach_vis_applic_pictures.html
Components sets	http://www.newport.com/Optics-Sets/311771/1033/catalog.aspx
Computer vision homepage	http://www.cs.cmu.edu/~cil/vision.html
Cyclotron radiation	http://en.wikipedia.org/wiki/Cyclotron_radiation
Cylindrical and special lenses	http://www.newport.com/Cylindrical-and-Specialty-Lenses/545083/1033/catalog.aspx
Cylindrical-lenses	http://www.newport.com/Beam-Shaping-with-Cylindrical-Lenses/144888/1033/catalog.aspx
Dark field lighting	http://www.advancedillumination.com/pList.php?cat=71f9f7b3b913e9b7e3d38cd7ebf09e4f
Dark field lighting	http://www.nerlite.com/Dark-Field.html
Dichroic prism	http://en.Wk.org/wiki/Dichroic_prism
Diffraction-gratings	http://www.newport.com/Diffraction-Gratings-Selection-Guide/421120/1033/catalog.aspx
Diffractive optics	http://www.stockeryale.com/i/lasers/accessories/diff_gratings.htm
Diffuse lighting	http://www.advancedillumination.com/pList.php?cat=9f3c3c04f83966f3becd654c257931d7
Digital library	http://spie.org/x2836.xml
Digital photography tutorials	http://www.cambridgeincolour.com/tutorials.htm
Diode laser	http://en.wikipedia.org/wiki/Laser_diode
Dome light	http://www.nerlite.com/Dome.html
Dove prism	http://en.Wk.org/wiki/Dove_prism
Drivers for lighting	http://www.advancedillumination.com/pList.php?cat=f0b609e611129d3a1b9615c1e982ba1d
Electrodeless lamp	http://en.wikipedia.org/wiki/Electrodeless_lamp
Electrodeless lamp	http://www.sylvania.com/LearnLighting/LightAndColor/LightingInnovations/
Electroluminescence	http://en.wikipedia.org/wiki/Electroluminescence
Electroluminescent wire	http://en.wikipedia.org/wiki/Electroluminescent_wire
Ellipsoidal-mirror	http://www.newport.com/Custom-Ellipsoidal-Replicated-Mirror/372498/1033/catalog.aspx
EOS – Home Page	http://www.myeos.org/
Fiber optic gooseneck	http://www.fiberoptix.com/products/goosenecks.html
Fiber optic light source	http://www.fiberoptix.com/products/light-sources.html
Fiberscopes	http://www.fiberoptix.com/products/fiberscopes.html
Fibre optic lighting	http://www.visionlighttech.com
Fibre optic strobe	http://optoelectronics.perkinelmer.com

Topic	URL
Fibre optice. lighting	http://www.visionlighttech.com
Fibre optics	http://www.machinevisiononline.org/public/articles/Volpi_FiberOpticsRevAa.PDF
Field splitter	http://www.lw4u.com/light-works-product-thumbnails.htm
Filters	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=273
Fluorescent	http://en.wikipedia.org/wiki/Fluorescent_lamp
Fluorescent	http://www.gelighting.com
Fluorescent	http://www.stockeryale.com/i/fluo/index.htm
Fluorescent	http://www.sylvania.com/LearnLighting/LightAndColor/FluorescentTechnology/
Fluorescent	http://www.stockeryale.com/i/fluo/faqs.htm
Fluorescent driver	http://www.lighting.philips.com
Fluorescent illumination	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=246
Fluorescent light panel	http://www.stockeryale.com/i/fluo/products/panels/p_ml_series.htm
Fluorescent lighting	http://www.stockeryale.com/i/fluo/products/linears/l_model_13.htm
Fluorescent ring light	http://www.stockeryale.com/i/fluo/circulars.htm
Focusing and collimating	http://www.newport.com/Focusing-and-Collimating/141191/1033/catalog.aspx
Formulas	http://www.newport.com/Optics-Formulas/144956/1033/catalog.aspx
Fresnel lens telecentric	http://www.lw4u.com/light-works-product-thumbnails.htm
Fresnel lens telecentric	http://www.lw4u.com/light-works-product-thumbnails.htm
Fundamental optics	http://optics.mellesgriot.com/optics_guide_1.asp
Gas-discharge lamp	http://en.wikipedia.org/wiki/Gas_discharge_lamp
Geometric optics	http://dev.physicslab.org/asp/applets/opticsbench/default.asp
Geometric optics	http://online.cctt.org/physicslab/content/phyapb/review/summaries/geometricoptics.asp
Geometric optics	http://phet.colorado.edu/simulations/sims.php?sim=Geometric_Optics
Geometric optics	http://www.astro.uvic.ca/~tatum/goptics.html
Geometric optics	http://www.physics.uq.edu.au/people/mcintyre/vergences/optics/geomalt.html
Geometric optics	http://www.sparknotes.com/physics/optics/geom
Germicidal lamp	http://en.wikipedia.org/wiki/Germicidal_lamp
Glasses	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=259
Glossary	http://spie.org/x32276.xml?WT.mc_id=KOPTIPEDIAAE
Glossary	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=297
Glow stick	http://en.wikipedia.org/wiki/Glowsticks
Grow light	http://en.wikipedia.org/wiki/Grow_lamp
Hg medium-arc iodide lamp	http://en.wikipedia.org/wiki/Hydrargyrum_medium-arc_iodide
High intensity discharge	http://www.gelighting.com
High intensity discharge	http://www.sylvania.com/LearnLighting/LightAndColor/HIDTechnology/
High-intensity discharge lamp	http://en.wikipedia.org/wiki/High-intensity_discharge_lamp
Hollow cathode lamp	http://en.wikipedia.org/wiki/Hollow_cathode_lamp
HSB colour model	http://dba.med.sc.edu/price/irf/Ad_tg/models/hsb.html

Topic	URL
Hue	http://en.wikipedia.org/wiki/Hue
Human vision	http://dba.med.sc.edu/price/irf/Ad_tg/color/vision.html
Human vision	http://dba.med.sc.edu/price/irf/Adobe_tg/color/vision.html
Illuminance	http://en.wikipedia.org/wiki/Illuminance
Illumination devices	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=34
Illumination engineering society	http://www.iesna.org/
Image conduit	http://www.us.schott.com
Image conduit	http://www.tegascience.co.jp/products/chi/technotes/image_conduits.htm
Image database	http://staff.science.uva.nl/~aloi/
Image database	http://www.imageprocessingplace.com/root_files_V3/image_databases.htm
Image database	http://www.xtekxray.com/applications/imagelibrary.html
Image databases	http://www-2.cs.cmu.edu/afs/cs/project/cil/www/v-images.html
Image processing fundamentals	http://www.ph.tn.tudelft.nl/Courses/FIP/frames/fip.html
Image processing tutorials	http://www.imageprocessingplace.com/root_files_V3/tutorials.htm
Image processing, books	http://www.imageprocessingplace.com/root_files_V3/publications.htm
Incandescent	http://en.wikipedia.org/wiki/Incandescent_light_bulb
Incandescent	http://www.gelighting.com
Incandescent and halogen	http://www.sylvania.com/LearnLighting/LightAndColor/IncandescentAndHalogen/
Infra-red (IR)	http://en.wikipedia.org/wiki/Infrared
Inteference	http://www.zeiss.de/C12567BB00549F37?Open
Integration of optical systems	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=268
Intensity	http://en.wikipedia.org/wiki/Intensity_(physics)
Interaction of light and matter	http://dba.med.sc.edu/price/irf/Adobe_tg/color/matter.html
Interaction of light and matter	http://dba.med.sc.edu/price/irf/Ad_tg/color/matter.html
Irradiance	http://en.wikipedia.org/wiki/Irradiance
Lambert's cosine law	http://en.Wk.org/wiki/Lambert%27s_cosine_law
Laser	http://en.wikipedia.org/wiki/Laser
Laser diode	http://en.wikipedia.org/wiki/Laser_diode
Laser pattern generators	http://www.stockeryale.com/i/lasers/index.htm
LED lighting	http://www.nerlite.com/Products.html
LED professional	http://www.led-professional.com/
LED strobe driver	http://www.advancedillumination.com/pList.php?cat=dfc88985bde6b7fb0190c1c2778a0564
LEDs	http://www.compumodules.com/image-processing/advanced-illumination
LEDs	http://en.wikipedia.org/wiki/LED
LEDs	http://www.gelighting.com

Topic	URL
LEDs	http://www.stockeryale.com/i/leds/products/cobra.htm
LEDs	http://www.sylvania.com/LearnLighting/LightAndColor/BeyondtheBulb/
Lens edge blackening	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=316
Lens selection guide	http://www.newport.com/Lens-Selection-Guide/140908/1033/catalog.aspx
Library	http://optics.mellesgriot.com/optics_guide.asp
Light	http://en.wikipedia.org/wiki/Category:Light
Light and colour	http://dba.med.sc.edu/price/irf/Adobe_tg/color/light.html
Light and colour ad	http://dba.med.sc.edu/price/irf/Ad_tg/color/light.html
Light bulb types	http://www.lightbulbs-direct.com/?gclid=CJ7z5dO2sZcCFQx3MAodTA7vig
Light guides	http://www.fiberoptix.com/products/light-guides.html
Light meters	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=1726
Light probe kit	http://www.fiberoptix.com/products/light-probe-kit.html
Light shaping diffusers	http://www.newport.com/Light-Shaping-Diffusers/141131/1033/catalog.aspx
Light table	http://www.stockeryale.com/i/fluo/products/panels/p_light_tables.htm
Light tent	http://www.memoryforless.co.uk/store/erol.html
Lighting	http://optoelectronics.perkinelmer.com
Lighting design guide	http://www.lumitex.com/mach_vis_design_guide.html
Lighting science	http://www.sylvania.com/LearnLighting/LightAndColor/TheScienceOfLight/
Lighting selector	http://www.nerlite.com/Selector.html
Line lighting	http://www.advancedillumination.com/pList.php?cat=hjklhjklhjk
Linear lighting array	http://www.advancedillumination.com/pList.php?cat=8a4931574e5d0bc822a3a61e63fd29ee
List of light sources	http://en.wikipedia.org/wiki/List_of_light_sources
Luminous flux	http://en.wikipedia.org/wiki/Luminous_flux
Material properties	http://optics.mellesgriot.com/optics_guide_4.asp
Materials	http://rmico.com/capabilities/materials
Materials	http://www.newport.com/Optical-Materials/144943/1033/catalog.aspx
Mercury-vapor lamp	http://en.wikipedia.org/wiki/Mercury-vapor_lamp
Metal halide lamp	http://en.wikipedia.org/wiki/Metal_halide_lamp
Metallic mirror coatings	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=269
Mirror selection guide	http://www.newport.com/Mirror-Selection-Guide/141086/1033/catalog.aspx
Modulation transfer function	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=317
Movie handling, MATLAB	http://www.ds.eng.monash.edu.au/ece4710/lab2/node8.html
Multi-axis lighting	http://www.nerlite.com/Illuminator.html
Multi-color/white LEDs	http://optoelectronics.perkinelmer.com/
Munsell color system	http://en.wikipedia.org/wiki/Munsell_color_system
Munsell display calculator	http://www.brucelindbloom.com/index.html?MunsellCalculator.html
Neon lamp	http://en.wikipedia.org/wiki/Neon_lamp
NIH image	http://rsb.info.nih.gov/nih-image/download.html
Non-visual imaging	http://www.alienvision.org/Infra red images

Topic	URL
Off-the-shelf integration	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=313
On-line textbook	http://www.lightandmatter.com/area1book5.html
Optical coatings	http://optics.mellesgriot.com/optics_guide_5.asp
Optical design and tolerancing	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=260
Optical materials	http://rmico.com/technical-notes/transmission-curves
Optical rods and tapers	http://fiberoptix.com/products/
Optical specifications	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=252
Optical transmission 1	http://rmico.com/technical-notes/103-transmission-curves
Optical transmission 2	http://rmico.com/technical-notes/zns-cleartran-sapphire-spinel#zns
Optics	http://en.Wk.org/wiki/Optics
Optics application	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=267
Optics calculator	http://www.newport.com/OpticalAssistant/
Optics fundamentals	http://www.newport.com/Optics-Fundamentals/604533/1033/catalog.aspx
Optics glossary	http://www.mellesgriot.com/glossary/wordlist/glossarylist.asp
Optipedia	http://spie.org/x32276.xml?WT.mc_id=KOPTIPEDIAAE
Organic LED	http://en.wikipedia.org/wiki/Organic_light-emitting_diode
Organic LED	http://en.wikipedia.org/wiki/Polymer_light-emitting_diode#PLED
OSA	http://www.osa.org/
Overview	http://www.nerlite.com/Technology.html
Overview	http://www.sylvania.com/LearnLighting/LightAndColor/default.htm
Overview	http://zone.ni.com/devzone/cda/tut/p/id/6901
Parallel windows	http://www.newport.com/Parallel-Windows/141051/1033/catalog.aspx
Pellicle Beamsplitters	http://optical- components.globalspec.com/
Pellin-Broca prism	http://en.Wk.org/wiki/Pellin-Broca_prism
Pentaprism, Wk	http://en.Wk.org/wiki/Pentaprism
Photographic optics	http://www.vanwalree.com/optics.html
Photonics-dictionary	http://www.newport.com/Photonics-Dictionary/604499/1033/catalog.aspx
Photoshop colour management	http://dba.med.sc.edu/price/irf/Ad_tg/ps5/main.html
Plasma lamp	http://en.wikipedia.org/wiki/Plasma_lamp
Point light LED	http://optoelectronics.perkinelmer.com
Polarisation	http://www.newport.com/Polarization/144921/1033/catalog.aspx
Polarisers	http://www.newport.com/Polarization-Optics-Selection-Guide/141146/1033/catalog.aspx
Polarization techniques	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=257
Porro prism	http://en.Wk.org/wiki/Porro_prism
Primary color	http://en.wikipedia.org/wiki/Primary_color
Prisms	http://en.Wk.org/wiki/Prism_%28optics%29
Projector design	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=251
Quartz halogen	http://www.gelighting.com

Topic	URL
Radiant energy	http://en.wikipedia.org/wiki/Radiant_energy
Radiant flux	http://en.wikipedia.org/wiki/Radiant_flux
Radioluminescence	http://en.wikipedia.org/wiki/Radioluminescence
Radiosity (heat transfer)	http://en.wikipedia.org/wiki/Radiosity_(heat_transfer)
Reflecting optics	http://rmico.com/technical-notes/102-reflecting-optics
Replicated mirrors	http://www.newport.com/Replication-Process/374840/1033/catalog.aspx
Replicated optics	http://www.newport.com/Replication-Justification/372497/1033/catalog.aspx
RGB color model	http://en.wikipedia.org/wiki/RGB_color_model
RGB color space	http://en.wikipedia.org/wiki/RGB_color_space
RGB colour cube	http://www.brucelindbloom.com/index.html?WorkingSpaceInfo.html
RGB colour cube	http://www.couleur.org/index.php?page=rgbcube
RGB colour spaces	http://www.babelcolor.com/download/A%20review%20of%20RGB%20color%20spaces.pdf
Ring light	http://www.advancedillumination.com/pList.php?cat=9a3f2406a06ed4c2403f2c04c9f4f668
Ring light	http://www.nerlite.com/Ring.html
RYB color model	http://en.wikipedia.org/wiki/RYB_color_model
Sapphire windows	http://www.newport.com/Sapphire-Windows/378626/1033/catalog.aspx
Scintillation	http://en.wikipedia.org/wiki/Scintillation_(physics)
Scotchlite	http://en.Wk.org/wiki/Scotchlite
Sealed lighting unit	http://www.stockeryale.com/i/fluo/products/task_lights/sealed_light.htm
Self-powered lighting	http://en.wikipedia.org/wiki/Self-powered_lighting
Sodium vapor lamp	http://en.wikipedia.org/wiki/Sodium_vapor_lamp
Solid-state lighting	http://en.wikipedia.org/wiki/Solid-state_lighting
Sonoluminescence	http://en.wikipedia.org/wiki/Sonoluminescence
Spatial-filters	http://www.newport.com/Spatial-Filters/144910/1033/catalog.aspx
Spectra	http://ioannis.virtualcomposer2000.com/spectroscope/amici.html
Spectrometers	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=1725
Spectrum	http://en.wikipedia.org/wiki/Spectrum
Spectrum	http://www.sylvania.com/LearnLighting/LightAndColor/SpectralPowerDistribution/
Specular and diffuse reflection	http://www.glenbrook.k12.il.us/gbssci/phys/class/refln/u13l1d.html
Spherical mirrors	http://www.du.edu/~jcalvert/optics/mirror.htm
SPIE – Home Page	http://spie.org/
Spot lights	http://www.advancedillumination.com/pList.php?cat=2c5aa011aa688eab3b4f1e69581cfb71
Standard iluminates	http://www.zeiss.de/C12567BB00549F37?Open
Standard observer	http://www.zeiss.de/C12567BB00549F37?Open
Substrates	http://rmico.com/technical-notes
Subtractive color	http://en.wikipedia.org/wiki/Subtractive_color

Topic	URL
Sulfur lamp	http://en.wikipedia.org/wiki/Sulfur_lamp
Sunlight	http://en.wikipedia.org/wiki/Sunlight
Supercontinuum generation	http://www.rp-photonics.com/supercontinuum_generation.html
Synchrotron light	http://en.wikipedia.org/wiki/Synchrotron_light
Synchrotron radiation	http://en.wikipedia.org/wiki/Synchrotron_radiation
Tanning lamp	http://en.wikipedia.org/wiki/Tanning_lamp
Technical library	http://optics.mellesgriot.com/optics_guide.asp
Technical literature	http://www.mellesgriot.com/products/technicalliterature.asp#optics
Telecentric lenses	http://www.lw4u.com/light-works-product-thumbnails.htm
Telecentric lenses	http://www.mellesgriot.com/resourcelibrary/specsheets/machinevision/default.asp
Temperature sensitive sheet	http://www.edmundoptics.com/onlinecatalog/displayproduct.cfm?productID=1642
Test targets	http://www.edmundoptics.com/onlinecatalog/browse.cfm?categoryid=289
Thermal radiation	http://en.wikipedia.org/wiki/Blackbody_radiation
Transmitting optics	http://rmico.com/technical-notes/transmitting-optics
Triangular prism	http://en.Wk.org/wiki/Triangular_prism_%28optics%29
Ultra-violet (UV)	http://en.wikipedia.org/wiki/Ultraviolet
Ultra-violet images	http://imageevent.com/austinrichards/infrared/aic
Units	http://www.sylvania.com/LearnLighting/LightAndColor/TheTechnologyofLight/
Using optical flats	http://www.edmundoptics.com/techSupport/DisplayArticle.cfm?articleid=254
View splitter	http://www.lw4u.com/light-works-product-thumbnails.htm
VirtualLab	http://www.lighttrans.com/starter_toolbox.html?gclid=COTezeDPyZ8CFcZe4wodz03QzQ
White reflectance coating	http://www.edmundoptics.com/onlinecatalog/displayproduct.cfm?productID=1325
Xenon arc lamp	http://en.wikipedia.org/wiki/Xenon_arc_lamp
Xenon arc lamp	http://en.wikipedia.org/wiki/Xenon_flash_lamp
X-ray images	http://www.xtekxray.com/applications/imagelibrary.html

I.2.2 Image Processing Software

Many of the programs/libraries listed below are intended non-industrial applications, such as medicine, astronomy etc.

The following sites provide numerous links to commercial and academic software packages for image processing:

http://www.roborealm.com/

http://www.cs.cmu.edu/~cil/v-source.html

(Note that *SUSAN* is not the same as *SUSIE*, the precursor of *QT*.)

Name	URL
ADISL	http://home.iitk.ac.in/~rksr/adisl.html
AForge	http://www.aforgenet.com/
AllSeeingI	http://www.uni-koblenz.de/~jesus/asi/index.php
Blepo	http://www.ces.clemson.edu/~stb/blepo/
CamCap & CLAG	http://www.cs.nott.ac.uk/~jzg/nottsvision/index.html
CImg	http://cimg.sourceforge.net/
CMVision	http://www-2.cs.cmu.edu/~jbruce/cmvision/
CVIPtools	http://www.ee.siue.edu/CVIPtools/
DigiFlow	http://www.damtp.cam.ac.uk/lab/digiflow/
DIPImage	http://www.diplib.org/
easyVision	http://www.easyvision.googlepages.com/
EDISON	http://www.caip.rutgers.edu/riul/research/code/EDISON/index.html
eLynx	http://sourceforge.net/projects/elynx/
eVisionEgine	http://www.braintech.com/products-eve.php
FILTERS	http://filters.sourceforge.net/
Fly Capture	http://www.ptgrey.com/
FreeImage	https://code.launchpad.net/~glennpierce/+junk/main
Gamera	http://gamera.informatik.hsnr.de/
Gandalf	http://gandalf-library.sourceforge.net/tutorial/report/report.html
GIMP OCTAVE	http://sites.google.com/site/elsamuko/gimp/gimp-octave
GIMP	http://www.gimp.org/
G'Mic	http://gmic.sourceforge.net/
Gwyddion	http://gwyddion.net/
Hornets Eye	http://www.wedesoft.demon.co.uk/hornetseye-api/files/HornetsEye-txt.html
iLab	http://ilab.usc.edu/toolkit/home.shtml
Image processing software	http://www.imageprocessingplace.com/root_files_V3/software/software.htm
Image SXM	http://www.liv.ac.uk/~sdb/ImageSXM/
ImageJ	http://rsb.info.nih.gov/ij/
Imalab	http://www-prima.imag.fr/Imalab/
ImLib3D	http://sourceforge.net/projects/imlib3d/
IMPROMPTU	http://www.i-clic.uihc.uiowa.edu/
IPL	http://developer.intel.com/software/products/perflib/ipl/index.htm
Java Advanced Imaging, JAI	http://java.sun.com/javase/technologies/desktop/media/jai/
JHLabs	http://www.jhlabs.com/ie/index.html
LABVIEW	http://www.ni.com/labview/family/image_signal_processing.htm
Leptonicay	http://www.leptonica.com/local-sources.html
Lispix	http://www.nist.gov/lispix/doc/contents.htm
LTI-Lib	http://ltilib.sourceforge.net/doc/homepage/index.shtml

Name	URL
MATLAB IP Toolbox	http://download.cnet.com/Image-Processing-Toolbox-for-Matlab/3000-2070_4-10832545.html
MATLAB IP Toolbox	http://ee.sharif.edu/~dip/Files/MatlabDIPinBrief.pdf
MATLAB IP Toolbox	http://search.techrepublic.com.com/search/matlab+image+processing+toolbox.html
MATLAB IP Toolbox	http://users.soe.ucsc.edu/~htakeda/KernelToolBox.htm
MATLAB IP Toolbox	http://www.dcs.gla.ac.uk/~wpc/diP/matlabip.htm
MATLAB IP Toolbox	http://www.eng.auburn.edu/~sjreeves/Classes/IP/IP.html
MATLAB IP Toolbox	http://www.imageprocessingplace.com/DIPUM-1E/dipum1e_main_page.htm
MATLAB IP Toolbox	http://www.petercorke.com/Machine_Vision_Toolbox.html
MATLAB IP Toolbox	http://www.softpedia.com
MATLAB IP Toolbox	http://en.wikibooks.org/wiki/MATLAB_Programming/Image_Processing_Toolbox
MATLAB IP Toolbox	http://homepages.inf.ed.ac.uk/rbf/HIPR2/impmatl.htm
MATLAB IP Toolbox	http://users.rsise.anu.edu.au/~luke/cvcourse_files/labs/Lab1.pdf
MATLAB IP Toolbox	http://vision.ucsd.edu/~pdollar/toolbox/doc/
MATLAB IP Toolbox	http://www.cogs.susx.ac.uk/users/davidy/compvis/matlab_demos/intro_demo.html
MATLAB IP Toolbox	http://www.ee.oulu.fi/mvmp/courses/mv/matlab/guide/matlab_guide.pdf
MATLAB IP Toolbox	http://www.eng.auburn.edu/~sjreeves/Classes/IP/IP.html
MATLAB IP Toolbox	http://www.mathworks.com/products/image/
MATLAB IP Toolbox	http://www.soft-go.com/view/Image-Processing-Toolbox-for-Matlab_68526.html
MATLAB IP Toolbox	http://www.uib.no/med/avd/miapr/arvid/MOD3_2002/Matlab/ip_getting_started_1.html
MATLAB ref. documentation	http://www.math.ufl.edu/help/matlab/ReferenceTOC.html
MegaWave	http://megawave.cmla.ens-cachan.fr/
MIRIAD	http://bima.astro.umd.edu/miriad/
Motion	http://www.lavrsen.dk/twiki/bin/view/Motion/WebHome
MPTx	http://mplab.ucsd.edu/grants/project1/free-software/MPTWebSite/introduction.html
NI Machine Vision Software	http://zone.ni.com/
NIH IMAGE	http://rsb.info.nih.gov/nih-image/
OCTAVE	http://www.csse.uwa.edu.au/~pk/Research/MatlabFns/
OpenVidia	http://openvidia.sourceforge.net/index.php/OpenVIDIA
Peter Kovesi	http://www.csse.uwa.edu.au/~pk/Research/MatlabFns/
QVision	http://perception.inf.um.es/QVision/
RAVL	http://ravl.sourceforge.net/
RoboRealm	http://www.roborealm.com/
RobotVisionCAD	http://www.robotvision2.com/Softwares/RvCAD/RvCAD.html
SPIDER	http://www.wadsworth.org/spider_doc/spider/docs/spider.html
SUSAN	http://users.fmrib.ox.ac.uk/~steve/susan/index.html

Name	URL
SVS	http://users.rcn.com/mclaughl.dnai/support.htm
Synios	http://www.synios.com/
TargetJr	http://www.targetjr.org
TINA	http://www.tina-vision.net/
tnimage	http://brneurosci.org/tnimage-manual/tnimage-manual.html
Univ of Calgary	http://pages.cpsc.ucalgary.ca/~parker/soft.htm
UNL	http://www.inf.ufes.br/~thomas/home/soft.html
UTHSCSA	http://ddsdx.uthscsa.edu/dig/itdesc.html
VIGRA	http://hci.iwr.uni-heidelberg.de/vigra/
VIP Base	http://vipbase.net/index.html#highlight
Vision Blox	http://en.commonvisionblox.de/en/pages/cvb/main.php
VisiQuest (aka*Khoros*)	http://www.accusoft.com/
VXL	http://vxl.sourceforge.net/
WIT	http://www.dalsa.com/mv/support/wit/wit.aspx
XVision	http://www.cs.jhu.edu/CIPS/xvision/

Applications: Industry

A Better View of Manufacturing (2007) Diesel Progr North Am Ed 73(8):77–78, ISSN: 1091370X

A Less-Elusive Rail Flaw (2008) RT and S: Railway track and structures, 104(2):21–23, ISSN: 00339016, ISSN: 00339016

Abdou G (1995) Automated recognition of multiple features of stamped sheet. Am Soc Mech Eng Manuf Eng Div MED 1:319–327

Abdullah MZ, Guan LC, Mohamed AMD, Noor MAM (2002) Color vision system for ripeness inspection of oil palm elaeis guineensis. J Food Proces Preserv 26(3):213–235, ISSN: 01458892

Abou-El-Ela MS, El-Amroussy H (1996) Machine vision system for the recognition and positioning of two-dimensional partially occluded objects. Proceedings of the Mediterranean Electrotechnical Conference – MELECON, 2, pp 1087–1092

Adam GK, Mastorakis NE (2002) Programming and control of an industrial robotic system. Recent advances in circuits, systems and signal processing, pp 425–430, ISBN: 9608052645

Adamo F, Attivissimo F, Di Nisio A, Savino M (2009) An online defects inspection system for satin glass based on machine vision. IEEE Intrumentation and Measurement Technology Conference, I2MTC 2009, art. no. 5168461, pp 288–293, ISBN: 9781424433537

Adams L (2004) Manipulating images made easier. Adv Imag 19(10):19–22, ISSN: 10420711

Adar R, Akerib A (1997) Associative architecture for image processing. Proc SPIE Int Soc Opt Eng 3166:238–246, ISSN: 0277786X

Adaway B (1987) Machine vision in practice. Prod Eng Lond 66(11):20–21, ISSN: 00329851

Adaway WGL, Thomas WV (1985) Hardware architectures for industrial vision systems. IEE Colloquium (Digest) (1985/24), pp 4.1–4.2, ISSN: 09633308

Advances in Materials Manufacturing Science and Technology XIII: Advanced Manufacturing Technology and Equipment, and Manufacturing Systems and Automation (2009) Materials Science Forum, 626 627, 814 p, ISSN: 02555476, ISBN: 0878493115; 9780878493111

Agapakis JE (1999) Look into machine-vision technology. EE Eval Eng 38(5):4, ISSN: 01490370

Agapakis JE, Masubuchi K (1985) Novel uses of machine vision in automated robotic welding fabrication, pp 109–110; Anon (1985) AUTOFACT '85 – Conference proceedings, AUTOFACT, Conference Proceedings, pp var paging, ISSN: 02555476, ISBN: 0878493115; 9780878493111

Agapakis JE, Bolstad J (1991) Vision sensing and processing for monitoring and control of welding and other high luminosity processes. Proc SPIE Int Soc Opt Eng 1385:32–38, ISSN: 0277786X

Al-Eidarous M (1998) Locating defects on shirt collars using image processing. Int J Cloth Sci Technol 10(5):365–378, ISSN: 09556222

Allada V, Anand S (1996) Efficient vertex detection algorithms using the hough transform. Int J Adv Manuf Technol 11(6):394–405, ISSN: 02683768

Allada V, Huffer N, Anand S (1993) Quad and octree structures for tolerance representation in solid models. Proceedings of the industrial engineering research conference, pp 461–465, ISBN: 0898061326

Allada V, Anand S, Chu Y-C (1994) Intelligent CNC cutting of sheet metal parts using machine vision. Int J Ind EngTheory Appl Pract 1(4):305–314, ISSN: 10724761

Allcock A (2006) More vision in manufacturing. Machinery 164(4122):63, ISSN: 0024919X

Allred RE, Wesson SP, Babow DA (2005) Real-time control of powder towpreg production using embedded processing and machine vision. International SAMPE Symposium and Exhibition (Proceedings), 50, pp 2057–2071, ISSN: 08910138

Alrad has a vision of print and packaging (2001) Packaging Magazine 4(19):26, ISSN: 02676117

American Society of Agricultural and Biological Engineers – Food Processing Automation Conference 2008 (2008) American society of agricultural and biological engineers – Food processing automation conference 2008, 235 p, ISBN: 9781605607900

Amin N, Khadem MS (2007) Interface development for cost effective automated IC orientation checking systems, 2007. 10th international conference on computer and information technology, ICCIT, art. no. 4579429, ISBN: 1424415519; 9781424415519

An L, Wang Z-Q, Lu J-R (2007) Calculating the waveguide invariant by passive sonar lofargram image. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430706, pp 13–17, ISBN: 1424413583; 9781424413584

Anand S, Raman S, Wysk RA (1988a) An algorithm for vertex detection. Comput Ind Eng 14(2):77–83, ISSN: 03608352

Anand S, Raman S, Wysk RA (1988b) Vision assisted NC milling path generation. J Manuf Syst 7(3):233–240, ISSN: 02786125

Andres NS, Marimuthu RP, Eom, Y-K, Jang B-C (2005) Development of a machine vision system for automotive part inspection. Proc SPIE Int Soc Opt Eng 6041, art. no. 60412J, ISSN: 0277786X

Anon (1983) IEEE IECON '83 proceedings, annual conference on industrial electronics, 347 p; West PC (1983) Machine vision in practice. IEEE Trans Indust Appl IA-19(5):794–801, ISSN: 00939994

Anon (1983b) Seeing eye robots for the automated factory. Prod Eng 30(8):48–51, ISSN: 01461737

Anon (1984) Utilization of artificial intelligence and pattern recognition techniques in manufacturing engineering: 16TH CIRP international seminar on manufacturing systems, 247 p; Wisnosky DE (1984) Computer integrated manufacturing (CIM) – The future is now. Technical paper – Society of manufacturing engineers, 16 p

Anon (1984) Third annual applied machine vision conference proceedings, pp var paging, ISBN: 0872631400

Anon (1984) Robots 8, conference proceedings, pp var paging, ISBN: 0872631478

Anon (1984d) Machine vision: A sense for CIM. Am Mach 128(6):101–120, ISSN: 00029858

Anon (1984e) Vision systems – Bringing sight to automation. Modern Mater Handl 39(6):64–69, ISSN: 00268038

Anon (1985) IEEE 1985 proceedings of the international conference on cybernetics and society, 1115 p; Daum RC (1985) Fiber optic lighting for machine vision, pp 4.1–4.13, ISBN: 0872631737

Anon (1985) Conference proceedings – vision '85, pp var paging, ISBN: 0872631737

Anon (1986) Robots build new markets. Engineering (Lond) 226(7–8):524–525, 527, ISSN: 00137782

Anon (1986) Proceedings – 1986 IEEE international conference on robotics and automation, 2051 p, ISBN: 0818606959

Anon (1986a) IEEE workstation technology and systems conference – Proceedings, 143 p, ISBN: 0818606932

Anon (1986d) Automated fish inspection system. Robot Eng 8(6):19, ISSN: 08880816

Anon (1986) Proceedings of the 1986 IEEE international conference on systems, man, and cybernetics, 1630 p; Cowan DA, Davies RC (1986) Next generation of flexible electronic assembly, pp 375–383, ISBN: 0948507098

Anon (1986) Vision '86 – Conference proceedings, pp var. paging, ISBN: 0872632296, ISBN: 0872632296

Anon (1986g) Vision: Finding a place in the factory. Prod Eng 33(1):54–58, 60 ISSN: 01461737

Anon (1987) Proceedings – IECON '87: 1987 International conference on industrial electronics, control, and instrumentation, 1200 p

Anon (1987) Future transportation technology conference and exposition. SAE Technical Paper Series, pp var. paging

Anon (1987c) Laser integration improves welding. Robot World 5(1):26–27, ISSN: 07377908

Anon (1987d) Machine vision in tomorrow's factory. Prod Eng Lond 66(3):14–15, ISSN: 00329851

Anon (1987e) Vision systems: Eyes for robots. Assembly Eng 30(8):14–16, ISSN: 00045063

Anon (1987f) X-ray inspections yield tireless quality control. Robot World 5(4):54–55, ISSN: 07377908

Anon (1987) Third IEEE/chmt international electronic manufacturing technology symposium, 257 p; Anon (1987) Future transportation technology conference and exposition, SAE technical paper series, pp var paging

Anon (1988) Imaging requirements of an underwater autonomous vehicle. Datacube World Rev 2(2):1

Anon (1989) Resin quality realized with on-line 'machine vision'. InTech 36(12):38–39, ISSN: 0192303X

Anon (1998) Bold applications demonstrate vision's prowess. Robot World 16(2):35–38, ISSN: 07377908

Applications of Artificial Intelligence IV (1986) Proc SPIE Int Soc Opt Eng 657:209, ISSN: 0277786X, ISBN: 0892526920

Armingol JM, Otamendi J, De La Escalera A, Pastor JM, Rodriguez FJ (2003) Statistical pattern modeling in vision-based quality control systems. J Intel Robot Syst Theory Appl 37(3):321–336, ISSN: 09210296

Asoudegi E (1992) Quantitative automated inspection of standard parts using machine vision. Comput Ind Eng 23(1–4):361–364, ISSN: 03608352

At Speeds of Light (2003) Canadian Packaging 56(12):S2, ISSN: 00084654

Atiquzzaman M (1999) Coarse-to-fine search technique to detect circles in images. Int J Adv Manuf Technol 15(2):96–102, ISSN: 02683768

Ault SK, Strand OT, Lowry ME (1996) Computer vision based machine for automated packaging of photonics components. Conference proceedings – lasers and electro-optics society annual meeting-LEOS, 2, pp 238–239, ISSN: 10928081

Automatic Inspection and Novel Instrumentation (1997) Proc SPIE Int Soc Opt Eng 3185:161 p, ISSN: 0277786X

Automatica Leads the Way (2006) Assembly 49(2):66–70, ISSN: 10508171

Ayres RU, Funk JL (1989) The role of machine sensing in CIM. Rob Comput Integr Manuf 5(1):53–71, ISSN: 07365845

Azar I, Weston RH (1987) Integrating vision systems in CIM. Comput Aided Eng J 4(6):227–232, ISSN: 02639327

Azar I, Weston RH (1988) On vision architecture for computer integrated manufacturing. Microprocess Microsyst 12(1):24–32, ISSN: 01419331

Azzam SJ (1986) Automatic robotic workcell programming for production of satellite equipment shelves, pp 13.33–13.44, ISBN: 087263258X

Babb M (1994) 'Ease of use' is key to new machine vision system. Control Eng 41(3):56–58, ISSN: 00108049

Babb M (1995) Machine vision moves into the mainstream of manufacturing. Control Eng 42(8):3, ISSN: 00108049

Babb M (1996) Machine vision: High-tech success on the factory floor. Control Eng 43(10):3, ISSN: 00108049

Babb M (2009) Single point machine vision. Control Eng 56(10), ISSN: 00108049

Bailey-Van Kuren M (2003) Vision based manifolds for surface modeling in robotic demanufacturing. Proceedings of the IASTED international conference on modelling and simulation, pp 632–637, ISBN: 0889863377

Bailey-Van Kuren MA (2005) demanufacturing projector-vision system for combined manual and automated processing of used electronics. Comput Ind 56(8–9):894–904, ISSN: 01663615

Bailey E, Tormey ES (2007) Achieving high CRI from warm to super white. Proc SPIE Int Soc Opt Eng 6669, art. no. 66690E, ISSN: 0277786X, ISBN: 9780819468178

Bakhadyrov I, Jafari MA, Fang T, Safari A, Danforth S, Langrana N (1998) Defect detection in indirect layered manufacturing. Proc IEEE Int Conf Syst Man Cybernet 5:4251–4254, ISSN: 08843627

Banks D (1984) Machine vision – The link between fixed and flexible automation, pp 21–25, ISBN: 0903608758

Banta L, Pertl F, Rosenecker C, Rosenberry-Friend K (1998) Machine vision-based bar code scanning for long range applications. Proc SPIE Int Soc Opt Eng 3517:130–138, ISSN: 0277786X

Barrett III JG, Bishop Jr AO (1991) Image transfer on an IEEE 802.4 network, conference proceedings – IEEE southeastcon, 1, pp 550–552, ISSN: 07347502, ISBN: 0780300335

Barth A, Herpers R, Greßnich M (2007) Real-time applicable visual quality control in industrial line production. IEEE/ASME international conference on advanced intelligent mechatronics, AIM, art. no. 4412499, , ISBN: 1424412641; 9781424412648

Bartlam P (1982) Linescan camera inspection systems in industrial applications, IEE Colloquium (Digest) (1982/77), pp 8/1–8/4, ISSN: 09633308

Barton J (2008) Quality through automation. Int Dyer 193(6):23–24+26–27, ISSN: 0020658X

Bartos FJ (1987) Automated manufacturing and on-line testing team up for product quality. Control Eng 34(12):64–66, ISSN: 00108049

Bartos FJ (1998) Motion control sails into a sea of applications. Control Eng 45(10):3, ISSN: 00108049

Batchelor BG (1999) Characterizing the illumination in an automated visual inspection work cell. Proc SPIE Int Soc Opt Eng 3836:134–143, ISSN: 0277786X

Batchelor BG, Hack R (1995) Robot vision system programmed in Prolog. Proc SPIE Int Soc Opt Eng 2597:239–252, ISSN: 0277786X, ISBN: 0819419613

Batchelor BG, Whelan PF (1995) Ethical, environmental and social issues for machine vision in manufacturing industry. Proc SPIE Int Soc Opt Eng 2597, pp 2–15, ISSN: 0277786X, ISBN: 0819419613

Batchelor BG, Daley MW, Karantalis G (2001) Teleprototyping environment for machine vision. J Electr Imag 10(1):316–328, ISSN: 10179909

Batchelor BG, Caton SJ, Chatburn LT, Crowther RA, Miller JWV (2005) Vision systems on the Internet.

Proc SPIE Int Soc Opt Eng 6000, art. no. 600003, ISSN: 0277786X

Bauchert KA, Serati SA, Sharp GD, McKnight DJ (1997) Complex phase/amplitude spatial light modulator advances and use in a multispectral optical correlator. Proc SPIE Int Soc Opt Eng 3073:170–177, ISSN: 0277786X, ISBN: 0819424889

Baumann EW (1981) Model based vision and the mcl language. Proceedings – International conference on cybernetics and society, pp 433–438, ISSN: 01636006

Behera BK (2003) Image-processing in textiles. Text Prog 35(2–4):195, ISSN: 00405167

Belbachir AN, Litzenberger M, Posch C, Schön P (2007) Real-time vision using a smart sensor system. IEEE international symposium on industrial electronics, art. no. 4374909, pp 1968–1973, ISBN: 1424407559; 9781424407552

Bell I (2005) The future of control. Manuf Eng 84(4):36–39, ISSN: 09569944

Benes J (2005) 20/20 machine vision. Am Mach 149(3):38–42, ISSN: 10417958

Beng-Hoe A (1998) On-line inspection of circular fluorescent lamp. Proc SPIE Int Soc Opt Eng 3521:229–236, ISSN: 0277786X

Berg DM (1997) Optical tooling for flexible manufacture. Proc SPIE Int Soc Opt Eng 3205:9–15, ISSN: 0277786X

Berger A, Dunbar P, Roberts C (1985) Machine vision character recognition in the electronics packaging industry. Three case studies, pp 1.16–1.35, ISBN: 0872631737

Berk DA, Judd FF, Wisniewski II, Stanley C (1988) Machine-vision-assisted workstations for lightwave device manufacture. AT&T Tech J 67(2):35–46, ISSN: 87562324

Berkstresser III GA, Buchanan DR (1986) Automation and robotics in the textile and apparel industries. Automation and robotics in the textile and apparel industry, 328 p, ISBN: 0815510772

Berkstresser GA, Buchanan DR (1986) Automation and robotics in the textile and apparel industries, ISBN: 0815510772

Bertuolo A, Diani M, Colet P (2003) Quality packaging; rapid ROI. Adv Imag 18(1):20–21, ISSN: 10420711

Berube J-N, Dupont P (1999) PC takes on major industrial vision tasks. Technical paper – Society of manufacturing engineers. MS (MS99-266), pp 1–8, ISSN: 01616382

Beving JE (1992) Application of "off-the-shelf" hardware for machine vision. Proc SPIE Int Soc Opt Eng 1778:122–127, ISSN: 0277786X, ISBN: 0819409502

Bhanu B, Ho C-C (1986) Computer-aided geometric design based 3-d models for machine vision. Proceedings – International conference on pattern recognition, pp 107–110, ISBN: 0818607424

Bhuvanesh A, Ratnam MM (2007) Automatic detection of stamping defects in leadframes using machine vision: Overcoming translational and rotational misalignment. Int J Adv Manuf Technol 32(11–12):1201–1210, ISSN: 02683768

Bhuvanesh A, Lam SSY (2006) Defect classification for leadframe manufacturing using artificial neural networks. IIE annual conference and exhibition, 6 p; Otieno A, Mirman C (2006) Machine vision applications in plastics injection molding: A case study of automated inspection. J Eng Technol 23(1):26–31, ISSN: 07479964

Bieman LH (1990) Coordinate mastering using optical coupling. Ind Metrol 1(2):101–110, ISSN: 09215956

Billingsley J (2007) Vision applications in agriculture. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430754, pp 6–7, ISBN: 1424413583; 9781424413584

Billingsley J (2007) Bovine Intelligence for training horses. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430708, pp 23–27, ISBN: 1424413583; 9781424413584

Billingsley J, Dunn M (2005) Unusual vision – Machine vision applications at the NCEA. Sensor Rev 25(3):202–208, ISSN: 02602288

Bilton T (2007) Machine vision lens selection makes the difference. Control Eng 54(12):IM1–IM10, ISSN: 00108049

Birenbaum RI (1991) Getting started with machine vision. Test Measure World 11(2):4, ISSN: 07441657

Birgmajer B, Kovačić Z, Postružin Z (2008) Integrated vision system for supervision and guidance of a steam generator tube inspection manipulator. Proceedings of the IEEE international conference on control applications, art. no. 4776721, pp 644–649; Sun J, Zhang Z (2008) One method of plane direction measurement by using of the grating fringe. Proceedings – 5th international conference on fuzzy systems and knowledge discovery, FSKD 2008, 4, art. no. 4666363, pp 91–95, ISBN: 9780769533056

Bitz K (2003) Defect detectors. Nonwovens Industry 34(2):50–56, ISSN: 01634429

Blackwell GF (1989) Machine vision in the tire industry. IEEE conference record of annual conference of electrical engineering problems in the rubber and plastics industry, pp 67–79, ISSN: 02724685

Blanz WE, Sanz JLC, Hinkle EB (1987) Image analyis methods for solder ball inspection in integrated circuit manufacturing, pp 509–514, ISBN: 0818607874

Blanz WE, Sanz JLC, Hinkle EB (1988) Image analysis methods for solder-ball inspection in integrated circuit manufacturing. IEEE J Robot Autom 4(2): 129–139, ISSN: 08824967

Blasco J, Gómez-Sanchís J, Gutierrez A, Chueca P, Argilés R, Moltó E (2009) Automatic sex detection of individuals of *Ceratitis capitata* by means of computer vision in a biofactory. Pest Manag Sci 65(1):99–104, ISSN: 1526498X

Blase WP (1987) Laboratory peripheral control system for the numerical stereo camera. IEEE proceedings of the national aerospace and electronics conference, pp 1528–1540

Bodenstab JC (1985) Industrial uses of machine vision technology. Proceedings of the annual control engineering conference, pp 314–316, ISBN: 091433154X

Bodenstab J (1986) Machine vision for electronics manufacturing. Robot Eng 8(9):2l–25, ISSN: 08880816

Boglea A, Bou S, Olowinsky A, Almansa A (2006) High accuracy laser based microassembly using a machine vision alignment system. Proceedings of the IASTED international conference on intelligent systems and control, pp 19–23, ISSN: 10258973

Boissenin M, Wedekind J, Selvan AN, Amavasai BP, Caparrelli F, Travis JR (2007) Computer vision methods for optical microscopes. Image Vis Comput 25(7):1107–1116, ISSN: 02628856

Bolhouse VC (1985) Machine vision applications for high volume electronics manufacturing, pp 5.75–5.98, ISBN: 0872631737

Bolhouse V (2007) Why not automotive? Adv Imag 22(9):20–26, ISSN: 10420711

Boo K, Yook H, Lee J, Che W-S (2006) A study on identification of bubble movements in an automatic wheel leakage detection system. Key Eng Mater 326–328 I:453–456, ISSN: 10139826

Borah S, Bhuyan M (2003) Quality indexing by machine vision during fermentation in black tea manufacturing. Proc SPIE Int Soc Opt Eng 5132:468–475, ISSN: 0277786X

Borangiu T, Gilbert P, Ivanescu N-A, Rosu A (2009) An implementing framework for holonic manufacturing control with multiple robot-vision stations. Eng Appl Artif Intell 22(4–5):505–521, ISSN: 09521976

Boukouvalas C, Kittler J, Marik R, Petrou M (1997) Automatic color grading of ceramic tiles using machine vision. IEEE Trans Ind Electron 44(1):132–135, ISSN: 02780046

Bourn CT (1996) Machine vision provides 100% inspection of connector pins plus real-time charting and statistics. Technical paper – Society of manufacturing engineers. MS, (MS96-268), 9 pp, ISSN: 01616382

Bowskill J, Katz T, Cattez D (1994) Object oriented approach to a machine vision framework. IEE Colloquium (Digest) (3):6/1–6/3, ISSN: 09633308

Bowskill JM, Katz T, Downie JH (1995) Solder inspection using an object-oriented approach to machine vision. Proc SPIE Int Soc Opt Eng 2423:34–45, ISSN: 0277786X, ISBN: 081941770X

Bozchalooi IS, Liang M (2007) Identification of the high SNR frequency band for bearing fault signature enhancement. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430711, pp 39–43, ISBN: 1424413583; 9781424413584

Bradbeer RS, Hodgson P, Lam K, Ku KKK (2007) Establishment of a three dimensional, real-time marine environment monitoring system, DataBuoy™ I, in the Hoi Ha Wan Marine Park, Hong Kong, with connection to the internet. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430738, pp 174–178, ISBN: 1424413583; 9781424413584

Bradley C (2001) Rapid prototyping models generated from machine vision data. Comput Ind 44(2):159–173, ISSN: 01663615

Bradley C, Kurada S (1995) Industrial inspection employing a three dimensional vision system and a neural network classifier. IEEE Pacific RIM conference on communications, computers, and signal processing – Proceedings, pp 505–508

Bradshaw M (1995a) The application of machine vision to the automated inspection of knitted fabrics. Mechatronics 5(2–3):233–243, ISSN: 09574158

Bradshaw M (1995b) Delineation of defects in noisy ternary images using a piecewise dynamic approach. Proc SPIE Int Soc Opt Eng 2423:334–342, ISBN: 081941770X

Braggins D (1983) Automated visual inspection – The technology exists. Prod Eng Lond 62(3):21–22, ISSN: 00329851

Braggins D (1987) Machine vision flexes its muscles, CME. Chartered Mech Eng 34(7–8):38–41, ISSN: 03069532

Braggins D (1996) Checking on machine vision markets – Orlando FL to Birmingham UK. Adv Imag 11(4): 67–X12, ISSN: 10420711

Braggins D (2000a) Machine vision research in the UK: Grants for markets beyond industrial. Adv Imag 15(8):56–57, ISSN: 10420711

Braggins D (2000b) Vision and statistical process control in the integrated factory: Upstream or downstream? Adv Imag 15(9):30–31, ISSN: 10420711

Braggins D (2001a) Imaging on display in France and the UK. Adv Imag 16(5):18–21, ISSN: 10420711

Braggins D (2001b) Imaging sensors at IPOT/MV/MTEC. Sensor Rev 21(3):199–203, ISSN: 02602288

Braggins D (2002) Space and asparagus at photonics Boston 2001. Sensor Rev 22(2):134–138, ISSN: 02602288

Braggins D (2004) Machine vision companies widen their scope. Adv Imag 19(9):28–30, +43, ISSN: 10420711

Braggins D (2005) Old and new at automatica exhibition. Sensor Rev 25(1):14–16, ISSN: 02602288

Braggins D (2006a) Seeing is believing. Engineering 247(5):60–61, ISSN: 00137782

Braggins D (2006b) Vision today for assembly automation. Assembly Autom 26(3):181–183, ISSN: 01445154

Braggins D (2007) Complete vision. Engineering 248(5):59–60, ISSN: 00137782

Braggins D (2009) The next landmark in vision. Eng Technol 4(8):40–42, ISSN: 17509637

Brandstetter R, Fonneland N, Zanella R, Yearwood M (1991) Optical correlator vision system for a manufacturing robot assembly cell. Proc SPIE Int Soc Opt Eng 1385:173–189, ISSN: 0277786X

Breen JM (1985) Force sensing or vision: Weighing the advantages. Robot Today 7(2):35–36, ISSN: 01936913

Brennan MA (1984) Machine vision and robotic testing, pp 6.1–6.6, ISBN: 0872631400

Brett PN (2007) Moving on from surgical robotics to robotic micro-tools in surgery. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430753, pp 1–5, ISBN: 1424413583; 9781424413584

Brett PN, Ma X, Holding DJ, Elliott MT, Petra I (2007) Real-time tracking of a moving contacting load using the distributive tactile sensing method. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430731, pp 136–139, ISBN: 1424413583; 9781424413584

Brosnan T, Sun D-W (2004) Improving quality inspection of food products by computer vision – A review. J Food Eng 61(1 Spec.):3–16, ISSN: 02608774

Brown M, Dawson L, Kaiser S (1991) Machine vision automates solderability inspection, electrecon. Proceedings, pp 192–200

Brown CR, Harrison S, Furness P (1997) Exploiting IEEE 1355 routable serial links in a real-time vision architecture. Real-Time Imag 3(5):355–361, ISSN: 10072014

Brown N, Jackson MR, Parkin RM, Bamforth PE (2004) Machine vision in conjunction with a knowledge-based system for semi-automatic control of a gravure printing process. Proceedings of the institution of mechanical engineers. Part I. J Syst Control Eng 218(7):583–593, ISSN: 09596518

Brzakovic D, Khani DT, Awad B (1992) A vision system for monitoring weld pool. Proc IEEE Int Conf Robot Autom 2:1609–1614, ISBN: 0818627204

Buchwald DL (1998) Machine vision and robotics laboratory. ASEE annual conference proceedings, 4 pp, ISSN: 01901052

Buffa MG (1985) Process control for automated component assembly of surface mounted devices utilizing a machine vision controller, 5 pp, 180–5. 200, ISBN: 0872631737

Bulanon DM, Burks TF, Alchanatis V (2009) Fruit visibility analysis for robotic citrus harvesting. Trans ASABE 52(1):277–283, ISSN: 00012351

Burggraaf P (1985) Inspection trends in ic assembly. Semiconductor Int 8(6):76–81, ISSN: 01633767

Burns BM (1987) Machine vision applications in powertrain manufacturing. Proceedings of the annual control engineering conference, pp 218–229, ISBN: 0914331566

Burns RA (1991) Machine vision solution to SMD inspection. Surface Mount Technol 5(11):44–46, ISSN: 08933588

Butler R (2006) What's cooking? Profession Eng 19(6):44, ISSN: 09536639

Byrne G, Sheahan C (2005) In-line color variation analysis to quantify the quality of electroplated deposits in high volume manufacture. Proc SPIE Int Soc Opt Eng 5823, art. no. 14, pp 113–120, ISSN: 0277786X

Cadogan WJ (1986) Designing a turnkey machine vision system. Photon Spectra 20(3):53–54, 56, ISSN: 07311230

Cai G-S, Zuo B-R, Tait RW, Harding KG, Peng H, Azer M (2005) Vision-based dual feedback control of deposition height for laser consolidation. 24th international congress on applications of lasers and electro-optics, ICALEO 2005 – Congress proceedings, pp 856–861; Chen L-C, Yang S-L, Cheng K-C, Huang Y-T (2005) TFT-LCD macro defect detection using innovative background reconstruction algorithm. J Chinese Soc Mech Eng Trans Chinese Inst Eng Ser C/Chung-Kuo Chi Hsueh Kung Ch'eng Hsuebo Pao, 26(6):785–789, ISSN: 02579731

Caldwell I, Daley W, Thompson C (1989) Development of a software based machine vision system. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–208, ISSN: 01616382

Caleb-Solly P, Smith JE (2007) Adaptive surface inspection via interactive evolution. Image Vis Comput 25(7):1058–1072, ISSN: 02628856

Callaghan MJ, McGinnity TM, McDaid L (2004) Third order loose coupled hybrid intelligent system for machine vision applications. Conference proceedings – IEEE international conference on systems, man and cybernetics, 4, pp 3680–3685, ISSN: 1062922X

Callaghan MJ, McGinnity TM, McDaid L (2005) Evolving task specific algorithms for machine vision applications. Proceedings – 3rd international conference on information technology and applications, ICITA 2005, I, art. no. 1488829, pp 371–374, ISBN: 0769523161; 9780769523163

Campoy P, Canaval J, Peña D (2005) InsPulp-I©: AAAAn on-line visual inspection system for the pulp industry. Comput Ind 56(8–9):935–942, ISSN: 01663615

Canadian Programmable Control and Automation Products Conference and Exhibition: Automation Integration: the Next Step (1988) 143 p; de Almeida A, Fachada H, Dias J, Amado P, Menezes P, Nunes U (1988) Low-cost, high performance servo-pneumatic manipulators with sensor feedback, pp 465–470, ISBN: 0818620129

Cao Z, Zheng H, Jiang Q, Song B (2007) Dynamic modeling and control programme simulation for sterilization cauldron. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430720, pp 82–85, ISBN: 1424413583; 9781424413584

Caracappa S (ed) (1997) On improving BGA reliability and quality. Surface Mount Technol 11(6):44–46, ISSN: 08933588

Carcone J, Jensen J, Johnson D (1984) Machine vision explosion in sight. Chilton's Instr Control Systems 57(12):29–33, ISSN: 01640089

Carey E (2009) GigE vision comes of age. Adv Imag 24(3):16–18, ISSN: 10420711

Carlson DE, Arya RR, Bennett M, Chen L-F, Jansen K, Li Y-M, Newton J, Rajan K, Romero R, Talenti D, Twesme E, Willing F, Yang L (1996) Commercialization of multijunction amorphous silicon modules. Conference record of the IEEE photovoltaic specialists conference, pp 1023–1028, ISSN: 01608371

Carmel D (1986) Implementing vision guided robotics, pp 7.1–7.20, ISBN: 0872632296

Carneiro N, Souto AP (2001) Application of machine vision to laboratory tests for the textile industry. Textile Science '93, pp 220–223; Braggins D (2001) Visionary developments for the 21st century. Mater World 9(1):17–18, ISSN: 09678638

Carrell RM (1981) Robots for industry. RCA Eng 26(4):6–11, ISSN: 00486574

Carter C (2006) Moving it about. Engineer 293(7698):45–48, ISSN: 00137758

Cassidy R, Morrow PJ, McCloskey JA (2006) machine vision system for quantifying velocity fields in complex rock models. Mach Vision Appl 16(6):343–355, ISSN: 09328092

Chamberlain G (2004) Gigabit ethernet: Finds a foothold in high-performance vision. Photon Spectra 38(12):84–86, ISSN: 07311230

Chan JP (1994) Automated inspection and monitoring of glassware at the hot end. Glass Technol 35(5): 200–203, ISSN: 00171050

Chan S (1995) Using machine vision for aspheric optical profiling measurements. Proc SPIE Int Soc Opt Eng 2600:151–157, ISSN: 0277786X, ISBN: 0819419648

Chan S (2005) Automated inspection systems for the electronics industry. EE Eval Eng 44(8):56–59, ISSN: 01490370

Chan JP, Batchelor BG (1993) Learning method for the inspection of continuously repeated patterns. Proc SPIE Int Soc Opt Eng 1823:186–194, ISSN: 0277786X, ISBN: 0819410241

Chan JP, Palmer GS (1995) Machine vision – Applications in industry. IEE Colloquium (Digest) (113): 1/1–1/6, ISSN: 09633308

Chan VH, Bradley C, Vickers GW (2001) A multi-sensor approach to automating co-ordinate measuring machine-based reverse engineering. Comput Ind 44(2):105–115, ISSN: 01663615

Chang MMT (2007) For the machine vision industry, moving forward is standard practice. Photon Spectra 41(5):44–45, ISSN: 07311230

Chang DS, Jiang S-T (2002) Assessing quality performance based on the on-line sensor measurements using neural networks. Comput Ind Eng 42(2–4): 417–424, ISSN: 03608352

Chang JY, Shen JY (2008) Machine vision assistance for flex cable-actuator comb manufacturing in hard disk drives. 15th international conference on mechatronics and machine vision in practice, M2VIP'08, art. no. 4749532, pp 195–200, ISBN: 9780473135324

Chao Y (1989) Using machine vision to generate patterns for cutting machines. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–221, ISSN: 01616382

Chapman KW (1987) Accurate placement of SMDS: Vision meets the challenge. Robot World 5(11): 27–30, ISSN: 07377908

Chapman D, Deacon A (1998) Virtual environments from panoramic images. Proc SPIE Int Soc Opt Eng 3641:118–126, ISSN: 0277786X

Chapman KW, Johnson WC, McLean TJ (1990) A high speed statistical process control application of machine vision to electronics manufacturing. Comput Ind Eng 19(1–4):234–238, ISSN: 03608352

Chatburn LT, Batchelor BG (2004) 2D packing using the myriad framework. Proc SPIE Int Soc Opt Eng 5265:189–198, ISSN: 0277786X

Chatburn LT, Jackson GR, Daley MW, Batchelor BG (2004) Framework for distributed and networked vision systems. Proc SPIE Int Soc Opt Eng 5265:98–109, ISSN: 0277786X

Chaudhury B (2008) Single axis manipulator control using machine vision and virtual reality technique. IEEE international symposium on industrial electronics, art. no. 4676986, pp 2555–2560, ISBN: 1424416655; 9781424416653

Che C (1992) Scanning compound surfaces with no existing CAD model by using laser probe of a coordinate measuring machine. Proc SPIE Int Soc Opt Eng 1779:56–67, ISSN: 0277786X, ISBN: 0819409510

Chen M-C (2002) Roundness measurements for discontinuous perimeters via machine visions. Comput Ind 47(2):185–197, ISSN: 01663615

Chen J-M (1997) A genetic-based vision system for cross-functional integration in flexible manufacturing:

A tutorial and application. Int J Flex Manuf Syst 9(4):343–365, ISSN: 09206299

Chen FL, Su CT (1996) Vision-based automated inspection system in computer integrated manufacturing. Int J Adv Manuf Technol 11(3):206–213, ISSN: 02683768

Chen A, He B (2007) A camera calibration technique based on planar geometry feature. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430736, pp 165–169, ISBN: 1424413583; 9781424413584

Chen S, Baietto H, Hathaway J (1991) SPC applications of 3-D solder measurements. National electronic packaging and production conference-proceedings of the technical program (West and East), 1, pp 113–124, ISSN: 04700155

Chen S-L, Kok KT, Huang S (2007) Improvement of tracking performance of servomechanical system by an accurate four-parameter friction modelling and compensation. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430709, pp 28–34, ISBN: 1424413583; 9781424413584

Chen SY, Li YF, Zhang J (2007) Realtime structured light vision with the principle of unique color codes. Proceedings – IEEE international conference on robotics and automation, art. no. 4209129, pp 429–434, ISBN: 1424406021; 9781424406029

Chen Y, Li F, Lu M (2007) Vision based gesture by structural features. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430744, pp 206–210, ISBN: 1424413583; 9781424413584

Chen W, Fang K (2008) A fast billet location algorithm using particle Swarm optimization. IEEE/ASME international conference on advanced intelligent mechatronics, AIM, art. no. 4601815, pp 1098–1102, ISBN: 9781424424955

Chen R, Huang R, Zhang Z, Shi J, Chen Z (2008) A distortion-correction method for workshop machine vision measurement system. Proc SPIE Int Soc Opt Eng 7130, art. no. 71304B, ISSN: 0277786X

Chen Z, Zhang Z, Shi J, Chen R, Huang R (2008) Effect of ambient light on surface roughness inspection. Proc SPIE Int Soc Opt Eng 7130, art. no. 71301P, ISSN: 0277786X

Chen J-M, Ventura JA, Melloy BJ (1994a) An optimization algorithm for shape analysis of regular polygons. Mach Vision Appl 7(2):82–92, ISSN: 09328092

Chen FL, Joo D, Black JT (1994b) Machine vision in the automated detection and diagnosis of dimensional errors in end machining. Comput Ind Eng 26(2):223–235, ISSN: 03608352

Chen J-M, Ventura JA (1995) Shape analysis of complex profiles. Int J Mach Tools Manuf 35(3):399–429, ISSN: 08906955

Chen J, Zhou H, O'Yang D, Buckley S (1997) Research on a geometric model based 3D inspection machine. Proc SPIE Int Soc Opt Eng 2909:215–222, ISSN: 0277786X

Chen Michael, Suetens Paul, Oosterlinck Andre (1985) Artificial intelligence in vision systems for future factories. Test Measure World 5(12):5, Between pp 52 and 60, ISSN: 07441657

Chen H-S, Chu L-H, Ito CT, Ofman L, Roy D, Srinivasan S (1988) Design of an automatic lead inspection system: An integrated application of machine vision and robotics, 5th edn, pp 221–232; Holt CB (1988) Physics in the food industry. Phys Technol 19(1), art. no. I03, pp 18–23, ISSN: 03054624

Chen H, Sun D, Yang J (2007) Global localization of multirobot formations using ceiling vision SLAM strategy. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430733, pp 146–151, ISBN: 1424413583; 9781424413584

Chen M, Hou X-R, Qiu X-H (2006) An explicit criterion of the positional relationship of two planar ellipses. Proceedings of the 2006 international conference on machine learning and cybernetics, 2006, art. no. 4028797, pp 4138–4143, ISBN: 1424400619; 9781424400614

Cheng Y, Jafari M (2003) Vision- based process control in layered manufacturing. Proc SPIE Int Soc Opt Eng 5132:303–313, ISSN: 0277786X

Cheng D, Xie S, Hämmerle E (2007) Comparison of local descriptors for image registration of geometrically-complex 3D scenes. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430732, pp 140–145, ISBN: 1424413583; 9781424413584

Cheng Y, Jafari MA (2008) Vision-based online process control in manufacturing applications. IEEE Trans Autom Sci Eng 5(1):140–153, ISSN: 15455955

Chetima MM, Payeur P (2008) Feature selection for a real-time vision-based food inspection system. ROSE 2008 – IEEE international workshop on robotic and sensors environments proceedings, art. no. 4669192, pp 120–125, ISBN: 9781424425952

Chetverikov D, Lerch A (1992) Prototype machine vision system for segmentation of hide images. Int J Imag Syst Technol 4(1):46–50, ISSN: 08999457

Chickering B, Fass L (1987) Robotic welding and machine vision combine to meet manufacturing demands. Weld J (Miami, FL) 66(11):42–43, ISSN: 00432296

Chin RT (1982) Machine vision for discrete part handling in industry: A survey, pp 26–32; Anon, IEEE IECI Proceedings, papers presented at IECI '81: Applications of mini and microcomputers, 1981, IEEE IECI proceedings (IEEE industrial electronics and control instrumentation society), 449 p

China meets machine vision: An emerging industry supports demand for automation and inspection (2006) Laser Focus World 42(6):55, ISSN: 10438092

Chiou Y-C, Liang Y-T (2010) An effective corner detection method using subpixel edge detector and Gaussian filter. Sensor Rev 30(1):51–61, ISSN: 02602288

Chiou R, Kwon Y, Rauniar S, Sosa H (2007) Internet-based robotics and mechatronics experiments for remote laboratory development. ASEE annual conference and exposition, conference proceedings, 16 p; Sako H (2007) Recognition strategies in machine vision applications. International machine vision and image processing conference, IMVIP 2007, art. no. 4318130, p 3, ISBN: 0769528872; 9780769528878

Chiou R, Mookiah P, Kwon Y (2009) Manufacturing e-quality through integrated web-enabled computer vision and robotics. Int J Adv Manuf Technol 43(7–8):720–730, ISSN: 02683768

Chiu M-C, Yeh L-J, Hsu C-J (2008) The deficiency recognition in PCBA's automatic optical inspection system by using back-propagation network method. J Appl Sci 8(16):2814–2824, ISSN: 18125654

Cho T-H, Chang H-K (1994) Automatic recognition of wire bobbins using machine vision techniques. Technical paper – Society of manufacturing engineers. MS, pp 1–6, ISSN: 01616382

Chou Jen-Ya, Plice R, Mead D (1985a) Machine vision system as an automatic sorting and decoding tool. Proc SPIE Int Soc Opt Eng 579:494–499, ISSN: 0277786X, ISBN: 0892526149

Chou J-Y, Plice R, Mead D (1985b) Machine vision system as an automatic sorting and decoding tool. Proc SPIE Int Soc Opt Eng 595:272–277, ISSN: 0277786X, ISBN: 0892526300

Chou PB, Rao AR, Sturzenbecker MC, Wu FY, Brecher VH (1996) Automatic defect classification for semiconductor manufacturing. Mach Vision Appl 9(4):201–214, ISSN: 09328092

Christian DJ, Shroff H (1997) Machine vision-based alignment: Space to factory to garage. Proc SPIE Int Soc Opt Eng 3205:218–225, ISSN: 0277786X

Chuang S-F, Chang W-T, Lin C-C, Tarng Y-S (2010) Misalignment inspection of multilayer PCBs with an automated X-ray machine vision system. Int J Adv Manufactur Technol, 1–14, ISSN: 02683768 (in press)

Chung YK, Kim KH (1998) Automated visual inspection system of automobile doors and windows using the adaptive feature extraction. International conference on knowledge-based intelligent electronic systems. Proceedings, KES, 3, pp 286–293

Cielo P, Vaudreuil G (1989) Optical inspection of continuously manufactured products, pp 95–107, ISBN: 1556172028

Clark LM (1991) Linear motor technology for SMD placement. Electronic materials: Technology, here and now, pp 202–208, ISBN: 0938994581

Close-Range Photogrammetry Meets Machine Vision (1990) Proc SPIE Int Soc Opt Eng 1395(pt 1): 653 p, ISBN: 0819404411

Coderre J (2000) Machine vision illuminates the future of electronics. Photon Spectra 34(12):90–92, ISSN: 07311230

Cognex machine vision chosen by major European subcontractor (2004) Assembly Autom 24(1):102–103

Colet P (2006) Machine vision aids quality inspection. Quality 45(10):22–23, ISSN: 03609936

Colet Philip, Kohli Inder (2000) Wintel and embedded vision processors: A new way of thinking for OEMs. Adv Imag 15(2):2, ISSN: 10420711

Coletta MP, Harding KG (1989) Lighting science. Tools to guide machine vision. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–216, ISSN: 01616382

Colish J (2006) OmniVision launches all digital automotive CameraChip™ sensor. Automotive Ind, 186(10), ISSN: 10994130

Complete inspection of complex parts (2003) Foundry Trade J 177(3606):30, ISSN: 00159042

Computers In Engineering 1986: Proceedings of the 1986 ASME international computers in engineering conference and exhibition (1986) Computers in engineering. Proceedings of the international computers in engineering conference, 1144 p

Conner R (2001) MEMS/MEOMS – Cognitive machine vision in a MOEMS world. Proc SPIE Int Soc Opt Eng 4532:517–528, ISSN: 0277786X

Conners RW, Kline DE, Araman PA, Drayer TH (1997) Machine vision technology for the forest products industry. Computer 30(7):43–48, ISSN: 00189162

Connolly C (2004) Optical inspection tools for the automotive industry. Sensor Rev 24(4):347–352, ISSN: 02602288

Connolly C (2005a) Machine vision developments at IPOT 2005. Sensor Rev 25(3):188–191, ISSN: 02602288

Connolly C (2005b) Part-tracking labelling and machine vision. Assembly Autom 25(3):182–187, ISSN: 01445154

Connolly C (2005c) The use of infrared imaging in industry. Assembly Autom 25(3):191–195, ISSN: 01445154

Connolly C (2007) Sensor trends in processing and packaging of foods and pharmaceuticals. Sensor Rev 27(2):103–108, ISSN: 02602288

Cooper EG, Young SD (2005) Database integrity monitoring for synthetic vision systems using machine vision and SHADE. Proc SPIE Int Soc Opt Eng 5802, art. no. 17, pp 185–194, ISSN: 0277786X

Cooper PW, Curnan HJ, Knarr C (1986) Machine vision inspection criteria for hybrid microelectronics multilayer substrates, pp 3.1–3.14, ISBN: 0872632296

Corke PI, Dunn PA, Mills DC (1994) Real time industrial machine vision. National conference publication – Institution of engineers. Australia 1(94 /11):371–375, ISSN: 03136922

Cremer SC (2001) Advanced sensor technologies replace CMM touch probes. Sensor Rev 21(1):23–27, ISSN: 02602288

Crida RC, De Jager G (1997) An approach to rock size measurement based on a model of the human visual system. Miner Eng 10(10):1085–1093, ISSN: 08926875

Cronhjort BT, Niemi AJ, Suoninen E (1997) Computer-aided measurements in mining, mineral and metal processing. Control Eng Pract 5(2):239–245

Crowley BW (1990) Application of one-dimensional machine vision in the textile industry. IEEE Trans Ind Appl 26(2):324–329, ISSN: 00939994

Crump D (2006) Machine vision sees through the heat. Control Eng 53(11):IP8–IP9

Cunningham R, Gennery D, Kan E (1983) From outer space to factory floor. Comput Mechan Eng 1(4):9–15

Dagli CH, Pillarisetti S (1994) Genetic neuro-vision system for autonomous parts assembly. Technical paper – Society of manufacturing engineers. MS, (MS94–192), pp MS94-225-1-9, ISSN: 01616382

Dahlgren RP, Wysocki JA, Pedrotti KD (2009) Aspheric nonimaging concentrators for multimode fiber coupling. Proc SPIE Int Soc Opt Eng 7221, art. no. 722117, ISSN: 0277786X, ISBN: 9780819474674

Dai XL, Hunt MA (2003) Automated image registration in the semiconductor industry: A case study in the direct to digital holography inspection system. Proc SPIE Int Soc Opt Eng 5011:212–221, ISSN: 0277786X

Daley WDR, Stewart J (2009) Proactive detection of bones in poultry processing. Proc SPIE Int Soc Opt Eng 7315, art. no. 73150B, ISSN: 0277786X, ISBN: 9780819475817

Daley W, Britton D (1996) Image compensation for camera and lighting variability. Proc SPIE Int Soc Opt Eng 2907:2–12, ISSN: 0277786X, ISBN: 0819423092

Daley WD, Doll TJ, McWhorter SW, Wasilewski AA (1999) Machine vision algorithm generation using human visual models. Proc SPIE Int Soc Opt Eng 3543:65–72, ISSN: 0277786X

Daley W, Grullon S, Britton D (1999b) Machine vision based quality control decision making for naturally varying product. Proc SPIE Int Soc Opt Eng 3836:226–235, ISSN: 0277786X

Dar IM (1997) Automated vision based quality control for electro-optical module manufacturing. Proc SPIE Int Soc Opt Eng 3205:236–245, ISSN: 0277786X

Dariush B (2003) Human motion analysis for biomechanics and biomedicine. Mach Vision Appl 14(4):202–205, ISSN: 09328092

Davies R (1988) Image analysis techniques for manufacturing. Eng Digest Toronto 34(3):25–26, ISSN: 00137901

Davies R, Carvalho FD, Correia BB, Dinis J (1994) Application of machine vision for automating the manufacture of cork stoppers. IEEE international symposium on industrial electronics, pp 374–379

Davis GW (1986) Classifying and coping with lighting variation, pp 89–98, ISBN: 0948507128

Davis K (2001) The easy way to connect your camera. Photon Spectra 35(10):158–161, ISSN: 07311230

Day W (2005) Engineering precision into variable biological systems. Ann Appl Biol 146(2):155–162, ISSN: 00034746

De La Fuente E, Trespaderne FM, Gayubo F (2003) Detection of small splits in car-body manufacturing. Proceedings of the IASTED international conference on signal processing, pattern reconition, and applications, pp 354–359, ISBN: 0889863784

De Sam Lazaro A, Gürocak H (1996) Precision assembly using force sensing. Int J Adv Manuf Technol 11(2):77–82, ISSN: 02683768

Dechow D (1987) Vision mapping guides laser. Robot World 5(13):40–41, ISSN: 07377908

DeCurtins J, Kremers J (1987) Sketch: A simple-to-use programming system for visually guided robotic arc welding, pp 625–630, ISBN: 0818607874

Demuynck O, Cedeño CP, Moore AL (2009) Industrial machine vision system for fast and precise 3D object localization. Proceedings of the 9th WSEAS international conference on signal processing, computational geometry and artificial vision, ISCGAV ’09, pp 160–164, ISBN: 9789604741083

Design features (2002) Nonwovens report international (377), pp 53–54, ISSN: 09531092

Devaprasad J, Nelson R, Stornant T (1998) Applying robotics and automation for instrument cluster final assembly. Am Soc Mech Eng Paper, pp 1–8, ISSN: 04021215

Dewar R (1987) Machine vision for process management in automotive assembly, SAE technical paper series, 6 p

Dewar R (1987) Machine vision measurement for process management in automotive assembly, p 72, ISBN: 093665967X

Di Stefano L, Boland F (1996) Solder-paste inspection by structured light methods based on phase measurement. Proc SPIE Int Soc Opt Eng 2899:702–713, ISSN: 0277786X, ISBN: 0819423009

Diaz E (2006) Pick the right camera interface. Sensors (Peterborough, NH) 23(12), ISSN: 07469462

Dickey JM (1988) Successes demonstrate inspection reliability. Robot World 6(2):37–39, ISSN: 07377908

Dickmanns ED (1992) A general dynamic vision architecture for UGV and UAV. Appl Intel 2(3):251–270, ISSN: 0924669X

Didocha RJ, Lyons DW, Thompson JC (1985) Integration of tactile sensors and machine vision for control of robotic manipulators, pp 6.37–6.71, ISBN: 0872631893

Digicom ’87 symposium: Mapping the course between automation technologies (1988) 208 p, ISBN: 155617103X

Dixon T (1986) Machine vision targets circuit errors automatically. Electr Prod (Garden City, NY) 28(4):34–41, ISSN: 00134953

Do Y, Kim G, Kim J (2007) Omnidirectional vision system developed for a home service robot. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430746, pp 217–222, ISBN: 1424413583; 9781424413584

Doignon C, Knittel D, Maurice X (2008) A vision-based technique for edge displacement and vibration estimations of a moving flexible web. IEEE Trans Instrum Meas 57(8):1605–1613, ISSN: 00189456

Dombre E, Fournier A, Quaro C, Borrel P (1986) Trends in CAD/CAM systems for robotics, pp 1913–1918, ISBN: 0818606959

Donlin M (1993) Incredible shrinking vision system. Electr Syst Technol Design/Comput Design 32(7):3, ISSN: 00104566

Don’t miss the big show assembly technology expo (2007) Assembly 50(10):50–51, ISSN: 10508171

Dori D (1989) Enhancing CAD/CAM systems communication by understanding engineering machine drawings, p 437, ISBN: 0897912993

Dori D (1996) Object-process analysis of computer integrated manufacturing documentation and inspection functions. Int J Computer Integr Manuf 9(5):339–353, ISSN: 0951192X

Dougan MJ, Celeiro JL, Rubio A (2009) Engineering systems to meet the challenge of zero-defect parts. Met Powder Rep 64(10):29–33, ISSN: 00260657

Dowell FE (1992) Automatic control of a peanut grade sample inspection system. Food Control 3(2):105–108, ISSN: 09567135

Doyle PC (1986) Machine vision: Coming of age. Robot Eng 8(6):vs10–vs16, ISSN: 08880816

Drayer TH, King IV WE, Tront JG, Conners RW (1995) Modular and reprogrammable real-time processing hardware, MORRPH. IEEE symposium on FPGAs for custom computing machines. Proceedings, pp 11–19, ISSN: 10823409

Drozda TJ (1984) Flexible assembly system features automatic setup. Manuf Eng 93(6):75–76, ISSN: 03610853

Duan G, Chen Y-W, Sukekawa T (2009a) Automatic optical phase identification of microdrill bits using Active Shape Models. IEEE intrumentation and measurement technology conference, I2MTC 2009, art. no. 5168718, pp 1642–1646, ISBN: 9781424433537

Duan G, Chen Y-W, Sukekawa T (2009b) Automatic optical phase identification of microdrill bits in printed circuit board manufacturing. IEEJ Trans Electr Inform Syst 129(7):1397–1407, ISSN: 03854221

Duarte K, LeBlanc S (1995) KAO – Vision aids floppy quality. Sensor Rev 15(1):19–21, ISSN: 02602288

Dunin-Barkowski I, Kim J-S (2006) Accuracy problems in phase shift based 3D machine vision inspection systems. Proc SPIE Int Soc Opt Eng 6382, art. no. 63820F, ISSN: 0277786X, ISBN: 0819464805; 9780819464804

Dunn M, Billingsley J, Bell D (2006) SR-06-119 vision based macadamia yield assessment. Sensor Rev 26(4):312–317, ISSN: 02602288

Dunn MT, Billingsley J (2007) The use of machine vision for assessment of fodder quality. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430739, pp 179–184, ISBN: 1424413583; 9781424413584

Durán ML, Cernadas E, Plaza A, Sánchez JM, Rodriguez F, Petrón MJ (2000) Could machine vision replace chemical procedure to evaluate fat content in Iberian pig meat? An experimental study. Proc Joint Conf Info Sci 5(2):256–259, ISBN: 0964345692

Dvorak KS, Kelly RT (1987) Assembly systems 2000. Assembly Eng 30(6):48–50, ISSN: 00045063

Dworkin SB, Nye TJ (2006) Image processing for machine vision measurement of hot formed parts. J Mater Process Technol 174(1–3):1–6, ISSN: 09240136

Edwards J (1990) Machine vision and its integration with CIM systems in the electronics manufacturing industry. Comput Aided Eng J 7(1):12–18, ISSN: 02639327

Eerola T, Kamarainen J-K, Leisti T, Halonen R, Lensu L, Kälviäinen H, Oittinen P, Nyman G (2008) Finding best measurable quantities for predicting human visual quality experience. Conference proceedings – IEEE international conference on systems, man and cybernetics, art. no. 4811365, pp 733–738, ISSN: 1062922X

Ejiri M (1989) Recent machine vision research in Japanese industry, p 8; Crowley BW (1989) Application of 1-D machine vision in the textile industry. IEEE annual textile industry technical conference, pp 8/1–8/5

Ejiri M (2001) Robotics and machine vision for the future – An industrial view, IEEE/ASME international conference on advanced intelligent mechatronics, AIM, 2, pp 917–922; Smith ML, Farooq AR, Smith LN, Midha PS (2001) An innovative approach to surface inspection using an alliance of machine vision and computer graphical techniques. Proc SPIE Int Soc Opt Eng 4189:99–109, ISSN: 0277786X

Ejiri M (2007) Machine vision in early days: Japan’s pioneering contributions. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), 4843 LNCS (PART 1), pp 35–53, ISSN: 03029743, ISBN: 9783540763857

El-Hakim S, Pizzi NJ (1993) Multicamera vision-based approach to flexible feature measurement for inspection and reverse engineering. Opt Eng 32(9): 2201–2215, ISSN: 00913286

El-Hakim SF, Pizzi NJ, Westmore DB (1992) VCM automated 3-D measurement system: Theory, application, and performance evaluation. Proc SPIE Int Soc Opt Eng 1708:460–482, ISSN: 0277786X, ISBN: 0819408735

Eliason LL, West JK (1986) Vision-based robotics for process management. Robot Eng 8(9):l2–l4, ISSN: 08880816

Elick R, Yoder S (1987) Automating speedometer inspections. Robot World 5(6):22–25, ISSN: 07377908

Elliott WA, Hannebaum R (1998) Fiber placement inspection system – An experimental approach. Int SAMPE Symp Exhibit Proc 43(1):957–963, ISSN: 08910138

ElMasry G, Nassar A, Wang N, Vigneault C (2008) Spectral methods for measuring quality changes of fresh fruits and vegetables. Stewart Postharvest Rev 4(4), art. no. 3, ISSN: 17459656

Emerson C (1987) 'Detroit-style' automation. Am Mach Autom Manuf 131(7):81–92, ISSN: 08860335

Emrich M (1988) Some assembly – and invention – required. Manuf Syst 6(3):26–29, ISSN: 0748948X

Englander AC (1986) Edge detection techniques for industrial machine vision, pp 5.85–5.105, ISBN: 0872632296

Erényi I, Pongrácz J (1991) Quality control in textile industry via machine vision. Microprocess Microprogram 32(1–5):807–813, ISSN: 01656074

Erevelles WF (1995) Implementing a flexible assembly cell (FAC) – phase I. ASEE Ann Conf Proc 1: 711–715, ISSN: 01901052

Erten G, Salam FM (1999) Real time realization of early visual perception. Comput Electr Eng 25(5):379–407, ISSN: 00457906

Ettenberg MH, Cohen MJ, Brubaker RM, Lange MJ, O'Grady MT, Olsen GH (2002) Indium gallium arsenide imaging with smaller cameras, higher resolution arrays, and greater material sensitivity. Proc SPIE Int Soc Opt Eng 4721:26–36, ISSN: 0277786X

Everett CHR, Jenkins RL (1989a) Robotics in the factory of the future. Robotic technology in shipbuilding applications. Rob Autom Syst 4(4):281–308, ISSN: 09218890

Everett HR, Jenkins RL (1989b) Robotic technology in shipbuilding applications. Robot Amsterdam 4(4):281–308

Fadum Ole (2000) Honeywell sells Web inspection business to Cognex. Process Control News (for the Pulp and Paper Industries) 20(10):1, ISSN: 07483236

Fang J, Lei W (2007) A study on dynamic errors of high speed press mechanism with clearance. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430749, pp 234–239, ISBN: 1424413583; 9781424413584

Farooq AR, Smith ML, Smith LN, Midha S (2005) Dynamic photometric stereo for on line quality control of ceramic tiles. Comput Ind 56(8-9):918–934, ISSN: 01663615

Felix R (2003) Image analysis based on qualicision. Sensor Rev 23(3):211–215, ISSN: 02602288

Ferariu L, Panescu D (2009) Multiobjective selection of features for pattern recognition. IEEE international workshop on robotic and sensors environments, ROSE 2009 – proceedings, art. no. 5355996, pp 139–144, ISBN: 9781424447787

Fernandez X, Amat J (1999) Research on small fiducial mark use for robotic manipulation and alignment of ophthalmic lenses. IEEE symposium on emerging technologies and factory automation. ETFA 2:1143–1146

Ferry W, Otieno A (2004) Development of a low cost laboratory system for teaching automation system integration in the manufacturing engineering technology curriculum. ASEE annual conference proceedings, pp 3517–3524, ISSN: 01901052

Fichter E (1986) Simulation of the sri vision algorithm on a microcomputer. Computers in engineering. Proceedings of the international computers in engineering conference, 3, pp 83–88

Finch NA, Murray PJ, Dunn MT, Billingsley J (2006) Using machine vision classification to control access of animals to water. Austr J Experiment Agric 46(6–7):837–839, ISSN: 08161089

Finney B (2007) Using machine vision. Quality 46(1):26–28, ISSN: 03609936

Fischer AL (2004) Available for hire: Robot with 20/20. Photon Spectra 38(11):44–46, ISSN: 07311230

Fischer AL (2006) German and European vision markets post gains. Photon Spectra 40(1):104–106, ISSN: 07311230

Fitzgerald CT (1989) Innovative vision on a standard platform. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–207, ISSN: 01616382

Flack WW, Flores GE, Tran T (1995) Application of pattern recognition in mix-and-match lithography. Proc SPIE Int Soc Opt Eng 2440:913–927, ISSN: 0277786X, ISBN: 0819417882

Flanagan D (1998) Infrared machine vision – A new contender. Sensors (Peterborough, NH) 15(4):29–33, ISSN: 07469462

Flynn PJ, Jain AK (1989) CAD-based computer vision: From CAD models to relational graphs. Proc IEEE Int Conf Syst Man Cybernet 1:162–167

Ford RM, Mercier JA (2003) A machine vision technique for measuring glass container thickness. Proc SPIE Int Soc Opt Eng 5011:82–89, ISSN: 0277786X

Forest J (2008) Found in the clouds. Adv Imag 23(9): 8–11, ISSN: 10420711

Foucher B (1999) Infrared machine vision – A new contender. Proc SPIE Int Soc Opt Eng 3700:210–213, ISSN: 0277786X

Francois S (2004) Industry standards spur innovation. Adv Imag 19(9):47–49, ISSN: 10420711

Freeman G (1989) Screen print insurance. Circuit Manuf 29(4):27–30, ISSN: 00097306

Friedrich WE, Malone DE (1993) Machine vision and expert programming. The brain for robots. Process Control Eng 46(6):4, ISSN: 08168148

From cams to computers (2007) Assembly 50(3 I):70–74, ISSN: 10508171

Fryburg GA (2000) ESD-safe lighting. The not-to-be forgotten part of ESD-safe working environments. Robot World 18(2):3, ISSN: 07377908

Fu JJC, Lee C-Y (1998) Synthetic wavelet approach to machine vision enhancement and feature extraction. J Chinese Soc Mech Eng Trans Chinese Inst Eng Ser C/Chung-Kuo Chi Hsueh Kung Ch'eng Hsuebo Pao 19(6):625–632, ISSN: 02579731

Fu S, Liu H-Y, Gao L-F, Gai Y-X (2007) SLAm for mobile robots using laser range finder and monocular vision. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430722, pp 91–96, ISBN: 1424413583; 9781424413584

Fujii Y, Hashimoto S (2007) Evaluation of the dynamic properties of a Voice Coil Motor (VCM). Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430715, pp 57–61, ISBN: 1424413583; 9781424413584

Furferi R, Governi L (2008) Machine vision tool for real-time detection of defects on textile raw fabrics. J Text Inst 99(1):57–66, ISSN: 00405000

Future vision (2007) Engineering 248(1):35–38, ISSN: 00137782

Gabbay D (2006) Highly integrated mm-wave motion and proximity detector front-end. IEEE convention of electrical and electronics engineers in Israel. Proceedings, art. no. 4115252, pp 87–90, ISBN: 1424402301; 9781424402304

Gagliardi G, Hatch GF, Sarkar N (1985) Machine vision applications in the food industry, pp 6.40–6.54, ISBN: 0872631737

Gain in Spain (2005) Converter 42(8):12, ISSN: 00108189

Gamage P, Xie SQ (2007) A real time vision system for defect inspection in a cast extrusion manufacturing process. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430750, pp 240–245, ISBN: 1424413583; 9781424413584

Gamage P, Xie SQ (2009) A real-time vision system for defect inspection in cast extrusion manufacturing process. Int J Adv Manuf Technol 40(1–2):144–156, ISSN: 02683768

Gan ZX, Zhang H, Wang JJ (2007) Behavior-based intelligent robotic technologies in industrial applications. Lecture Notes Control Info Sci 362:1–12, ISSN: 01708643, ISBN: 3540733736; 9783540733737

Gao X-J, Tang Z-L, Zhang X-C, Chen Y, Jiang L-Q, Cheng H-B (2009) Short wave infrared InGaAs focal plane arrays detector: The performance optimization of photosensitive element. Proc SPIE Int Soc Opt Eng 7383, art. no. 73832S, ISBN: 9780819476647

Garakani A, Cobb J (1986) Machine vision meets the challenge for small measures. Robot World 4(4):5 p between p 60 and 68, ISSN: 07377908

García E, Lamela H (2001) Low-cost three-dimensional vision system based on a low-power semiconductor laser rangefinder and a single scanning mirror. Opt Eng 40(1):61–66, ISSN: 00913286

Gat N, Subramanian S, Barhen J, Toomarian N (1997) Spectral imaging applications: Remote sensing, environmental monitoring, medicine, military operations, factory automation, and manufacturing. Proc SPIE Int Soc Opt Eng 2962:63–77, ISSN: 0277786X

Gearman TA, Culbreth CT (1989) Making wood furniture: Robot lends helping hands. Robot World 7(1):19–20, ISSN: 07377908

Gedeon DV, Gedeon TD (1993) Applying machine vision in electrical component manufacturing. Proceedings of the 21st electrical electronics insulation conference and electrical manufacturing and coil winding, pp 737–749, ISBN: 0780308476

Geraghty S (2001) Emerging trends in machine vision. Sensors (Peterborough, NH), 18(8):44+46+48, ISSN: 07469462

Geraghty S (2003) Seeing it my way. InTech 50(4):34–36, ISSN: 0192303X

Geraghty S, Ciardiello C (2008) Factory automation systems in sight. InTech 55(6), ISSN: 0192303X

Gerber J, Notni G, Schreiber W, Kowarschik R (1993) 3D-shape measuring system using projected fringes. Proc SPIE Int Soc Opt Eng 1983(pt 2):938–939, ISSN: 0277786X, ISBN: 0819412309

Gerst C (2003) Machine vision bolsters quality control. Photon Spectra 37(12):49–50, ISSN: 07311230

Ghanbari M, Hojjat Y, Dadkhah M, Bagherinia M (2007) Designing and modelling of a modern artificial fluid wave generation system with flexible mechatronic control. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430729, pp 123–129, ISBN: 1424413583; 9781424413584

Ghazali KH, Razali S, Mustafa MM, Hussain A (2008) Machine vision system for automatic weeding strategy in oil palm plantation using image filtering technique. 3rd international conference on information and communication technologies: From theory to applications, ICTTA, art. no. 4530075, ISBN: 1424417511; 9781424417513

Ghio A, Pischiutta S (2007) A support vector machine based pedestrian recognition system on resource-limited hardware architectures. Proceedings of the 2007 PhD research in microelectronics and electronics conference, PRIME 2007, art. no. 4401836, pp 161–163, ISBN: 1424410002; 9781424410002

Ghosal S, Mehrotra R (1992) Range image segmentation using Zernike moment-based generalized edge detector. Proc IEEE Int Conf Robot Autom 2: 1584–1589, ISBN: 0818627204

Gieles ACM, Venema WJ (1989) Inspection of SMD's with 3-D laser scanning. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–204, ISSN: 01616382

Gill KD, Jawaid A (1990) Machine vision technology – Applications in the automotive industry, SAE technical paper series, pp 1–7, ISSN: 01487191

Gill KD, Jawaid A (1990) Machine vision technology. Applications in the automotive industry. SAE (Society of Automotive Engineers) Transactions 99 (Sect. 5):1126–1132

Godber SX, Robinson M (1993) Machine vision using line-scan sensors. Proc SPIE Int Soc Opt Eng 1823:114–130, ISSN: 0277786X, ISBN: 0819410241

Goharla'ee M, Raaberg M (1987) Applications of structured lighting for volume measurement. Proc SPIE Int Soc Opt Eng 728:65–73, ISSN: 0277786X, ISBN: 0892527633

Golnabi H (2002) Laser based sensor systems for automation and intelligent manufacturing. Technical paper – Society of manufacturing engineers. MS (MS02–275), pp 1–10, ISSN: 01616382

Golnabi H, Asadpour A (2007) Design and application of industrial machine vision systems. Rob Comput Integr Manuf 23(6):630–637, ISSN: 07365845

Gonçalves PJS, Furtado HAM, Morato JPFR, Gonçalves MAC (2002) Automatic fabric inspection by machine-vision, applying simple algorithms. Proc SPIE Int Soc Opt Eng 4664:198–206, ISSN: 0277786X

Goodman H (1990) Assembly cell speeds riveting. Tool Prod 56(9):50–53, ISSN: 00409243

Gordon GB, Tella RP, Martins HAS (1995) New high-sensitivity capillary electrophoresis detector cell and advanced manufacturing paradigm. Hewlett-Packard J 46(3):62–70, ISSN: 00181153

Govindan R, Rao AS (1994) Optimal strategies for recognizing polygonal parts. Proceedings – IEEE international conference on robotics and automation (pt 4):3564–3569, ISSN: 10504729, ISBN: 0818653329

Granitto PM, Navone HD, Verdes PF, Ceccatto HA (2002) Weed seeds identification by machine vision. Comp Electr Agric 33(2):91–103, ISSN: 01681699

Green J (1985) Image manipulating camera for machine vision, pp 7.54–7.58, ISBN: 0872631893

Gregory R (1998) Planning a PC-based machine vision system. Sensors (Peterborough, NH) 15(4):12–14, 16–18, ISSN: 07469462

Groombridge P, Oloyede A, Doherty-Bigara P (2003) Development and implementation of visual feedback technology in automotive windscreen manufacture. J Mater Process Technol 139(1–3 SPEC):357–361, ISSN: 09240136

Gruver WA, Ansari AA, Schmitt L (1984) Modular robot vision system for manufacturing, 3, pp 9.1–9.20; Banks D (1984) Machine vision – The link between fixed and flexible automation, pp 2.1–2.6, ISBN: 0872631400

Guerra E, Manriquez A, Schwartz D, Villalobos JR (1997) Three dimensional automated visual inspection of surface mounted devices. Comput Ind Eng 33(1–2):365–368, ISSN: 03608352

Guivens NR Jr, Henshaw PD (1992) Image processor development with synthetic images. Proc SPIE Int Soc Opt Eng 1614:167–180, ISSN: 0277786X, ISBN: 0819407518

Guo J, Heyler R (2004) Fast active alignment in photonics device packaging. Proc Electr Component Technol Conf 1:813–817, ISSN: 05695503

Guo J, Ying Y (2008) Progress on detecting technique and sorter of raw cotton foreign matters. Nongye Jixie Xuebao/Trans Chinese Soc Agric Machin 39(7):107–113+106, ISSN: 10001298

Habibi B (2002) Creating a convergence in part-handling. Manuf Eng 128(6), ISSN: 03610853

Häcker J, Engelhardt F, Frey DD (2002) Robust manufacturing inspection and classification with machine vision. Int J Prod Res 40(6):1319–1334, ISSN: 00207543

Hage P, Jones B (1995) Machine vision-based quality control systems for the automotive industry. Assembly Autom 15(4):32–34, ISSN: 01445154

Haggren H (1986) Mapvision: The photogrammetric machine vision system for engineering applications. Photogramm J Finland 10(1):47–54, ISSN: 05541069

Haggren H (1987) Mapvision – The photogrammetric machine vision system for engineering applications. Proc SPIE Int Soc Opt Eng 730:210–215, ISSN: 0277786X, ISBN: 089252765X

Haggren H, Leikas E (1987) Mapvision: The photogrammetric machine vision system. Photogramm Eng Remote Sens 53(8):1103–1108, ISSN: 00991112

Hajimowlana SH, Jullien GA, Muscedere R, Roberts JW (1997) Efficient pre-processing algorithms for an FPGA based in-camera video-stream processing system for industry inspection. Can Conf Electr Comput Eng 2:835–838, ISSN: 08407789

Hakanson WP (1987) Optical recognition speeds accurate data entry. Chilton's Instrum Control Syst 60(5):57–59, ISSN: 01640089

Hakulinen A, Hakkarainen J (1996) A neural network approach to quality control of padlock manufacturing. Pattern Recognit Lett 17(4 SPEC. ISS):357–362, ISSN: 01678655

Halimeh JC, Roser M (2009) Raindrop detection on car windshields using geometric-photometric environment construction and intensity-based correlation. IEEE intelligent vehicles symposium. Proceedings, art. no. 5164347, pp 610–615, ISBN: 9781424435043

Halschka M, Schwarz J, Wildmann D, Wagner F (2003) Machine vision – The powerful tool for quality assurance of laser welding and brazing, ICALEO 2003 – 22nd international congress on applications of laser and electro-optics, congress proceedings, art. no. 1302, ISBN: 0912035757; 9780912035758

Hammer GS II (1987) Future of seeing-eye robots in rehabilitation engineering, pp 672–674; Stone JR, Moritz WE (1987) Machine vision system for the detection of missing components on through-hole, printed circuit boards prior to soldering, pp 46–52; Bieman LH (1987) Novel method for spatial measurement of holes. Proc SPIE Int Soc Opt Eng 728:116–122, ISSN: 0277786X, ISBN: 0892527633

Hamza R, Cofer D (2001) Virtual moiré interference approach for an industrial safety monitoring system. Proc SPIE Int Soc Opt Eng 4399:107–115, ISSN: 0277786X

Han L, Tso SK (2003) Mechatronic design of a flexible vibratory feeding system. Proc Inst Mech Eng B J Eng Manuf 217(6):837–842, ISSN: 09544054

Hanks John, Nair Dinesh (1998) Sum and substance of color in machine vision. Sensors (Peterborough, NH) 15(4):20–22, 24–28, ISSN: 07469462

Hara Y (1998) Progress in industrial application of image processing in Japan. Int J Jpn Soc Precis Eng 32(2):75–80, ISSN: 0916782X

Harding KG (1987) Development of moire machine vision, NASA conference publication (2491), pp 413–419, ISSN: 01917811

Harding K (1996) Promise of machine vision. Opt Photon News 7(5):29–33, ISSN: 10476938

Harding K (2004) The promise and payoff of 2D and 3D machine vision: Where are we today. Proc SPIE Int Soc Opt Eng 5265:1–15, ISSN: 0277786X

Harding KG, Bieman L (1989) Machine vision assisted control in a turning center. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–218, ISSN: 01616382

Harkavy B (1989a) Application of a flexible machine vision architecture to the inspection of continuous process materials. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–165, ISSN: 01616382

Harkavy B (1989b) Denser semiconductor circuitry demands accurate alignment. Robot World 7(1):37–38, ISSN: 07377908

Harms TM (1992) Machine vision: What can it do for you, IEEE conference record of annual conference of electrical engineering problems in the rubber and plastics industries, pp 30–34, ISSN: 02724685, ISBN: 0780305434

Harris R (1994) Machine vision looks to lighting. Light Design Appl LD and A 24(12):15–18, ISSN: 03606325

Harvey AL, Cohen HA (1991) Software speedup techniques for binary image object recognition. IECON Proc (Ind Electron Conf) 3:1827–1831

Hashimoto S, Fujii Y, Kigure M (2007) High-precision control of linear actuators with nonlinear friction. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430716, pp 62–67, ISBN: 1424413583; 9781424413584

He BW, Li YF (2007) A novel method for camera calibration using vanishing points. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430712, pp 44–47, ISBN: 1424413583; 9781424413584

Hegland D (2005) Beantown hosts triple show. Assembly 48(4):84–86, ISSN: 10508171

Heil P (2003) Using your smarts. Canadian Packaging 56(7):S5, ISSN: 00084654

Heleno P, Davies R, Correia B, Dinis J (2001) Vision based algorithms for high accuracy measurements in an industrial bakery. Proc SPIE Int Soc Opt Eng 4567:121–128, ISSN: 0277786X

Heleno P, Davies R, Brazio Correia BA, Dinis JA (2002) machine vision quality control system for industrial acrylic fibre production. Eurasip J Appl Signal Proces 2002(7):728–735, ISSN: 11108657

Hendrawan Y, Murase H (2009) Machine vision-based precision irrigation system for Sunagoke moss production. American society of agricultural and biological engineers annual international meeting 2009, 3, pp 1864–1875, ISBN: 9781615673629

Heng KJ, Even JC Jr (1989) Vision system for quality control. International J Appl Eng Edu 5(3):403–410, ISSN: 07420269

Henkel SvL (1986) Fundamentals of machine vision systems. Robot Eng 8(6):6, ISSN: 08880816

Henke-Reed MB, Cheng SNC (1993) Cloth texture classification using the wavelet transform. J Imag Sci Technol 37(6):610–614, ISSN: 87509237

Henry J, Preston C (1988) Implementing machine vision: An IBM case study. Sensor Rev 8(2):73–78, ISSN: 02602288

Heralić A, Christiansson A-K, Ottosson M, Lennartson B (2010) Increased stability in laser metal wire deposition through feedback from optical measurements. Opt Lasers Eng 48(4):478–485, ISSN: 01438166

Herbon R, Cillessen D, Gamillo E, Hyde A (2009) Engineering a machine to remove stems from chile peppers – A critical need for the new Mexico chile industry. Am Soc Agric Biol Eng Ann Int Meet 2009(2):775–785, ISBN: 9781615673629

Hertel DW, Chang E (2007) Image quality standards in automotive vision applications. IEEE intelligent vehicles symposium. Proceedings, art. no. 4290148, pp 404–409, ISBN: 1424410681; 9781424410682

Heywood J (1987) Shift and correlate pattern recognition for machine vision inspection, CIME. Comput Mech Eng 5(6):21–25, ISSN: 07459726

Heywood J (1989) Robots meet machine vision. Photon Spectra 23(5):3, ISSN: 07311230

Hilton PJ, Gabric R, Walternberg PT (1991) Laser image detection system (LIDS), laser based imaging. Proc SPIE Int Soc Opt Eng 1385:27–31, ISSN: 0277786X

Hoang K, Nachimuthu A (1996) Image processing techniques for leather hide ranking in the footwear industry. Mach Vision Appl 9(3):119–129, ISSN: 09328092

Hoang K, Wen W, Nachimuthu A, Jiang XL (1997) Achieving automation in leather surface inspection. Comput Ind 34(1):43–54, ISSN: 01663615

Hocenski Ž, Keser T (2007) Failure detection and isolation in ceramic tile edges based on contour descriptor analysis. Mediterranean conference on control and automation, MED, art. no. 4433713, ISBN: 142441282X; 9781424412822

Hochfelder B (2009) Back to stuttgart: Nineteen countries, 292 exhibitors and 6, 200 visitors are expected for VISION 2009. Adv Imag 24(7):22–23, ISSN: 10420711

Hodges SE, Richards RJ (1995) Uncalibrated stereo vision for PCB drilling. IEE Colloquium (Digest) (113):4/1–4/6, ISSN: 09633308

Hodgson RM (1992) Digital image processing – A developing technology for enhancing productivity. National conference publication – Institution of engineers. Australia (92 pt 15):273–278, ISSN: 03136922

Hoehn J (1984) Vision systems applications at owens-illinois, pp 3.10–3.14, ISBN: 0872631400

Hoffman BD (1991) Use of 2-d linear motors in surface mount technology. Electronic materials: Technology, here and now, pp 141–151, ISBN: 0938994581

Hoffman R, Keshavan HR (1989) Evidence-based object recognition and pose estimation. Proc IEEE Int Conf Syst Man Cybernet 1:173–178, ISSN: 08843627

Hofhauser A, Steger C, Navab N (2009) Perspective planar shape matching. Proc SPIE Int Soc Opt Eng 7251, art. no. 72510G, ISSN: 0277786X

Hogan H (2005) Machine vision checks chicken and confirms cookies. Photon Spectra 39(10):84–89, ISSN: 07311230

Hogan H (2008) That's entertainment! Photon Spectra 42(9):44–47, ISSN: 07311230

Hogarth S (1999) Machines with vision. Manuf Eng 122(4):100, 102–104, 106–107, ISSN: 03610853

Hojjat Y, Dadkhah M, Modabberifar M, Ghanbari M (2007) Electrostatic rotation of plexiglas disc supported on air bearing. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430730, pp 130–135, ISBN: 1424413583; 9781424413584

Holmsten D (1986) Thermographic sensing for on-line industrial control. Proc SPIE Int Soc Opt Eng 665:75–88, ISSN: 0277786X, ISBN: 0892527005

Holton C (2004a) A sense of place. Laser Focus World 40(1):62, ISSN: 10438092

Holton C (2004b) Machine vision reads tea leaves. Laser Focus World 40((3):57, ISSN: 10438092

Holton C (2004c) Providing an education. Laser Focus World 40(11):66, ISSN: 10438092

Holton C (2005a) Contrast and compare. Laser Focus World 41(11):79, ISSN: 10438092

Holton C (2005b) Engineering sans frontiers. Laser Focus World 41(10):61, ISSN: 10438092

Holton C (2005c) Just what the doctor ordered. Laser Focus World 41(6):77, ISSN: 10438092

Holton C (2005d) The world of tomorrow. Laser Focus World 41(3):53, ISSN: 10438092

Holton C (2006) Made in India. For India, Laser Focus World 42(8):73, ISSN: 10438092

Hölzl PA, Schleicher DCH, Zagar BG (2009) Steel coil, straps and packaging recognition under natural illumination. Proceedings of the 9th IASTED international conference on visualization, imaging, and image processing, VIIP 2009, pp 59–66, ISBN: 9780889868007

Hong Q, Zhang C, Chen X, Chen Y (2007) Embedded speech recognition system for intelligent robot. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430710, pp 35–38, ISBN: 1424413583; 9781424413584

Hong-Dar L, Hsing-Lun C (2009) Automated surface defect inspection of ball grid array substrates by machine vision system. J Qual 16(1):43–60, ISSN: 10220690

Hori T (2005) Developing standards for the automated imaging industry. Photon Spectra 39(1):72–74, ISSN: 07311230

Hormozi AM, Jacobs SM, Wharton TJ (1992) Manufacturing process improvement. The role of vision systems. Prod Invent Manage J 33(4):59–63, ISSN: 08978336

Horn D (1989) Smarter sensors respond to factory stimuli. Mech Eng 111(9):64–67, ISSN: 00256501

Horst RL, Negin M (1989) Machine vision system for precision dimensional measurements and on-line SPC. Conference record – IAS annual meeting

(IEEE industry applications society) (pt 2): 1712–1716, ISSN: 00939994

Horst RL, Negin M (1992) Vision system for high-resolution dimensional measurements and on-line SPC: Web process application. IEEE Trans Ind Appl 28(4):993–997, ISSN: 00939994

Hoske MT (2003) 'See' the results. Control Eng 50(8):24–28, ISSN: 00108049

Hospod TF (1991a) Robust pattern recognition: Key to effective machine vision. Robot World 9(2):24–27, ISSN: 07377908

Hospod Thomas (1991b) Application-specific machine vision software. Sensors (Peterborough, NH) 8(11):34–36, ISSN: 07469462

Houghton H (2002) LEDs: A flexible option for machine vision. Sensor Rev 22(2):130–133, ISSN: 02602288

Hsieh C-L, Weng S-F (2005) Prediction of physical properties of orchid seedlings 'Phalaenopsis Sogo Vivien F819' in a flask by digital imaging. Can Biosyst Eng/ Le Genie des biosystems au Canada 47:3.23–3.32, ISSN: 14929058

Hsieh L-L, Wang C-C (2008) Algorithm design of liquid lens inspection system. Proc SPIE Int Soc Opt Eng 7073, art. no. 70731D, ISSN: 0277786X, ISBN: 9780819472939

Hu P-W, Johnson C, Griffin P (1986) Computer vision system for inspection of light emitting diodes, pp 3.83–3.100, ISBN: 0872632296

Hu W-C, Sheu H-T (2000) Quadratic B-spline for curve fitting. Proceedings of the national science council, Republic of China. Part A Phys Sci Eng 24(5): 373–381, ISSN: 02556588

Hu Y-L, Ding Q (2006) Design of an architecture for multiprocessor system-on-chip (MPSoC). Conference on high density microsystem design and packaging and component failure analysis, HDP'06, art. no. 1707566, pp 63–66, ISBN: 1424404886; 9781424404889

Hu Z, Zheng K (2006) Research on manufacturing technology based on machine vision. Wuhan Ligong Daxue Xuebao/J Wuhan Univ Technol 28(Suppl 1):402–406, ISSN: 16714431

Huang CK, Tarng YS, Chiu CY, Huang AP (2009) Investigation of machine vision assisted automatic resharpening process of micro-drills. J Mater Process Technol 209(18–19):5944–5954, ISSN: 09240136

Huang C-N, Lambert BK (1992) An integrated implementation for product design. Comput Ind Eng 23(1–4):41–44, ISSN: 03608352

Huang C-N, Lim CC, Ming C (1992) Comparison of image processing algorithms and neural networks in machine vision inspection. Comput Ind Eng 23(1-4):105–108, ISSN: 03608352

Huang C-N, Motavalli S (1994) Reverse engineering of planar parts using machine vision. Comput Ind Eng 26(2):369–379, ISSN: 03608352

Huang KT, Chao A, Yu CH (2007) P-123: Image sticking analysis of different Q-time LC Cell by machine vision. Digest Tech PapersSID Int Symp 38(1):665–668, ISSN: 0097966X

Huang S-J, Tsai J-P (2005) Robotic automatic assembly system for random operating condition. Int J Adv Manuf Technol 27(3–4):334–344, ISSN: 02683768

Huang Y-C, Chuang K-C, Lin M-S, Chen C-F (2005) Inspecting LED micro structure by piezo servo system. Proceedings of the 2005 IEEE international conference on mechatronics, ICM '05, 2005, art. no. 1529244, pp 151–156, ISBN: 0780389980; 9780780389984

Huang Y-C, Chuang K-C, Lin M-S (2006) Design of innovative LED micro structure inspection apparatus by piezo servo system. Hangkong Taikong ji Minhang Xuekan/J Aeronaut Astronaut Aviation 38A(3):191–198, ISSN: 10220666

Hubele NF, Beaumariage T, Baweja G, Hong S-C, Chu R (1994) Using experimental design to assess the capability of a system. J Qual Technol 26(1):1–11, ISSN: 00224065

Hudson DL (1983) Practical solution using a new approach to robot vision, pp 9–15; Banks D (1983) Machine vision – The link between fixed and flexible automation. Fall industrial engineering conference, American Institute of Industrial Engineers, pp 198–201

Hudson DL (1984) Food for thought and appearance, pp 3.1–3.9, ISBN: 0872631400

Hudson DL, Solomon SS (1983) Practical solution using a new approach to machine vision. Technical paper – Society of manufacturing engineers, 15 p

Hudson DL, Trombly JE (1983) Developing industrial applications for machine vision. Comput Mech Eng 1(4):18–23

Hughes T (1986) Trends in testing: Much more to come? Prod Eng 33(4):68–69, 72, 75, ISSN: 01461737

Hull DM (1995) Natl. photonics skills standards for technicians. Proc SPIE Int Soc Opt Eng 2525:538–544, ISSN: 0277786X, ISBN: 0819418846

Huller D (1989) Right image, right away. Energy Autom 11(3):28–31, ISSN: 09316221

Hung YY, Park BG (1996) Automated inspection and quantification of dents in automotive bodies using 3-D computer vision technique. Technical paper – Society of manufacturing engineers. AD (282), pp 1–4, ISSN: 03618765

Hunter J (2003) Vision of the future. Packaging Magazine 6(21):5, ISSN: 14614200

Huth MC, Conder RA, Ferry GP, Helterline BL, Aman RF, Hubley TS (1988) CIM and machine vision in the production of thermal inkjet printheads. Hewlett-Packard J 39(5):91–98, ISSN: 00181153

IEE Computing and Control Division Colloquium on Intelligent Automation for Processing Non-rigid

Products (1994) IEE Colloquium (Digest) (191), pp var paging, ISSN: 09633308

IEEE 1989 Annual Textile Industry Technical Conference (1989) IEEE annual textile industry technical conference, 72 p

IEEE Conference Record of 1989 Forty-First Annual Conference of Electrical Engineering Problems in the Rubber and Plastics Industries (1989) IEEE conference record of annual Conference of Electrical Engineering Problems in the Rubber and Plastics Industry, 108 p, ISSN: 02724685

Igathinathane C, Melin S, Sokhansanj S, Bi X, Lim CJ, Pordesimo LO, Columbus EP (2009) Machine vision based particle size and size distribution determination of airborne dust particles of wood and bark pellets. Powder Technol 196(2):202–212, ISSN: 00325910

Ihn YS, Ha HS, Choi BJ, Choi HR, Lee SM, Koo JC (2007) Design of a modified binary region median filtering for micro electronic device assembly manipulations. ICCAS 2007 – International conference on control, automation and systems, art. no. 4406835, pp 2749–2753, ISBN: 8995003871; 9788995003879

Ihn YS, Ryu SH, Choi BJ, Ha SH, Choi HR, Lee SM, Koo JC (2007) An enhanced vision processing algorithm for a micro-manipulation system, ROSE 2007 – International workshop on robotic and sensor environments. Proceedings, art. no. 4373959, pp 7–12, ISBN: 1424415276; 9781424415274

Ikemoto N, Fujinami K, Isomura M (2002) Modeling the luminance distribution of 3-d object surface with specular reflection component and shape measurement. J Light Visual Environ 26(2):19–28, ISSN: 03878805

Inglesby T (1985) General does it robotically. Manuf Syst 3(9):22–24, ISSN: 0748948X

Inglesby T (1989) CMM in CIM. Manuf Syst 7(10):5, ISSN: 0748948X

Inigo RM, Angulo JM (1985) Robotics education in the university. Robotics 1(1):37–47, ISSN: 01678493

Inigo RM, Angulo JM, Aviles R (1984) Robotics education in Spain, pp 18.26–18.38, ISBN: 0872631478

Integrated service solutions (2007) World Cement 38(9 Suppl):67, ISSN: 02636050

Intelligent Robots And Computer Vision (fourth in a series) (1985) Proc SPIE Int Soc Opt Eng 579:562, ISBN: 0892526149

Intelligent video makes real-time surveillance a reality (2006) Laser Focus World 42(8):113–117, ISSN: 10438092

International Electronics Manufacturing Technology Symposium – IEMTS '88, (1990) IEEE transactions on components, hybrids, and manufacturing technology, 13(1), pp 83–130, ISSN: 01486411

Irvine D (1995) Inspection and quality control – An overview, Northcon – Conference Record, pp 376–380

Irvine D, Narayanan M (1995) Inspection and quality control – An overview, Wescon conference record, pp 500–504

Irvine D, Narayanan M (1995) Robotics and vision systems – An overview, Northcon – Conference record, pp 2–5

ISA/93 International Conference, Exhibition and Training Program (1993) Advances in Instrumentation and Control: International Conference and Exhibition, 48 (pt 3), pp 1483–2188, ISSN: 10540032, ISBN: 1556174632

Iwamoto S, Trivedi MM, Checkley DM (1998) Real-time detection and classification of objects in flowing water. Proc SPIE Int Soc Opt Eng 3521:214–220, ISSN: 0277786X

Jackson MR, Preston ME (1998) Lace cutting for the next millennium. Integr Manuf Syst 9(1):34–40, ISSN: 09576061

Jackson M, Preston M, Tao L (1995) High speed cutting of patterned shapes from fabrics. Mechatronics 5(2–3):197–213, ISSN: 09574158

Jain AN, Krig DB (1986) Robust hough technique for machine vision, pp 4.75–4.87, ISBN: 0872632296

Jain KR, Modi CK, Pithadiya KJ (2009) Non-destructive quality evaluation in spice industry with specific reference to cuminum cyminum L (cumin) seeds. Innovative technologies in intelligent systems and industrial applications, CITISIA 2009, art. no. 5224191, pp 311–316, ISBN: 9781424428878

Jain KR, Modi CK, Patel JJ (2009) Non-destructive quality evaluation in spice industry with specific reference to *Cuminum cyminum* L. (Cumin) seeds, 2nd international conference on emerging trends in engineering and technology, ICETET 2009, art. no. 5395509, pp 458–463, ISBN: 9780769538846

Jain KR, Modi CK, Pithadiya KJ (2009) Non-destructive quality evaluation in spice industry with specific reference to Foeniculum vulgare (Fennel) seeds. Proceedings of the 9th IASTED international conference on visualization, imaging, and image processing, VIIP 2009, pp 80–85, ISBN: 9780889868007

Jansen G (2007) Five traps (and a bonus) to avoid in the machine vision business. Adv Imag 22(5):16–19, ISSN: 10420711

Jawaid A, Gill KD (1990) Vision system engineering, Automotive applications at rover group. Proceedings. Int Symp Autom Technol Autom 3:164–175

Jayabal S, Natarajan U (2010) Optimization of thrust force, torque, and tool wear in drilling of coir fiber-reinforced composites using Nelder-Mead and genetic algorithm methods. Int J Adva Manuf Technol, pp 1–11, ISSN: 02683768 (in press)

Jeffery H (2005) Magnetic attraction. Engineering 246(9):27–28, ISSN: 00137782

Jenkins K (1988) Artificial intelligence market in the early l990's, U.S. Woman Eng 34(4):7–9, ISSN: 02727838

Jennings LM (1985) Real-time image processing in manufacturing applications, pp 688–693; Narayanan M (1985) Robot vision systems, pp 82–84; Stewart J (1985) Machine vision, robotics, and printed circuit assembly, Northcon – Conference record, 9 p

Jennings RB, Bright G (1999) Machine vision and intelligence incorporating motion control. Assembly Autom 19(1):55–58, ISSN: 01445154

Ji C, Zhang X, Ying J, Wang Y, Zeng G (2006) PRIC key technologies used in storage yard based on machine vision. WSEAS Trans Info Sci Appl 3(7):1341–1346, ISSN: 17900832

Jia J (2009) A machine vision application for industrial assembly inspection. 2nd international conference on machine vision, ICMV 2009, art. no. 5381107, pp 172–176, ISBN: 9780769539447

Jia J, Krutz GW, Gibson HG (1991) Corn plant locating by image processing. Proc SPIE Int Soc Opt Eng 1379:246–253, ISSN: 0277786X

Jian L, Jun Z, Zhan P-P, Rui M, Sun H-L (2009) Design and development of quality inspection system for forming the longitudinal-seam submerged arc pipes with JCOE process. WRI world congress on computer science and information engineering, CSIE 2009, 6, art. no. 5170724, pp 376–381, ISBN: 9780769535074

Jiang BC (1989) Experience in the development of a vision system for manufacturing education. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–323, ISSN: 01616382

Jiang BC, Chen T-C (2001) Machine vision inspection for the protrusion rate of a diamond tool. J Manuf Syst 20(5):357–362, ISSN: 02786125

Jiang BC, Tasi S-L, Wang C-C (2002) Machine vision-based gray relational theory applied to IC marking inspection. IEEE Trans Semicond Manuf 15(4): 531–539, ISSN: 08946507

Jiao Yu, Mayer RM, Cunningham AP (2000) Vision instrumentation for process control of float baths. Glass Res 10(1):6–7, 22, ISSN: 10868747

Jin F, Qin L, Jiang L, Zhu B, Tao Y (2008) Novel separation method of black walnut meat from shell using invariant features and a supervised self-organizing map. J Food Eng 88(1):75–85, ISSN: 02608774

Johnson C, Foster III, Joseph W (1991a) Achieving quality through flexible inspection. SAE technical paper series, pp 1–7, ISSN: 01487191

Johnson C, Foster III, Joseph W (1991b) Achieving quality through flexible inspection. SAE (Society of Automotive Engineers) Trans 100(Sect 5):935–941, ISSN: 0096736X, ISBN: 1560912766

Jones RE (1991) Machine vision applications. Mechatronics 1(4):439–446, ISSN: 09574158

Jones S, Thomas R, Awcock G, Humphrey K (1995a) Machine vision techniques for ink particle analysis within the paper recycling process. IEE conference publication (410), pp 682–686, ISSN: 05379989

Jones SA, Awcock GJ, Thomas R, Humphrey K (1995b) On-line visual inspection techniques for quality control during the deinking of secondary fiber paper. Proc SPIE Int Soc Opt Eng 2423:215–224, ISSN: 0277786X, ISBN: 081941770X

Joseph J (2008) Tall order. Engineering 249(8):34–36, ISSN: 00137782

Joyner K, Shipman J, Mott B, Harper D, Morris E, Eslami A (2004) A capstone design project- Machine vision system in inspection process. ASEE annual conference proceedings, pp 1579–1586, ISSN: 01901052

Kaartinen J, Hyötyniemi H (2003) Determination of ore size distribution with image analysis. Proceedings of the IASTED international conference on intelligent systems and control, pp 406–411, ISBN: 0889863555

Kachan A (2002) Machine vision guides the automotive industry. Sensor Rev 22(2):119–124, ISSN: 02602288

Kahwati GN (1985) Vision technologies expand manufacturing horizons. Assembly Eng 28(10):18–20, ISSN: 00045063

Kaiser S, Paine W (1989) Automatic tooling hole locator/ punch system. Technical paper – society of manufacturing engineers. MS, pp var paging MS89–210, ISSN: 01616382

Kaiser S, Brown M, Dawson L (1991) Machine vision automates solderability inspection. National electronic packaging and production conference-proceedings of the technical program (West and East) 2, pp 1006–1017, ISSN: 04700155

Kalb B (1987) Robot vision for the PC comes of age. Autom Ind 167(1):66–67, ISSN: 00051527

Kalviainen H, Kukkonen S, Hyvarinen T, Parkkinen J (1998) Quality control in tile production. Proc SPIE Int Soc Opt Eng 3522:355–360, ISSN: 0277786X

Kälviäinen H, Saarinen P, Salmela P, Sadovnikov A, Drobchenko A (2003) Visual inspection on paper by machine vision. Proc SPIE Int Soc Opt Eng 5267:321–332, ISSN: 0277786X

Kamat V, Ganesan S (1998) Robust Hough transform technique for description of multiple line segments in an image. IEEE Int Conf Image Proces 1:216–220

Kameyama K, Kosugit Y, Okahashi T, Izumita M (1998) Automatic defect classification in visual inspection of semiconductors using neural networks. IEICE Trans Info Syst E81-D(11):1261–1271, ISSN: 09168532

Kang D-K, Chung Y-K, Doh W-R, Jung W, Park S-B (1999) Applying object modelling technique to automated visual inspection of automotive compressor parts omission. Int J Mach Tools Manuf 39(11):1779–1792, ISSN: 08906955

Kantola K, Tenno R (2009) Machine vision in detection of corrosion products on SO_2 exposed ENIG surface and an in situ analysis of the corrosion factors. J Mater Process Technol 209(5):2707–2714, ISSN: 09240136

Kaplan G (1991) Technology 1991: Industrial electronics. IEEE Spectr 28(1):59–60, ISSN: 00189235

Kaplan G (1996) Technology 1996: Industrial electronics. IEEE Spectr 33(1):96–100, ISSN: 00189235

Karantalis G, Batchelor BG (1998) Prototyping machine vision software on the World Wide Web. Proc SPIE Int Soc Opt Eng 3521:304–311, ISSN: 0277786X

Karkanis S, Metaxaki-Kossionides C, Dimitriadis B (1990) A machine-vision quality inspection system for textile industries supported by parallel multitransputer architecture. Microproces Microprogramm 28(1–5):247–252, ISSN: 01656074

Karkanis S, Tsoutsou K, Metaxaki-Kossionidis C, Dimitriadis B (1991) An on-line quality inspection system for textile industries, Melecon, pp 1255–1259, ISBN: 0879426551

Kassler M (1990) Robotics and prawn-handling. Robotica 8(pt 4):299–301, ISSN: 02635747

Kavdir I, Guyer DE (2003a) Apple grading using fuzzy logic [Bulanik mantik kullanarak elma siniflama]. Turk J Agric For 27(6):375–382, ISSN: 1300011X

Keeler R (1985a) Machine vision helps to acquire the image of quality. Electron Packaging Prod 25(1):116–123, ISSN: 00134945

Keeler R (1985b) Vision guides machines in automated manufacturing. Electron Packaging Prod 25(5): 58–65, ISSN: 00134945

Kellet P (2008) Economic challenges for machine vision. Photon Spectra 42(5):61–62, ISSN: 07311230

Kelley M (2004) A self-learning machine vision system. Proc SPIE Int Soc Opt Eng 5263:66–76, ISSN: 0277786X

Ken C, Zaniewski J, Zhao P, Yang R (2008) 2D image based sieving for particle aggregate gradation. J Electron 25(2):277–282, ISSN: 02179822

Kent EW, Shneier MO (1985) Eyes for automatons. IEEE Spectr 23(3):37–45, ISSN: 00189235

Ker J-I, Chen FF, Yang Z-J (1993) Flexible inspection of machined parts with complicated circular contours. Proceedings of the industrial engineering research conference, pp 679–683, ISBN: 0898061326

Kerr D, Shi F, Brown N, Jackson M, Parkin R (2004) Quality inspection of food packaging seals using machine vision with texture analysis. Proc Inst Mech Eng B J Eng Manuf 218(11):1591–1599, ISSN: 09544054

Khoshroo A, Keyhani A, Rafiee S, Zoroofi RA, Zamani Z (2009) Pomegranate quality evaluation using machine vision. Acta Horti 818:347–352, ISSN: 05677572

Kim G-H, Kim S-W, Lim S-G (2000) Automatic inspection of geometric accuracy of optical fiber ferrules by machine vision. Proc SPIE Int Soc Opt Eng 4222: 39–43, ISSN: 0277786X

Kim H, Yang H, Lee K (2006) Software architecture and subclassing technique for semiconductor manufacturing machines. 2nd international conference on information and automation, ICIA 2006, art. no. 4250231, pp 325–330, ISBN: 1424405556; 9781424405558

Kim S-M, Lee Y-C, Lee S-C (2006) Vision based automatic inspection system for nuts welded on the support hinge. SICE-ICASE international joint conference, art. no. 4109209, pp 1508–1512, ISBN: 8995003855; 9788995003855

Kim HT, Lee KW, Yang HJ, Kim SC (2008) A self-learning method for automatic alignment in wafer processing. Proceedings of the 7th international Caribbean conference on devices, circuits and systems, ICCDCS, art. no. 4542650, ISBN: 9781424419579

Kim HT, Kang SB, Kang HS, Cho YJ, Park NG, Kim JO (2009) Optical distance control for a multi focus image in camera phone module assembly. ISOT 2009 – International symposium on optomechatronic technologies, art. no. 5326098, pp 52–58, ISBN: 9781424442102

Kimberlin JA (1983) Use of machine vision for PCB inspection. Electri-onics 29(1):23–25, ISSN: 07454309

King T (2003) Vision-in-the-loop for control in manufacturing. Mechatronics 13(10 SPEC): 1123–1147, ISSN: 09574158

King B, Krishnan A (2006) Look within. InTech 53(9): 32–40, ISSN: 0192303X

King DV, Hudson DL (1983) How machine vision tells the difference. Photon Spectra 17(6):62–64, 66, ISSN: 07311230

King WE, Drayer TH, Conners RW, Araman P (1996) Using MORRPH in an industrial machine vision system. Proceedings of the IEEE symposium on FPGAs for custom computing machines, pp 18–26, ISSN: 10823409

Kipman Y (1995) Image quality metrics for digital image capture devices. Proceedings of the IS&T annual conference, pp 456–459

Kipman Y (1995) Image quality metrics for printers/plotters. Proceedings of the IS&T's technical symposium on prepress, proofing & printing, pp 39–42

Kipman Y (1996) Image quality metrics for printers/plotters. Final program and proceedings – IS and T/SID color imaging conference, pp 134–138; Chan J, Braggins D (1996) Untiring eyes, Manuf Eng 75(5):233–235, ISSN: 09569944

Kipman Y, Wolin D (2001) Scanning machine vision for fiber optic connector inspection. Proc SPIE Int Soc Opt Eng 4567:216–223, ISSN: 0277786X

Kirgis F-P (2006) Robotics in consumer industries new applications in food, pharma and healthcare, VDI Berichte (1956), p 119, ISSN: 00835560

Kise M, Park B, Lawrence KC, Windham WR (2007) A compact multispectral imaging system for online poultry contaminant inspection. ASABE annual international meeting, Technical papers, 7 book, 8 p; Armstrong BSR, Veettil SKP (2007) Soft synchronization: Synchronization for network-connected machine vision systems. IEEE Trans Ind Info 3(4):263–274, ISSN: 15513203

Knight A (1996) Laser marking looks to machine vision. Laser Focus World 32(7):5, ISSN: 07402511

Knyaz VA, Vizilter YV (2001) Method for 3D non-contact measurements of cut trees package area. Proc SPIE Int Soc Opt Eng 4189:276–285, ISSN: 0277786X

Kochan A (2001) Hanover round up of machine vision and robotics. Ind Robot 28(6):483–488, ISSN: 0143991X

Kochan A (2002a) A new wave of synchronous robots. Assembly Autom 22(3):223–225, ISSN: 01445154

Kochan A (2002b) Robots great and small at Hanover. Assembly Autom 22(3):235–238, ISSN: 01445154

Koh TK, Miles NJ, Morgan SP, Hayes-Gill BR (2009) Improving particle size measurement using multi-flash imaging. Miner Eng 22(6):537–543, ISSN: 08926875

Kohkonen KE (1985) Vision inspection sensitivity. ASEE Ann Conf Proc 3:1305–1308, ISSN: 01901052

Kona SR, Foster JW, Pignatiello JJ (1993) Statistical process control of continuous materials using on-line machine vision. Proceedings of the industrial engineering research conference, pp 107–111, ISBN: 0898061326

Konnai K, Hoshino Y (2001) Method of capturing image of object passing through focus area. Proc SPIE Int Soc Opt Eng 4308:86–94, ISSN: 0277786X

Kosner J (1986) Performance evaluation of machine vision inspection of assembled printed circuit boards, pp 3.15–3.31, ISBN: 0872632296

Krieger M (1993) Multiactivity paradigm for the design and coordination of FMSs. Comput Integr Manuf Syst 6(3):195–203, ISSN: 09515240

Kruppa VD (1992) MCR for high volume fine pitch IC packages. National electronic packaging and production conference – Proceedings of the technical program (West and East), 3, pp 1229–1230, ISSN: 04700155

Kubel (ed) (1998) Machine vision: Eyes for industry. Manuf Eng 120(4):42, 44, 46, 49, 50–51, ISSN: 03610853

Kukkonen S, Kälviäinen H, Parkkinen J (2001) Color features for quality control in ceramic tile industry. Opt Eng 40(2):170–177, ISSN: 00913286

Kuo C-H, Lee M-Y, Chiu Y-S, Huang C-S, Hung K-F (2003) Development of computer aided retained auricular prosthesis surgery system – A clinical case study on microtia. Proc IEEE Int Conf Syst Man Cybernet 3:2920–2925, ISSN: 08843627

Kurada S, Bradley C (1997a) A machine vision system for tool wear assessment. Tribol Int 30(4):295–304, ISSN: 0301679X

Kurada S, Bradley C (1997b) A review of machine vision sensors for tool condition monitoring. Comput Ind 34(1):55–72, ISSN: 01663615

Kutila M (2006) Methods for machine vision based driver monitoring applications. VTT Publications (621), pp 1–82, ISSN: 12350621, ISBN: 9513868753; 9789513868758

Kwok SK, Lee WB (1995) The development of a machine vision system for adaptive bending of sheet metals. J Mater Proces Tech 48(1–4):43–49, ISSN: 09240136

Kwon Y, Chiou R, Rauniar S, Sosa H (2005) Performance characterization of precision micro robot using a machine vision system over the Internet for guaranteed positioning accuracy. Proc SPIE Int Soc Opt Eng 5999, art. no. 599906, ISSN: 0277786X

Lackey JD, Gerth DM, Longenbaker WE (1987) Machine vision for process improvement: A design experience, pp 660–666

Lahajnar F, Bernard R, Pernuš F, Kovačič S (1998) Automated visual inspection of cooking plates. Proc SPIE Int Soc Opt Eng 3521:260–267, ISSN: 0277786X

Lam EY (2008) Compact and thin multi-lens system for machine vision applications. Proc SPIE Int Soc Opt Eng 6813, art. no. 681305, ISSN: 0277786X, ISBN: 9780819469854

Langston KF (1996) Machine vision inspection – Is the time right? National electronic packaging and production conference – Proceedings of the technical program (West and East), 2, pp 565–584, ISSN: 04700155

Lapidus SN (1984) New techniques for industrial vision inspection, pp 4.1–4.13, ISBN: 0872631400

Lapidus SN (1985) Machine vision: The unblinking link to inspection and positioning systems. Plant Eng (Barrington, IL) 39(15 A):78–83, ISSN: 0032082X

Lapidus SN (1986) Advanced gray scale techniques improve machine vision inspection. Robot Eng 8(6):22–25, ISSN: 08880816

Lapidus SN (1987) Machine vision: Eyes for the intelligent factory. Prod Eng 34(4):52–55, ISSN: 01461737

Larson M (1998) Auto industry sees: Future of vision inspection. Quality 37(6):53, ISSN: 03609936

Lecky N (1998) Board level industrial imaging/machine vision markets now: The impact of getting easier. Adv Imag 13(2):10–X1, ISSN: 10420711

Lee Jr JJ (1986) Machine vision applications in microelectronic production, pp 11–15, ISBN: 0930815165

Lee J (1987) Applying machine vision to robotic automation, pp 723–727

Lee K-M (1991) Flexible part-feeding system for machine loading and assembly. Part I. A state-of-the-art survey. Int J Prod Econ 25(1-3):141–153, ISSN: 09255273

Lee D-J (2000) Color space conversion for linear color grading. Proc SPIE Int Soc Opt Eng 4197, pp 358–366, ISSN: 0277786X

Lee K-M, Li Da-Rein (1991) Principle and applications of retroreflective vision sensing for discrete part-presentation. J Robot Syst 8(1):55–73, ISSN: 07412223

Lee K-M, Blenis R (1994) Design concept and prototype development of a flexible integrated vision system. J Robotic Syst 11(5):387–398, ISSN: 07412223

Lee K-M, Zhou Z, Blenis R, Blasch E (1995) Real-time vision-based tracking control of an unmanned vehicle. Mechatronics 5(8):973–991, ISSN: 09574158

Lee K-M, Sobh TM (2001) Visionary prototyping. IEEE Rob Autom Mag 8(3):15–24, ISSN: 10709932

Lee D-J, Lane RM, Chang G-H (2001) Three-dimensional reconstruction for high-speed volume measurement. Proc SPIE Int Soc Opt Eng 4189:258–267, ISSN: 0277786X

Lee JY, Yoo SI (2004) Automatic detection of region-mura defect in TFT-LCD. IEICE Trans Info Syst E87-D(10):2371–2378, ISSN: 0277786X, ISSN: 09168532

Lee D-J, Xu X, Lane RM, Zhan P (2004) Shape analysis for an automatic oyster grading system. Proc SPIE Int Soc Opt Eng 5606, art. no. 05, pp 27–36, ISSN: 0277786X

Lee D-J, Schoenberger R, Archibald J, McCollumS (2008) Development of a machine vision system for automatic date grading using digital reflective near-infrared imaging. J Food Eng 86(3):388–398, ISSN: 02608774

Legrand A-C, Suzeau P, Renier E, Truchetet F, Gorria P, Meriaudeau F (2001) Machine vision systems in the metallurgy industry. J Electron Imag 10(1):274–282, ISSN: 10179909

Leika E (1999) Robot guidance with a photogrammetric 3-D measuring system. Ind Robot 26(2):105–108, ISSN: 0143991X

Lemstrom GF (1995) Color line-scan technology in industrial applications. Proc SPIE Int Soc Opt Eng 2588:190–199, ISSN: 0277786X, ISBN: 0819419524

Lepistö L, Kunttu I, Lähdeniemi M, Tähti T, Nurmi J (2007) Grain size measurement of crystalline products using maximum difference method. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), 4522 LNCS, pp 403–410, ISSN: 03029743, ISBN: 9783540730392

Lerch A, Chetverikov D (1991) Knowledge-based line-correction rules in a machine-vision system for the leather industry. Eng Appl Artif Intell 4(6):433–438, ISSN: 09521976

Lerner EJ (1986) Machines sharpen their vision. Aerosp Am 24(11):36–39, ISSN: 0740722X

Les CB (2007) Market forecast: Positive for machine vision. Photon Spectra 41(6):64, ISSN: 07311230

Les CB (2009) Machine vision market: Weathering the storm. Photon Spectra 43(4):31–32, ISSN: 07311230

Leung C-C, Chan K-S, Chan H-M, Tsui W-K (2005) A new approach for image enhancement applied to low-contrast-low-illumination IC and document images. Pattern Recognit Lett 26(6):769–778, ISSN: 01678655

Lev ZH (1999) Machine vision in the packing house – Putting neural networks to work. Proc SPIE Int Soc Opt Eng 3543:82–79, ISSN: 0277786X

Lewotsky K (2006) Vive le ROI. Adv Imag 21(4):26–28, ISSN: 10420711

Lewotsky K (2009) Manufacturing on the move. Adv Imag 24(9):8–11, ISSN: 10420711

Lex DG (1985) Vision for vision, pp 229–234; Leonard PF, Svetkoff DJ, Kelley RW, Rohrer DK Machine vision applications in electronic manufacturing, pp 5.99–5.116. ISBN: 0872631737

Ley D (2007) Machine vision in Germany and Europe. Photon Spectra 41(1):68–69, ISSN: 07311230

Li Ze-Nian, Zhang D (1993) Real-time line-based motion stereo. Proc IEEE Int Conf Robot Autom 2:367–372, ISSN: 10504729, ISBN: 0818634529

Li W, Tansel IN, Arch A (1996) Spur gear inspection with a machine vision system. Am Soc Mechan Eng Petrol Div Publ PD 74(2):245–250

Li X, Fang S-L, Zhang Y-C (2007) The study on clustering algorithm of the underwater acoustic sensor networks. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430719, pp 78–81, ISBN: 1424413583; 9781424413584

Lian B, Jiang L, McGrath JJ, Jaranson J (2000) Quantitative determination of morphological features of triaxially braided composites by the use of machine vision. Compos Sci Technol 60(2):159–166, ISSN: 02663538

Liang Z (1996) Teaching robot vision in manufacturing technology. ASEE annual conference proceedings, pp 3053–3057, ISSN: 01901052

Liatsis P, Goulermas JY, Zeng X-J, Milonidis E (2009) A flexible visual inspection system based on neural networks. Int J Syst Sci 40(2):173–186, ISSN: 00207721

Liberatore MJ, O'Neill J (1985) Planning a successful robotics implementation. J Oper Manage 5(3): 247–257, ISSN: 02726963

Lifshits M, Rivlin E, Rudzsky M (2004) Image-based navigation on a chip. Conference Record. IEEE Instrument Measure Technol Conf 1:504–509, ISSN: 10915281

Lin H-D, Chung C-Y, Lin W-T (2008) Principal component analysis based on wavelet characteristics applied to automated surface defect inspection. WSEAS Trans Comput Res 3(4):193–202, ISSN: 19918755

Linder KD, Baumgart C, Creager J, Heinen B, Troupe T, Meyer D, Carr J, Quint J(1998) Automated detection of Karnal bunt teliospores. Proc SPIE Int Soc Opt Eng 3306:80–87, ISSN: 0277786X

Lindgren J (1985) Different techniques for automatic identification in fms installations, pp 175–178, ISBN: 0903608979

Ling PP, Ruzhitsky VN, Kapanidis AN, Lee T-C (1996) Correlation between color machine vision and colorimeter for food applications, ACS symposium series, 631, pp 253–278, ISSN: 00976156

Litchfield JB, Reid JF, Schmidt SJ (1994) Machine vision microscopy and magnetic resonance microscopy. Food Technol 48(6):163–166, ISSN: 00156639

Little J (1999) Random access CMOS imaging technology applied to an industrial machine vision problem. IEE Colloquium (Digest) (143):37–38, ISSN: 09633308

Liu BC, Shi XQ (1997) Reverse-multiplicate method of moire interferometry and application to measure deformation of a microelectronic package. Proc SPIE Int Soc Opt Eng 2921:512–517, ISSN: 0277786X, ISBN: 0819423238

Liu Y, Rodrigues MA (2000) Geometrical pose and structural estimation from a single image for automatic inspection of filter components. Proc SPIE Int Soc Opt Eng 3966:323–331, ISSN: 0277786X

Liu JJ, MacGregor JF (2004) Estimation of visual quality of injection-molded polymer panels, AIChE annual meeting, conference proceedings, pp 7509–7522; Wang K, Tsung F (2004) SPC for a data-rich environment using profile techniques. IIE annual conference and exhibition 2004, pp 301–306; Lam EY (2004) Robust minimization of lighting variation for real-time defect detection. Real-Time Imag 10(6):365–370, ISSN: 10772014

Liu JJ, MacGregor JF (2005) Modeling and optimization of product appearance: Application to injection-molded plastic panels. Ind Eng Chem Res 44(13):4687–4696, ISSN: 08885885

Liu JJ, MacGregor JF (2006) Estimation and monitoring of product aesthetics: Application to manufacturing of "engineered stone" countertops. Mach Vision Appl 16(6):374–383, ISSN: 09328092

Liu PC-H, Trivedi A, Lee BP (1996) Implementing automatic data collection and identification laboratory, Southcon conference record, pp 222–224

Liu X, Luo X, Li S, Zhao H (2006) Integration method research for the detection of moving multi-targets in complex dynamic scenes. Proceedings of the world congress on intelligent control and automation (WCICA), 2, art. no. 1714004, pp 10230–10234, ISBN: 1424403324; 9781424403325

Lizotte T, Ohar O (2004) The use of holographic and diffractive optics for optimized machine vision illumination for critical dimension inspection. Proc SPIE Int Soc Opt Eng 5265:16–23, ISSN: 0277786X

Lloyd CJ (1998) Towards the development of more effective surface inspection methods. Technical paper – Society of manufacturing engineers. FC, 98 (268), pp X1–11, ISSN: 01611844

Lloyd CJ (1998) Towards the development of more effective surface inspection methods. Technical paper – Society of manufacturing engineers. FC, 12 p, ISSN: 01611844

Locht P, Thomsen K, Mikkelsen P (1997) Full color image analysis as a tool for quality control and process development in the food industry. Paper. Am Soc Agric Eng 1:15, ISSN: 01450166

Looking ahead (2007) Adv Imag 22(10):10–14, ISSN: 10420711

Lopez-Juarez I, Corona-Castuera J, Peña-Cabrera M, Ordaz-Hernandez K (2005) On the design of intelligent robotic agents for assembly. Inf Sci 171(4):377–402, ISSN: 00200255

Lorinez JA (1994) Vision thing. Machines provide it for control. Tool Prod 60(6):3, ISSN: 00409243

Losty JA (1982) Development environment for industrial vision systems. IEE Colloquium (Digest) (1982 / 77):1/1–1/2, ISSN: 09633308

Lotufo RA, Taube-Neto M, Conejo E, Hoyos FJ (1999) Using machine vision to estimate bird weight in the poultry industry. Proc SPIE Int Soc Opt Eng 3652:43–49, ISSN: 0277786X

Lu RF (2004) Design and configuration of machine vision robotic cells in a manufacturing system. Proceedings of the ASME design engineering technical conference, 2 A, pp 621–626; Mariño P, Pastoriza V, Santamaría M, Martínez E (2004) Fuzzy inference system to inspect coating in canmaking industry. Proceedings of the IEEE international conference on industrial technology, 3, pp 1144–1149

Lu K (2006) Perfect vision. Manuf Eng 85(5):42–45, ISSN: 09569944

Lu S, Zhang X (2009) Intelligent inspection of soldered joint based on artificial neuron network. Hanjie Xuebao/Trans China Weld Inst 30(5):57–60, ISSN: 0253360X

Lu Y, Chen TQ, Chen J, Zhang J, Tisle A (2001) Machine vision systems using machine learning for industrial product inspection. Proc SPIE Int Soc Opt Eng 4567:161–170, ISSN: 0277786X

Lu Y, Gao Y, Chen H, Chen Z (2007) Intelligent spectral design and colorimetric parameter analysis for light-emitting diodes. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430728, pp 118–122, ISBN: 1424413583; 9781424413584

Lyons C (1996) Robots on the move. Process Control Eng 49(2):12, ISSN: 08168148

Lubofsky E (1999) Machine vision streamlines robotic handling of engine parts. Robot World 17(2): 20–25, ISSN: 07377908

Lubofsky E (2000) Machine vision and flexible manufacturing. Sensors (Peterborough, NH) 17((4):78–80, ISSN: 07469462

Lubofsky E (2001) Machine vision – Moving beyond the end of the line. Sensors (Peterborough, NH) 18(8):40–42–43, ISSN: 07469462

Ma R, Zhou H, Sun P, Hou S (2007) dSPACE-based PID controller for a linear motor driven inverted pendulum. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430717, pp 68–72, ISBN: 1424413583; 9781424413584

MacCalla J (1986) Robotics in industry. Wescon Conference Record, 4 p

Macedo J, Colvin K, Waldorf D (2006) Machine vision course for manufacturing engineering undergraduate students. J Manuf Syst 24(3):256–265, ISSN: 02786125

MacGillivray G (2009) Keeping an eye on the bad guys with GigE. Photon Spectra 43(2):46–47, ISSN: 07311230

Machine Vision Applications in Industrial Inspection II (1994) Proc SPIE Int Soc Opt Eng 2183:333, ISSN: 0277786X, ISBN: 0819414786

Machine Vision Applications, Architectures, and Systems Integration IV (1995) Proc SPIE Int Soc Opt Eng 2597:325, ISSN: 0277786X, ISBN: 0819419613

Machine vision systems multiplying in the automotive industry (2001) IEEE Robot Autom Magazine 8(2):59, ISSN: 10709932

Machine vision systems (2009) Springer series in optical sciences, 136, pp 31–39, ISSN: 03424111, ISBN: 9783540719267

Mackie BR (1994) Video feedback closed loop (VFCL^{TM}) process control systems, Technical paper – Society of manufacturing engineers. MS, pp 1–23, ISSN: 01616382

Mackrory JD (1995) Mark, impact of new technology in machine vision. Sensor Rev 15(1):8–11, ISSN: 02602288

Madhuri P, Nagesh AS, Thirumalaikumar M, Varghese Z, Varun AV (2009) Performance analysis of smart camera based distributed control flow logic for machine vision applications. Proceedings of the IEEE international conference on industrial technology, art. no. 4939499, ISBN: 1424435064; 9781424435067

Mahony Tom, Gromala Jerry (1989) Diode lasers move into machine vision. Lasers Optronics 8(11):5, ISSN: 08929947

Majumder A (2009) Image processing algorithms for improved character recognition and components inspection. World Congress on Nature and Biologically Inspired Computing, NABIC 2009 – Proceedings, art. no. 5393389, pp 531–536, ISBN: 9781424456123

Malamas EN, Petrakis EGM, Zervakis M, Petit L, Legat J-D (2003) A survey on industrial vision systems, applications and tools. Image Vis Comput 21(2):171–188, ISSN: 02628856

Malchow DS, Brubaker RM, Hansen MP (2008) Development of linear array ROIC for InGaAs detector arrays with wavelength response to 2.5 microns for NIR spectroscopy and machine vision. Proc SPIE Int Soc Opt Eng 6940, art. no. 69402V, ISSN: 0277786X, ISBN: 9780819471314

Manickavasagan A, Sathya G, Jayas DS (2008) Comparison of illuminations to identify wheat classes using monochrome images. Comput Electron Agric 63(2):237–244, ISSN: 01681699

Manji JF (1986) Automation at gm assembly plants boosts quality, cuts costs. Prod Eng 33(12):16–17, 19, ISSN: 01461737

Mansoory MS, Tajik HN, Pashna M (2009) Surface defect isolation in ceramic tile based on texture feature analysis using radon transform and FCM. International Conference on Signal Processing Systems, ICSPS 2009, art. no. 5166752, pp 85–90, ISBN: 9780769536545

Marino P, Dominguez MA (1997) Artificial vision for automated manufacturing systems in communications industry. IEEE symposium on emerging technologies and factory automation, ETFA, pp 421–426

Marino P, Dominguez MA (1997) On-line automated inspection based on image digitization boards. IECON proceedings (industrial electronics conference), 3, pp 1228–1233

Marino P, Dominguez MA (1998) Image processing and automated testing in flexible manufacturing systems. Proceedings of the SICE annual conference, pp 1121–1126

Marino P, Dominguez MA (1998) Machine vision system for total quality control of SMT electronic cards. Proceedings of the IEEE international conference on intelligent processing systems, ICIPS, 2, pp 1427–1431

Marino P, Dominguez M, Alonso M (1999a) Inspection of steel sheets based on CCD image sensors. IEEE international conference on multisensor fusion and integration for intelligent systems, pp 68–73

Marino P, Dominguez MA, Alonso M (1999b) Machine-vision based detection for sheet metal industries. IECON Proc (Ind Electr Conf) 3:1330–1335

Mariño P, Pastoriza V, Santamaría M, Martínez E (2004) Can end inspection using neuro-fuzzy modeling. IEEE conference on cybernetics and intelligent systems, pp 925–929, ISBN: 0780386442; 9780780386440

Mariño P, Pastoriza V, Santamaría M, Martínez E (2005) Can end inspection with image processing and fuzzy modelling. Proceedings – International conference on computational intelligence for modelling, control and automation, CIMCA 2005 and international conference on intelligent agents. Web Technologies and Internet, 2, art. no. 1631545, pp 666–671, ISBN: 0769525040; 9780769525044

Mariño P, Pastoriza V, Santamaría M, Martínez E (2005) Easy open can end inspection using UV black light. Proceedings of the 2005 international conference on computer vision, VISION'05, pp 178–184, ISBN: 9781932415650

Mariño P, Pastoriza V, Santamaría M, Martínez E (2005) Fuzzy image processing in quality control application. Proceedings – Sixth international conference on computational intelligence and multimedia applications, ICCIMA 2005, art. no. 1540703, pp. 55–61, ISBN: 0769523587; 9780769523583

Mariño P, Pastoriza V, Santamaría M, Martínez E (2005) Image processing application with a TSK fuzzy model. Lecture notes in artificial intelligence (subseries of lecture notes in computer science), 3614 (PART II), pp 950–960, ISSN: 03029743

Mariño P, Pastoriza V, Santamaría M, Martínez E (2006a) Fuzzy system to inspect can end repair coating in canmaking industry [Sistema borroso de inspección del rebarnizado de tapas en la industria metalgráfica]. Informacion Tecnologica 17(4):4, ISSN: 07168756

Mariño P, Pastoriza V, Santamaría M, Martínez E (2006) Image processing application with a TSK fuzzy model. Lecture notes in computer science (including subseries Lecture notes in artificial intelligence and lecture notes in bioinformatics), 3614 LNAI, pp 950–960, ISSN: 03029743, ISBN: 9783540283317

Marokkey SR, Tay CJ, Shang HM, Asundi AK (1997) Time delay and integration imaging for inspection and profilometry of moving objects. Opt Eng 36(9):2573–2578, ISSN: 00913286

Marszalec E, Pietikäinen M (1996) Some aspects of RGB vision and its applications in industry. Int J Pattern Recognit Artif Intell 10(1):55–72, ISSN: 02180014

Masi CG (2006) At IMTS 2006, demos showed how simulation software also helps robotics with sorting, positioning, and high-speed handoffs. Control Eng 53(10), ISSN: 00108049

Massen R (2001) Aesthetic measures. Int Carpet Bull September:18–21, ISSN: 02682966

Massen R (2006) Multisensoric tile inspection. CFI Ceramic Forum Int 83(6–7):E26–E28, ISSN: 01739913

Matthews SJ (2004) Vision on the factory floor. Laser Focus World 40(9):94–97, ISSN: 10438092

Maves S (2007) Open your eyes. Motion System Design 49(3):46–49, ISSN: 15371794

Maxwell A (1998) M^{2}VIP – Mechatronics walking and swimming down under. Ind Robot 25(2):109–111, ISSN: 0143991X

Mazor B (2001) International marketing issues for image processing and vision. Adv Imag 16(1):10–14, 16, 57, ISSN: 10420711

Mazor B (2002) Integrating machine vision with other factory automation technologies. Adv Imag 17(5):18–23, ISSN: 10420711

Mazor B (2004) A robust look at software trends. Adv Imag 19(10):27–29, ISSN: 10420711

McConnell RK (1996) Train-by-show in color based assembly and packaging inspection. Proc SPIE Int Soc Opt Eng 3205:210–217, ISSN: 0277786X

McCracken WT (1985) Integrating a machine vision system into the manufacturing process, pp 1.36–1.46, ISBN: 0872631737

McGarry SL (1984) Acceptable quality levels are not acceptable any more, autofact. Conference proceedings, pp 15.1–15.13, ISBN: 0872631613

McVey ES, Davis VL, Inigo RM (1987) Fault tolerant array sensor design. Conference proceedings – IEEE Southeastcon, pp 221–224, ISSN: 07347502

McWalter K (1989a) Future trends for vision systems in surface mount placement. Surface Mount Technol 3(2):37–39, ISSN: 08933588

McWalter K (1989b) Vision technology boosts SM device placement on hybrids. Hybrid Circuit Technol 6(6):11–14, ISSN: 07471599

Medina R, Gayubo F, González LM, Olmedo D, Gómez J, Zalama E, Perán JR (2008) Surface defects detection on rolled steel strips by Gabor filters, VISAPP 2008 – 3rd international conference on computer vision theory and applications. Proceedings, 1, pp 479–485, ISBN: 9789898111210

Mehta SI (1994) Machine vision system for inspecting hydraulic hose assemblies. Am Soc Mech Eng Design Eng Div Publ DE 73:79–83

Melamed D (1997) New look. Robot World 15(1):48–50, ISSN: 07377908

Meledin VG (2009) Informatics of optoelectronic measurements: Science and innovative industrial technologies. J Eng Thermophys 18(2):99–128, ISSN: 18102328

Meledin VG, Bakakin GV, Naumov IV, Pavlov VA, Sotnikov VV (2003) Real time machine vision system for non-contact measurements of the masses of free falling hot glass drops. Proc SPIE Int Soc Opt Eng 5134:139–148, ISSN: 0277786X

Melikian S (1999) Geometric searching improves machine vision. Lasers Optronics 18(7):13–15, ISSN: 08929947

Melin S, Sokhansanj S, Bi X, Lim CJ, Pordesimo LO, Igathinathane C (2009) Pellet industry airborne dust particles size and size distribution using machine vision ImageJ plugin. American society of agricultural and biological engineers annual international meeting 2009, 5, pp 3113–3128, ISBN: 9781615673629

Mendonça PRS, Kaucic R (2008) Single view metrology: A practical example. 2008 IEEE workshop on applications of computer vision, WACV, art. no. 4544020, ISBN: 1424419131; 9781424419135

Merat FL, Barendt NA, Quinn RD, Causey GC, Newman WS, Jr V, Virgilio B, Podgurski A, Kim Y, Ozsoyoglu G, Jo Ju-Yeon (1997) Advances in agile manufacturing. Proc IEEE Int Conf Robot Autom 2:1216–1222, ISSN: 10504729

Meriaudeau F (2007) Real time multispectral high temperature measurement: Application to control in the industry. Image Vis Comput 25(7):1124–1133, ISSN: 02628856

Meriaudeau F, Legrand A-C (2000) Machine vision systems in metallurgy industry. Proc SPIE Int Soc Opt Eng 3966:228–237, ISSN: 0277786X

Meriaudeau F, Chaouki A (2001) Machine vision system for the inspection of laminated tubes. Proc SPIE Int Soc Opt Eng 4301:226–235, ISSN: 0277786X

Meriaudeau F, Lavallée G, Fauvet E (2002) Machine vision prototype for defect detection on metallic tubes. Proc SPIE Int Soc Opt Eng 4664:190–197, ISSN: 0277786X

Merva J, Kellett P (2007) Current trends in machine vision. Photon Spectra 41(1):65–67, ISSN: 07311230

Meyer JD (1984) Role of machine vision in flexible manufacturing systems, pp 2.7–2.16, ISBN: 0872631400

Miller RK (1983) Machine vision for robotics and automated inspection. Mach vision for rob and autom inspect, 528 p, ISBN: 0896710467

Miller FW (1987) Revised look at machine vision trends. Manuf Syst 5(3):40, 42–43, 45, ISSN: 0748948X

Miller JW, Shabestari BN, Sweney R (1995) Determination of geometric stability of glass substrates using machine vision. Proc SPIE Int Soc Opt Eng 2597:164–167, ISSN: 0277786X, ISBN: 0819419613

Millman M (2004) The PC for high-end surface inspection: Practical or problematic? IEE Comput Control Eng 15(4):15–18, ISSN: 09563385

Mills RN, Miller TW (2007) Substrate alignment for single and multilayer ink jet materials deposition in digital fabrication systems. International conference on digital printing technologies, pp 877–881, ISBN: 0892082739; 9780892082735

Min L, Zheng LX (2007) Contour detection based on gabor filter and directional DoG filter. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430740, pp 185–190, ISBN: 1424413583; 9781424413584

Mintchell GA (1999) Vision systems see new technology, bring lower prices. Control Eng 46(6):4, ISSN: 00108049

Mintchell GA (2002) Vision systems eyes in the factory. Control Eng 49(8):20–24, ISSN: 00108049

Mishra S, Gharpure DC (2001) Cost effective tactile sensing system for object recognition. Proc SPIE Int Soc Opt Eng 4301:175–180, ISSN: 0277786X

Mitchell TA, Bowden R, Sarhadi M (1999) Automated visual inspection of dry carbon-fibre reinforced composite preforms. Proceedings of the institution of mechanical engineers, Part G. J Aerospace Eng 213(6):377–386, ISSN: 09544100

Moeslund TB, Kirkegaard J (2005) Pose estimation of randomly organized stator housings. Lect Notes Comput Sci 3540:679–688, ISSN: 03029743

Mohan AL, Jayas DS, White NDG, Karunakaran C (2004) Automation for unloading grain. ASAE annual international meeting 2004, pp 4019–4025; Lydén S, Kälviäinen H, Nykänen J (2004) Surface roughness estimation of shot blasted steel bars using machine vision. Proc SPIE Int Soc Opt Eng 5608, art. no. 26, pp 278–289, ISSN: 0277786X

Moni MA, Shawkat Ali, ABM (2009) HMM based hand gesture recognition: A review on techniques and approaches. Proceedings – 2nd IEEE international conference on computer science and information technology, ICCSIT 2009, art. no. 5234536, pp 433–437, ISBN: 9781424445196

Morales O (2001) Automatic part recognition and checkout for color automobile door panels. Adv Imag 16(3):11, ISSN: 10420711

Morrisey J, Cobb J (1985) Automatic visual testing equipment designed for loaded pcb inspection. Electrionics 31(3):17–20, ISSN: 07454309

Moskal FR (1991) Automated combustor hole drilling and inspection system. Ann Forum Proc Am Helicopter Soc 1:391–398, ISSN: 07334249

Motavalli S, Valenzuela J (1995) Building a 3D wireframe CAD model for an existing prismatic part. Proc SPIE Int Soc Opt Eng 2598:327–333, ISSN: 0277786X

Motavalli S, Valenzuela J (1998) A system for reverse engineering of prismatic parts using orthographic images. Int J Computer Integr Manuf 11(2): 103–110, ISSN: 0951192X

Motavalli S, Suharitdamrong V, Alrashdan A (1998) Design model generation for reverse engineering using multi-sensors. IIE Trans (Inst Ind Eng) 30(4):357–366, ISSN: 0740817X

Movich RC (1984) Intelligent robotic riveting cell. Technical paper – Society of manufacturing engineers, 19 p

Mulliner J (2001) Making measurements with vision. Adv Imag 16(10):28–30, 66, ISSN: 10420711

Mumzhiu A (1998) Basic design principles of colorimetric vision systems. Proc SPIE Int Soc Opt Eng 3521:179–183, ISSN: 0277786X

Mumzhiu A (1999) Application of vision systems for color and appearance measurements in industrial inspection. Proc SPIE Int Soc Opt Eng 3652: 102–109, ISSN: 0277786X

Munden DL, Norton-Wayne L (1988) Machine vision in textile manufacture; Riley FJ (1988) Evolution of automatic assembly. Assembly Eng 31(1):36–38, ISSN: 00045063

Muracciole V, Plainchault P, Mannino M-R, Bertrand D, Vigouroux B (2007) Methodology for creating dedicated machine and algorithm on sunflower counting. Proc SPIE Int Soc Opt Eng 6761, art. no. 67610X, ISSN: 0277786X, ISBN: 9780819469212

Murphy FP (1987) Can machine vision do the job for you? Chilton's Instruments Control Syst 60(10):53–56, ISSN: 01640089

Murray LA, Cooper JE (1985) Intelligent vision systems: Today and tomorrow. Test Measure World 5(2):6, between p 76 and 90, ISSN: 07441657

Murtagh F, Qiao X, Walsh P, Basheer PAM, Crookes D, Long A (2005) Grading of construction aggregate through machine vision: Results and prospects. Comput Ind 56(8–9):905–917, ISSN: 01663615

Myyrä J (2002) The 4th dimension in glass measuring. Glass 79(11):381, ISSN: 00170984

Na HS (1986) Inspection of surface-mounted component assembly, pp 3.33–3.40, ISBN: 0872632296

Na HS (1987) Integration of machine vision into flexible assembly process. Proceedings of the annual control engineering conference, pp 230–234, ISBN: 0914331566

Nachimuthu A, Hoang K (1995) High-speed scanning: An improved algorithm. Proc SPIE Int Soc Opt Eng 2588:40–47, ISSN: 0277786X, ISBN: 0819419524

Nadabar SG, Hajj HH, Anbalagan RS (1995) Machine vision system for high-speed discrete parts inspection. Proc SPIE Int Soc Opt Eng 2622(2):616–624, ISSN: 0277786X, ISBN: 0819419869

Nagamatsu I, Hirakawa K, Nobuhiro K (1986) Practical method to adopt map in industrial robot controller, pp 7.83–7.97, ISBN: 0872632237

Nagarajan R, Yaacob S, Pandian P, Karthigayan M, Amin SH, Khalid M (2007) A real time marking inspection scheme for semiconductor industries. Int J Adv Manuf Technol 34(9–10):926–932, ISSN: 02683768

Nagchaudhuri A (2002) Robotics and machine vision for introduction to flexible automation to engineering undergraduates. Proceedings – Frontiers in education conference, 1, pp T2D/23–T2D/28, ISSN: 01905848

Nair M, Jayas DS (1998) Dockage identification in wheat using machine vision. Can Agric Eng 40(4):293–298, ISSN: 0045432X

Naitove MH (2004) Small molder builds a high-tech showplace. Plastics Technol 50(3):37–39, ISSN: 00321257

Nanko N, Okumura T (2009) Possibility of machine vision in the pulp and paper industry. Kami Pa Gikyoshi/Jap Tappi J 63(3):28–31, ISSN: 0022815X

Navas D (2004) Plant-floor process tracking's new look. Supply Chain Syst Magazine 24(8):10–15, ISSN: 1539865X

Neethirajan S, Karunakaran C, Jayas DS, White NDG (2007) Detection techniques for stored-product insects in grain. Food Control 18(2):157–162, ISSN: 09567135

Negin M, Kase JP (1986) Machine vision software for automatic setup for tab tape inspection for microelectonic assembly, pp 3.73–3.81, ISBN: 0872632296

Ng GS, Sim HC (1998) Recognition of partially occluded objects with back-propagation neural network. Int J Pattern Recognit Artif Intell 12(5):645–660, ISSN: 02180014

Ng H-F (2004) Automatic thresholding for defect detection. Proceedings – Third international conference on image and graphics, pp 532–535, ISBN: 0769522440

Ng H-F (2006) Automatic thresholding for defect detection. Pattern Recognit Lett 27(14):1644–1649, ISSN: 01678655

Nickols FMJ, Le Vasan M (2007) The pedagogy of creating a mechatronic product integrated with English communication skills for teaching design and innovation to engineering undergraduates. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430751, pp 246–251, ISBN: 1424413583; 9781424413584

Niles A (2007) Optics advances for every application. Adv Imag 22(4):18–21, ISSN: 10420711

Nilsson J, Ericsson M, Danielsson F (2009) Virtual machine vision in computer aided robotics. ETFA 2009 – 2009 IEEE conference on emerging technologies and factory automation, art. no. 5347003, ISBN: 9781424427284

Norton-Wayne L (1990) Automated garment inspection using machine vision, pp 374–377; Johnson C, Chapman K, Bowery J, Man X (1990) A machine vision based interactive control system applied to circuit printing on a mylar substrate. Comput Ind Eng 19(1–4):219–223, ISSN: 03608352

Norton-Wayne L (1995) Automated inspection of moving webs using machine vision. IEE Colloquium (Digest) (113):3/1–3/8, ISSN: 09633308

Norton-Wayne L, Harwood RJ (1990) Intelligent inspection. Textile Asia 21(5):137–140, ISSN: 00493554

Novini AR (1989a) Fundamentals of on-line gaging for machine vision. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–599, ISSN: 01616382

Novini AR (1989b) Fundamentals of machine vision component selection. Carbide Tool J 21(4):7–14, ISSN: 01928333

Novini A (2007) Machine vision achieves more. Quality 46(2):36–39, ISSN: 03609936

Nunes J (2004) Lockheed Martin: Vision-guided robots assist in weapon system assembly. Robot World 23(4):9–11, ISSN: 10534644

Oberly RE, Brumfield JO (1992) Flexible new optics facilities with electronics and computer systems integration in a renovated building at a smaller state university. Proc SPIE Int Soc Opt Eng 1603:137–145, ISSN: 0277786X, ISBN: 0819407321

Obstgarten M (1999) Now: Image archiving for machine vision troubleshooting. Adv Imag 14(4):48, ISSN: 10420711

O'Dell J, McWalter K (1989) Visual inspection system boosts pharmaceutical packaging process. Chilton's Instruments Control Syst 62(3):69–70, ISSN: 01640089

Oh J-K, Lee C-H (2007) Development of a stereo vision system for industrial robots. ICCAS 2007 – International conference on control, automation and systems, art. no. 4406981, pp 658–662, ISBN: 8995003871; 9788995003879

Ojala T, Pietikäinen M, Nisula J (1996) Determining composition of grain mixtures by texture classification based on feature distributions. Int J Pattern Recognit Artif Intell 10(1):73–82, ISSN: 02180014

Okamura NK, Delwiche MJ, Thompson JF (1993) Raisin grading by machine vision. Trans Am Soc Agric Eng 36(2):485–492, ISSN: 00012351

O'Kane KC, Stanley DA, Meredith DL, Davis BE (1990) Preliminary evaluation of a computer vision sensor for analysis of phosphate tailings. Minerals and metallurgical processing, pp 137–142, ISSN: 07479182, ISBN: 087335088X

Olson JG (1987) Automated visual inspection of devices. Eval Eng 26(9):5, Between p 24 and 32, ISSN: 00143316

Omar M, Viti V, Saito K, Liu J (2006) Self-adjusting robotic painting system. Ind Robot 33(1 SPEC. ISS):50–55, ISSN: 0143991X

Onaga E (1996) Adept vision – Examples for guidance. Robot Tokyo 111:10–16, ISSN: 03871940

Optical Inspection and Metrology for Non-Optics Industries (2009) Proc SPIE Int Soc Opt Eng 7432:358 p, ISSN: 0277786X, ISBN: 9780819477224

Ordaz MA, Lush GB (2000) Machine vision for solar cell characterization. Proc SPIE Int Soc Opt Eng 3966:238–248, ISSN: 0277786X

Otieno AW, Mirman CR (2002) Machine vision applications within a manufacturing engineering technology program. ASEE annual conference proceedings, pp 1277–1285, ISSN: 01901052

Ozkul T (2009) Equipping legacy robots with vision: Performance, availability and accuracy considerations. Int J Mechatron Manuf Syst 2(3):331–347, ISSN: 17531039

Paakkari J (1998) On-line flatness measurement of large steel plates using moiré topography. VTT Publ 350: X–88, ISSN: 12350621

Padir T (2009) A portable workcell design for robotics industry. ASEE annual conference and exposition. Conference proceedings, 11 p; Hsieh S-J, Hoermann K (2009) Integrated virtual learning system for programmable logic controller (Virtual PLC): Current progress and future directions. ASEE annual conference and exposition, conference proceedings, 17 p; Chiou R, Mauk M, Agarwal S, Yang Y-T (2009) Development of E-quality laboratory modules for use in engineering quality control courses. ASEE annual conference and exposition, conference proceedings, 13 p; O'Rourke K (2009) Smart cameras get smaller and easier to network, Adv Imag 24(7):27–28, ISSN: 10420711

Paliwal J, Borhan MS, Jayas DS (2004) Classification of cereal grains using a flatbed scanner. Can Biosyst Eng/Le Genie des biosystems au Canada 46:3.1–3.5, ISSN: 14929058

Pan L, Liu R, Peng S, Chai Y, Yang SX (2007) An wireless electronic nose network for odours around livestock farms. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430745, pp 211–216, ISBN: 1424413583; 9781424413584

Pandey P, Turton R (2005) Studies to investigate variables affecting coating uniformity in a pan coating device. AIChE annual meeting. Conference proceedings, p 2899; Amavasai BP, Caparrelli F, Selvan A, Boissenin M, Travis JR, Meikle S (2005) Machine vision methods for autonomous micro-robotic systems, Kybernetes, 34 (9–10), pp 1421–1439, ISSN: 0368492X

Pandit SM, Guo R (1997) Profile recognition and mensuration in machine vision. J Manuf Sci Eng Trans ASME 119(3):417–424, ISSN: 10871357

Park B, Lawrence KC, Windham WR, Snead MP (2006) Real-time multispectral imaging application for poultry safety inspection. Proc SPIE Int Soc Opt Eng 6070, art. no. 607007, ISSN: 0277786X, ISBN: 0819461105; 9780819461100

Park B, Kise M, Lawrence KC, Windham WR, Smith DP, Thai CN (2006) Real-time multispectral imaging system for online poultry fecal inspection using UML. Proc SPIE Int Soc Opt Eng 6381, art. no. 63810W, ISSN: 0277786X, ISBN: 0819464791; 9780819464798

Park B, Yoon SC, Kise M, Lawrence KC, Windham WR (2008) Improving performance of real-time multispectral imaging system. American society of agricultural and biological engineers annual international meeting 2008, 11, pp 6877–6888, ISBN: 9781605605364

Park B, Kise M, Lawrence KC, Windham WR, Yoon SC (2008) Portable multispectral imaging instrument for food industry. American society of agricultural and biological engineers – Food processing automation conference 2008, pp 1–8, ISBN: 9781605607900

Park B, Yoon SC, Kise M, Lawrence KC, Windham WR (2009) Adaptive image processing methods for improving contaminant detection accuracy on poultry carcasses. Trans ASABE 52(3):999–1008, ISSN: 00012351

Parker JM (2001) A robust machine vision system design to facilitate the automation of surface appearance inspections. IEEE/ASME international conference on advanced intelligent mechatronics, AIM, 1, pp 87–92; Davies R, Heleno P, Correia B, Dinis J (2001) VIP3D – An application of image processing technology for quality control in the food industry. IEEE Int Conf Image Proces 1:293–296

Parker JM, Lee K-M (1996) Physically-accurate synthetic images for machine vision design. Proc Japan/USA Symp Flexible Autom 2:937–943

Parker JM, Lee K-M (1999) Physically-accurate synthetic images for machine vision design. J Manuf Sci Eng Trans ASME 121(4):763–770, ISSN: 10871357

Parker JM, Cheong YL, Gnanaprakasam P, Hou Z, Istre J (2002) Inspection technology to facilitate automated quality control of highly specular, smooth coated surfaces. Proc IEEE Int Conf Robot Autom 3: 2567–2574, ISSN: 10504729

Partee J, Paul C, Sartor M, West J, Wichowski N, McIntyre B (2008) Automated intensifier tube measuring system. Proc SPIE Int Soc Opt Eng 6956, art. no. 695608, ISSN: 0277786X, ISBN: 9780819471475

Parthasarathy S, Wolf D, Hu E, Hackwood S, Beni G (1987) Color vision for microelectronics inspection. Proc SPIE Int Soc Opt Eng 726:125–130, ISSN: 0277786X, ISBN: 0892527617

Pastorius WJ (1988) Machine vision for industrial inspection metrology and guidance, pp 13A2.1/1-13A2.1/5; Wambol JC, Sommer III HJ (1988) Machine vision for inspection of codfish, Instrumentation in the Aerospace Industry. Proceedings of the ISAAerospace Instrumentation Symposium, 34, pp 71–82, ISSN: 00967238

Pastorius WJ (1989) Vision technologies for aerospace manufacturing. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–162, ISSN: 01616382

Patterson P, Capindale D (1997) Flexible manufacturing for the assembly of small electric DC motors. Proceedings of the electrical/electronics insulation conference, pp 333–336, ISSN: 03622479

Pau L-F (1996) An intelligent camera for active vision. Int J Pattern Recognit Artif Intell 10(1):33–42, ISSN: 02180014

Payne LA, Toomajian MA (1987) Vision for inspection and robot guidance: An assessment of processing and interface capabilities required. Proceedings of the annual control engineering conference, pp 235–246, ISBN: 0914331566

Peach M (2005) Electronics sheds light on the vision sector. Electronics World 111(1826):16–19, ISSN: 13654675

Peled A, Saphier O (2007) Manufacturing large LCDs while maintaining yields. Photon Spectra 41(7): 78–80, ISSN: 07311230

Pellerin C (1995) Machine vision in experimental poultry inspection. Sensor Rev 15(4):23–24, ISSN: 02602288

Pelowski KR (1986) Three dimensional measurement with machine vision, pp 2.17–2.31, ISBN: 0872632296

Pelton D (2003) Seeing things. Canadian Packaging 56(12):S3, ISSN: 00084654

Peña-Cabrera M, Lopez-Juarez I, Rios-Cabrera R, Corona-Castuera J (2005) Machine vision approach for robotic assembly. Assembly Autom 25(3): 204–216, ISSN: 01445154

Marques P, Jose A, Briones L, Florez J (1997) Color machine vision system for process control in the ceramics industry. Proc SPIE Int Soc Opt Eng 3101:182–192, ISSN: 0277786X, ISBN: 0819425214

Peng X, Chen Y, Yu W, Zhou Z, Sun G (2008) An online defects inspection method for float glass fabrication based on machine vision. Int J Adv Manuf Technol 39(11–12):1180–1189, ISSN: 02683768

Perng D-B, Chen Y-C (2010) An advanced auto-inspection system for micro-router collapse. Machine vision and applications, pp 1–14. ISSN: 09328092

Petersen V, Jr E, John C (1986) Microcomputer vision and ranging system. Robot Eng 8(7):26–28, ISSN: 08880816

Pfeifer T, Wiegers L (2000) Reliable tool wear monitoring by optimized image and illumination control in machine vision. Measure J Int Measure Conf 28(3):209–218, ISSN: 02632241

Picón A, Bereciartua A, Gutiérrez JA, Pérez J (2006) 3D high precision tube bevel measurement using laser based rotating scanner. IEEE symposium on emerging technologies and factory automation, ETFA, art. no. 4178210, pp 1190–1197, ISBN: 1424406811; 9781424406814

Pierce J (2007) Wider view. Engineer 293(7735):36–40, ISSN: 00137758

Pinto J (2007) ABB back on the winning track. PACE Process Control Eng 60(3):6, ISSN: 13296221

Piuri V, Scotti F, Roveri M (2005) Computational intelligence in industrial quality control. IEEE international workshop on intelligent signal processing – Proceedings, art. no. 1531623, pp 4–9, ISBN: 078039030X; 9780780390300

Placer M, Murovec B (2005) Algorithm for locating solder joint positions on digital images [Algoritem lociranja zvarov na slikovnih vzorcih]. Elektrotehniski

Vestnik/Electrotech Rev 72(2–3):116–120, ISSN: 00135852

Plain-Jones C, Ludlow JE (1995) Machine vision in the electronics industry. Assembly Autom 15(2):30–35, ISSN: 01445154

Poelzleitner W (1995) Optimizing the performance of near-perfect defect detection machines. Proc SPIE Int Soc Opt Eng 2597:178–188, ISSN: 0277786X, ISBN: 0819419613

Polkowski ET (1997) Real-time machine vision for semiconductor manufacturing. Proc SPIE Int Soc Opt Eng 3028:48–52, ISSN: 0277786X, ISBN: 0819424390

Ponsa D, Benavente R, Lumbreras F, Martínez J, Roca X (2003) Quality control of safety belts by machine vision inspection for real-time production. Opt Eng 42(4):1114–1120, ISSN: 00913286

Pourreza HR, Pourreza RS, Fazeli S, Taghizadeh B (2008) Automatic detection of eggshell defects based on machine vision. J Anim Vet Adv 7(10):1200–1203, ISSN: 16805593

Powell PM (2003) Process understanding promotes successful machine vision. Photon Spectra 37(12):54–56, ISSN: 07311230

Prange JM, Peyton JA (1986) Standards development. Robot Today 8(6):23–24, ISSN: 01936913

Preußer Th (2003) Surface inspection on strip or web materials for smallest defects at high speed [Bahnwareninspektion für kleinste Fehler bei hoher Geschwindigkeit]. VDI Berichte 1806:47–56+264, ISSN: 00835560

Prieto F, Boulanger P, Lepage R, Redarce T (2002) Automated inspection system using range data. Proc IEEE Int Conf Robot Autom 3:2557–2562, ISSN: 10504729

Proceedings of Electronic Imaging Science and Technology Machine Vision Applications in Industrial Inspection XIII (2005) Proc SPIE Int Soc Opt Eng 5679:308, ISSN: 0277786X

Proceedings of SPIE – The International Society of Optical Engineering (1996) Proc SPIE Int Soc Opt Eng 2911:141, ISSN: 0277786X

Proceedings of SPIE: Intelligent Manufacturing (2004) Proc SPIE Int Soc Opt Eng 5263:195, ISSN: 0277786X

Proceedings of SPIE: Intelligent Systems in Design and Manufacturing VI (2005) Proc SPIE Int Soc Opt Eng 5999:344, ISSN: 0277786X

Proceedings of SPIE: Machine vision applications in industrial inspection XI (2003) Proc SPIE Int Soc Opt Eng 5011:320, ISSN: 0277786X

Proceedings of the 12th Annual Conference on Computers and Industrial Engineering (1990) Comput Ind Eng 19(1–4):592 p, ISSN: 03608352

Proceedings of the 1992 Artificial Neural Networks in Engineering, ANNIE'92 (1992) Intelligent engineering systems through artificial neural networks, 2, 1004 p

Proceedings of the 1992 IEEE International Conference on Robotics and Automation (1992) Proceedings – IEEE international conference on robotics and automation, 2, pp 920–1874, ISBN: 0818627204

Proceedings of the 1993 Artificial Neural Networks in Engineering, ANNIE'93 (1993) Intelligent engineering systems through artificial neural networks, 3, 920 p

Proceedings of the 1997 1st IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM'97 (1997) IEEE/ASME international conference on advanced intelligent mechatronics, AIM, 155 p

Proceedings of the 1997 Display Manufacturing Technology Conference (1997) Display manufacturing technology conference, digest of technical papers, 83 p

Proceedings of the 39th IEEE Conference on Decision and Control December 12–15, 2000, Sydney Convention and Exhibition Centre Sydney, Australia (2000) Proceedings of the IEEE conference on decision and control, 5, 1043 p, ISSN: 01912216

Proceedings of the 4th International Conference on Robot Vision and Sensory Controls (1984) 500 p, ISBN: 0903608758

Proceedings of the Applied Machine Vision '94 Conference (1994) Technical paper – Society of manufacturing engineers. MS, pp var paging, ISSN: 01616382

Proceedings of the IEEE 1990 National Aerospace and Electronics Conference – NAECON 1990 (1990) IEEE proceedings of the national aerospace and electronics conference, 3, pp 961–1378

Proceedings of the IFAC Symposium on Information Control Problems in Manufacturing Technology (1993) Control Eng Practice 1(1):121–195, ISSN: 09670661

Proceedings: 2001 8th IEEE International Conference on Emerging Technologies and Factory Automation (2001) IEEE symposium on emerging technologies and factory automation, ETFA, 2, 799 p

Productivity forum: Proximity Sensing (2007) Motion System Design 49(7):38–41, ISSN: 15371794

Prusi T, Rokka P, Tuokko R (2010) Miniaturized camera systems for microfactories. IFIP Adv Info Commun Technol 315:115–122, ISSN: 18684238, ISBN: 9783642115974

Pryor TR (1983) Applications of electro-optical measurements in the automotive and related industries, pp 193–207, ISBN: 0444865527

Pryor T, Pastorius W (1983) Applications of machine vision to parts inspection and machine control in the piece part manufacturing industries, Technical paper – Society of manufacturing engineers. MS, 16 p, ISSN: 01616382

Pryor T, Pastorius W (1983) Applications of machine vision to parts inspection and machine control in the piece part manufacturing industries, pp 3.21–3.36, ISBN: 0872631141

Purcell I (2004) A vision of the future. Process Eng London 85(11):20–22, ISSN: 03701859

Putting Reliability into Quality Checking (2003) Foundry Trade J 177(3610):12+14, ISSN: 00159042

Putting It All Together in the Northeast (2006) Assembly 49(4):46–50, ISSN: 10508171

Pydipati R, Burks TF, Lee WS (2005) Statistical and neural network classifiers for citrus disease detection using machine vision. Trans Am Soc Agric Eng 48(5):2007–2014, ISSN: 00012351

Pydipati R, Burks TF, Lee WS (2006) Identification of citrus disease using color texture features and discriminant analysis. Comput Electron Agric 52(1–2): 49–59, ISSN: 01681699

Qi S-P, Zhang H-J, Li X-J, Zhou H-L (2006) Counting steel rods online using LQV neural network in real-time images. Proceedings of IEEE ICIA 2006 – 2006 IEEE International conference on information acquisition, art. no. 4097798, pp 956–960; Guo R, Zhao W, Li G (2006) Development of a CNC micro-EDM machine, IET Conference Publications, 524, pp 705–709, ISBN: 0863416969; 9780863416965

Qiao X, Murtagh F, Crookes D, Walsh P, Basheer PAM, Long A (2002) Machine vision methods for the grading of crushed aggregate. Proc SPIE Int Soc Opt Eng 4877:264–270, ISSN: 0277786X

Qin J, Burks TF, Kim DG, Bulanon DM (2008) Classification of citrus peel diseases using color texture feature analysis. American society of agricultural and biological engineers – Food processing automation conference 2008, pp 170–178, ISBN: 9781605607900

Qin Y, Wang L, Xie L, Huang Y (2008) Vision guided automatic measuring in coordinate metrology. Proc SPIE Int Soc Opt Eng 7130, art. no. 71300E, ISSN: 0277786X

Quin F Jr, Steele PH, Shmulsky R (1998) Locating knots in wood with an infrared detector system. Forest Prod J 48(10):80–84, ISSN: 00157473

Raafat H, Taboun S (1996) An integrated robotic and machine vision system for surface flaw detection and classification. Comput Ind Eng 30(1):27–40, ISSN: 03608352

Raafat HM, Taboun SM, Hill FJ (1989) Automated inspection of finished surfaces. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–206, ISSN: 01616382

Rajurkar KP, Roste MC (1986) Data compression technique for vision systems, pp 4.109–4.120, ISBN: 0872632296

Ramirez CA (1984) Robotic processes geared towards the space shuttle external tank manufacturing. Technical paper – Society of manufacturing engineers, 11 p

Rao AR (1996) Future directions in industrial machine vision: A case study of semiconductor manufacturing applications. Image Vis Comput 14(1):3–19, ISSN: 02628856

Rather ED (1986) Forth takes aim at real-time applications. Electron Syst Technol Design/Comput Design 25(19):85–90, ISSN: 00104566

Rebner R (1985) Vision sensor technology, pp 4.14–4.21, ISBN: 0872631737

Redarce T, Lucas Y, Betemps M, Jutard A (1991) CAD off-line programming for industrial machine vision. J Intell Rob Syst 4(2):129–143, ISSN: 09210296

Redner AS, Hoffman BPC (1999) based systems provide improved stress analysis. Glass 76(2):57–58, ISSN: 00170984

Reid JF (1994) Knowledge-based supervision and control of bioprocess with a machine vision-based sensing system. J Biotechnol 36(1):25–34, ISSN: 01681656

Reid GT, Rixon RC (1985) Machine vision techniques for 3-d surface form measurement. Proc Int Symp Autom Technol Autom 1:297–312, ISBN: 0947719040

Reid GT, Marshall SJ, Rixon RC (1985) 3-D machine vision for automatic measurements of complex shapes. Proceedings of the international conference on automated inspection and product control, pp 129–138, ISBN: 0903608863

Reiner J (2008) Rendering for machine vision prototyping. Proc SPIE Int Soc Opt Eng 7100, art. no. 710009, ISSN: 0277786X, ISBN: 9780819473301

Reynolds MR, Campana C, Shetty D (2004) Design of machine vision systems for improving solder paste inspection. American society of mechanical engineers, manufacturing engineering division, MED, 15, art. no. IMECE2004-62133, pp 921–929; Kelley M (2004) A self-learning camera for the validation of highly variable and pseudo-random patterns. Proc SPIE Int Soc Opt Eng 5303:116–127, ISSN: 0277786X

Richter R, Kirtley KB (1993) Pad analysis system: A successful machine vision application for solder ball inspection. Proc SPIE Int Soc Opt Eng 1823:233–244, ISSN: 0277786X, ISBN: 0819410241

Robinson T (2008) The evolution towards more competitive apple orchard systems in the USA. Acta Horti 772:491–500, ISSN: 05677572, ISBN: 9789066055315

Robotic Parts Feeding (2007) Assembly 50(6):50–54, ISSN: 10508171

Röning J, Kalaoja J, Okkonen A, Kauniskangas H (1996) Development of real-time machine-vision applications [Reaaliaikaisten konenäkösovellusten kehittäminen], VTT Tiedotteita – Valtion Teknillinen Tutkimuskeskus, 1777, pp X–72, ISSN: 12350605

Rooks BW (1997) Vision arrives at manufacturing week. Sensor Rev 17(1):33–37, ISSN: 02602288

Rooks B (1998) Vision applications emerge at manufacturing week. Sensor Rev 18(2):88–91, ISSN: 02602288

Rooks B (1999) Vision again the star at manufacturing week. Ind Robot 26(2):115–120, ISSN: 0143991X

Rooks B (2002) Matrix code reading comes to the fore at manufacturing week. Sensor Rev 22(2):125–129, ISSN: 02602288

Rooks B (2003a) Robotics and assembly automation at TEAM. Assembly Autom 23(2):166–171, ISSN: 01445154

Rooks B (2003b) Vision systems feature at TEAM. Sensor Rev 23(2):123–127, ISSN: 02602288

Rooks B (2005) The vision solution to direct part mark identification. Sensor Rev 25(1):10–13, ISSN: 02602288

Rosati G, Boschetti G, Biondi A, Rossi A (2009a) On-line dimensional measurement of small components on the eyeglasses assembly line. Opt Lasers Eng 47(3–4):320–328, ISSN: 01438166

Rosati G, Boschetti G, Biondi A, Rossi A (2009b) Real-time defect detection on highly reflective curved surfaces. Opt Lasers Eng 47(3–4):379–384, ISSN: 01438166

Rosenfeld A (1985a) Machine vision for industry: Tasks, tools and techniques. Image Vis Comput 3(3):122–135, ISSN: 02628856

Rosenfeld A (1985b) Machine vision for industry: Concepts and techniques. Robot Today 7(6):19–22, ISSN: 01936913

Rughooputh HCS, Rughooputh SDDV, Kinser J (2000) Neural network based automated texture classification system. Proc SPIE Int Soc Opt Eng 3966:340–348, ISSN: 0277786X

Russo AB (1989) Implementing machine vision with a systems integrator. Manuf Syst 7(4):42–44, ISSN: 0748948X

Rutledge GJ (1985) Machine vision: Some perspectives. Mach Tool Blue Book 80(3):42–45, ISSN: 00249106

Rutledge WCy (1990) Trends in optics and opto-electronics. ISA Trans 29(1):27–30, ISSN: 00190578

Saarelainen E, Kämäräinen J-K, Kälviäinen H, Väinölä R (2001) Neural prediction of hydrogen in vacuum tank degassing. Iron Steelmaker (I and SM) 28(3):55–59, ISSN: 02758687

Sadiku MNO (1989) Artificial intelligence. IEEE Potential 8(2):35–39, ISSN: 02786648

Saeed G, Zhang YM (2007) Weld pool surface depth measurement using a calibrated camera and structured light. Measure Sci Technol 18(8), art. no. 033, pp 2570–2578, ISSN: 09570233

Salis G, Seulin R, Morel O, Meriaudeau F (2007) Machine vision system for the inspection of reflective parts in the automotive industry. Proc SPIE Int Soc Opt Eng 6503, art. no. 65030O, ISSN: 08906955

Sallade JS, Philpott ML (1997) Synthetic template methodology for CAD directed robot vision. Int J Mach Tools Manuf 37(12):1733–1744

Samad R, Arshad MR, Samad Z (2005) Design of crack inspection system software for IC package with decision-making methods: Neural network and rule-based. WSEAS Trans Syst 4(5):645–653, ISSN: 11092777

Samuel GL, Yang S-H (2005) Determination and mapping of measurement and design coordinate systems using computational geometric techniques. Int J Adv Manuf Technol 26(7–8):819–824, ISSN: 02683768

Sanathkumara KN, Cubero SN (2007) Automated soil hardness testing machine. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430747, pp 223–228, ISBN: 1424413583; 9781424413584

Sanby C, Norton-Wayne L (1995) Machine vision inspection of lace using a neural network. Proc SPIE Int Soc Opt Eng 2423:314–322, ISSN: 0277786X, ISBN: 081941770X

Sanby C, Norton-Wayne L, Harwood R (1995) The automated inspection of lace using machine vision. Mechatronics 5(2–3):215–231, ISSN: 09574158

Saphier O, Adin R, Cohen N (2006) Challenges and approaches in auto defect classification for large-size LCD-TV. Digest Tech Papers SID Int Symp 37(4):1757–1760, ISSN: 0097966X

Sari-Sarraf H (2002) Customized algorithm automates textile shrinkage measurement. Laser Focus World 38(5):193–198, ISSN: 10438092

Sarkodie-Gyan Th, Lam Ch-W, Hong D, Campbell AW (1996) Fuzzy clustering method for efficient 2-D object recognition. IEEE Int Conf Fuzzy Syst 2:1400–1406

Sarr DP (1991) Aircraft exterior scratch measurement system using machine vision. Proc SPIE Int Soc Opt Eng 1472:177–184, ISSN: 0277786X, ISBN: 0819405817

Sarr DP (1992) Scratch measurement system using machine vision: Part II. Proc SPIE Int Soc Opt Eng 1708:811–818, ISSN: 0277786X, ISBN: 0819408735

Sarr DP (1996) Measurement of surface scratches on aircraft structures. Proc SPIE Int Soc Opt Eng 2599:196–205, ISSN: 0277786X, ISBN: 081941963X

Satish Chandra DV (1993) Orientation estimation of industrial parts for automated inspection. Midwest Symposium on Circuits and Systems, 2, pp 1436–1438, ISBN: 0780317610

Sato K, Kan'no H, Ito T (1991) System for inspecting pad-printed characters using the normalized correlation of the segmented character images. IECON Proc (Ind Electr Conf) 3:1929–1932

Satyan P (1985) Microprocessor-controlled, vision-based, automated material handling system, pp 7.75–7.84, ISBN: 0872631893

Schaffer GH (1985) Integrated QA: Closing the CIM loop. Am Mach 129(4):137–160, ISSN: 00029858

Scheiber SF (1985) Factory of the future is here! Test Measure World 5(12):24–26, ISSN: 07441657

Schaefer R, Schwab J, Lauinger N (1997) New developments in optical grating technology for machine vision and industrial sensors. Proc SPIE Int Soc Opt Eng 3208:428–436, ISSN: 0277786X

Schmid V, Bader G, Lueder E (1998) Shift, rotation and scale invariant shape recognition system using an optical Hough transform. Proc SPIE Int Soc Opt Eng 3306:102–112, ISSN: 0277786X

Schmitt LA, Gruver WA, Ansari A (1986) Robot vision system based on two-dimensional object-oriented models. IEEE Trans Systems Man Cybernet SMC-16 4:582–589, ISSN: 00189472

Schmitt R, Niggemann C, Mersmann C (2008) Contour scanning of textile preforms using a light-section sensor for the automated manufacturing of fibre-reinforced plastics. Proc SPIE Int Soc Opt Eng 7003, art. no. 70031I, ISSN: 0277786X, ISBN: 9780819472014

Schoenwald JS (1985) Strategies for robotic sensing using acoustics. Ultrasonics symposium proceedings, pp 472–482, ISSN: 00905607

Schreiber RR (1983) Robot vision: An eye to the future. Robot Today 5(3):53–57, ISSN: 01936913

Schreiber RR (1985) Robots and electronics manufacturing. Manuf Eng 95(6):43–46, ISSN: 03610853

Scogin DN (1989) Integration of design, engineering and automation systems (IDEAS). Computers in engineering. Proceedings of the international computers in engineering conference and exhibit, pp 545–551

Scott SL (1995) Successful machine vision for the pharmaceutical industry. Proc SPIE Int Soc Opt Eng 2597:40–46, ISSN: 0277786X, ISBN: 0819419613

Scott C (2002) Vision guided robotics is revolutionizing automotive manufacturing competitiveness. Technical paper – Society of manufacturing, RP (RP02-257), pp 1–24

Seitz P (1995) From electronic photography to seeing chips – The future of image sensing, AT: Advanced technologies intelligent vision. Proceedings, pp 3–9

Seitzler T (1996) Color machine vision: Why its time has finally come. Adv Imag 11(4):56–X9, ISSN: 10420711

Sempere VM, Silvestre J (2003) Multimedia applications in industrial networks: Integration of image processing in Profibus. IEEE Trans Ind Electron 50(3):440–448, ISSN: 02780046

Serafim AFL (1991) Multiresolution pyramids for segmentation of natural images based on autoregressive models: Application to calf leather classification. IECON Proc Ind Electr Conf 3:1842–1847

Seulin R, Voisin Y, Fofi D, Meriaudeau F (2004) Complete machine vision solution for tube inspection in nuclear industry. Proc SPIE Int Soc Opt Eng 5303:128–136, ISSN: 0277786X

Shabestari BN, Kourous H, Luster S, Sacha J (1997) Challenges of adapting a dual-wavelength infrared imaging system as an industrial inspection tool. Proc SPIE Int Soc Opt Eng 3205:37–44, ISSN: 0277786X

Shafi A (2004) Machine vision in automotive manufacturing. Sensor Rev 24(4):337–342, ISSN: 02602288

Shah H (2007) Why vision? Motion Syst Design 49(1):32, ISSN: 15371794

Shahin MA, Symons SJ, Meng AX (2004) Seed sizing with image analysis. ASAE annual international meeting 2004, pp 4407–4415; Shankar NG, Zhong ZW, Ravi N (2004) Classification of defects on semiconductor wafers using priority rules. Diffusion and defect data. Pt A defect and diffusion forum, 230–232, pp 135–149, ISSN: 10120386

Shaikh AA, Abbasi T, Bhatti MI (2006) Real-time multimedia data acquisition protocol for industrial applications. J Appl Sci 6(14):2912–2916, ISSN: 18125654

Shamri G (2006) Automatic inspection: Current and future technologies and solutions. Gatfworld 18(1):63–65, ISSN: 10480293

Shapiro SF (1986a) Machine vision finds a niche in automated inspection. Electr Syst Technol Design/Comput Design 25(6):46, 48–50, ISSN: 00104566

Shapiro S (1986b) Getting the most out of today's technology. Electr Syst Technol Design/Comput Design 25(13):10, Between p 77 and 92, ISSN: 00104566

Shen Y, Lam EY (2006) Simultaneous photometric correction and defect detection in semiconductor manufacturing. Proc SPIE Int Soc Opt Eng 6070, art. no. 60700F, ISSN: 0277786X, ISBN: 0819461105; 9780819461100

Shiau Y-R, Jiang BC (1999) Study of a measurement algorithm and the measurement loss in machine vision metrology. J Manuf Syst 18(1):22–34, ISSN: 02786125

Shiau Y-R (2002) Repeatability/linearity study and measurement loss cost analysis of an automatic gauge. Int J Adv Manuf Technol 20(8):603–611, ISSN: 02683768

Shih C-HV, Sherkat N, Thomas P (1996) Inexact algorithms for correction of errors due to flexibility of dynamic structures. Proceedings of the Mediterranean electrotechnical conference – MELECON, 1, pp 242–245

Shih FY (2006) General sweep mathematical morphology. Adv Imag Electr Phys 140:265–306, ISSN: 10765670

Shin D (2003) Development of PCB aligning system for exposure machine using machine vision. Proceedings of the international conference on imaging science, systems and technology, CISST, 2, pp 569–575; Dudziak R, Pautzke F (2003) Integration of a colour machine vision system for the identification of automobile tyres into a production line for cars. International Conference on Mechatronics, ICOM 2003, pp 353–358, ISBN: 1860584209

Shin DK, To HM, Ko SL (2008) Development of on-line wrinkle measurement system using machine vision. Trans Korean Soc Mech Eng A 32(3):274–279, ISSN: 12264873

Shome D, Ray PK, Mahanty B (2009) Quality improvement of machine vision-based non-contact inspection of surface roughness in turning through adaptive neuro-fuzzy interference system. Int J Prod Qual Manag 4(3):324–344, ISSN: 17466474

Shu Y, Chung R, Tan Z, Cheng J, Lam EY, Fung KSM, Wang F (2006) A novel design of grating projection system for 3D reconstruction of wafer bumps. Proc SPIE Int Soc Opt Eng 6056, art. no. 605601, ISSN: 0277786X, ISBN: 0819460966; 9780819460967

Siddaiah M, Lieberman MA, Prasad NR, Hughs SE (2004) Automation in cotton ginning. Int J Intell Syst 19(1–2):111–129, ISSN: 08848173

Siddiqi MH, Ahmad I, Sulaiman SB (2009) Edge link detector based weed classifier. Proceedings – 2009 international conference on digital image processing, ICDIP 2009, art. no. 5190618, pp 255–259, ISBN: 9780769535654

Sieger's Impressive Performance (2005) Textile Magazine 46(10):54, ISSN: 00405078

Silvestre Blanes J, Sempere Payá VM (2001) Machine vision in profibus networks. IEEE Symp Emerg Technol Factory Autom ETFA 1:367–375

Silvestre J, Almeida L, Marau R, Pedreiras P (2007) Dynamic QoS management for multimedia realtime transmission in industrial environments. IEEE symposium on emerging technologies and factory automation, ETFA, art. no. 4416963, pp 1473–1480, ISBN: 1424408261; 9781424408269

Sim D (1984) New perspective on plant vision. Control Instrument 17(6):69–71, ISSN: 00108022

Sim SK, Chua PSK, Tay ML, Gao Y (2006) Recognition of features of parts subjected to motion using ARTMAP incorporated in a flexible vibratory bowl feeder system. Artificial intelligence for engineering design. Anal Manuf AIEDAM 20(1):19–34, ISSN: 08900604

Jr Simmons JP (1986) Real-time three-dimensional vision system for robotic guidance. Robot Eng 8(1):23–25, ISSN: 08880816

Sin CL, Eng TC, Amiruddin MS, Lee TS (2010) Epoxy die attach challenges in miniature and compact DFN/QFN packages. 10th electronics packaging technology conference 2010, EPTC 2008, art. no. 4763479, pp 475–480, ISBN: 9781424421183

Singer V, Haase MA (1989) Vision system tackles glass inspection. Chilton Instrument Control Syst 62(2):3, ISSN: 01640089

Singh R, Chapman DP, Atkinson KB (1997) Machine vision for dimensional measurements in mines. Geotech Geol Eng 15(3):185–203, ISSN: 09603182

Sink D (1996) Machine vision setting new standards in remote surveillance. Defense Security Electr 28(3):26–27, ISSN: 02783479

Sippel M (2005) A closer look at vision-based gaging. Quality 44(11):20–21, ISSN: 03609936

Sirakoulis GCh, Karafyllidis I, Thanailakis A (2005) A cellular automaton for the propagation of circular fronts and its applications. Eng Appl Artif Intell 18(6):731–744, ISSN: 09521976

Skaggs FL (1984) Machine vision system speeds PCB inspection and increases measurement efficiency. Test Measure World 4(6):7, between p 94 and 111, ISSN: 07441657

Slaughter M (2008) Lighting manufacturer integrates vision into new machine. Adv Imag 23(1):32–33, ISSN: 10420711

Smallridge B (2004) Stand-alone board processes images cost-effectively. Photon Spectra 38(3):134–135, ISSN: 07311230

Smith M, Smith L (2003) Machine vision inspection for polished stone manufacture. Key Eng Mater 250:131–137, ISSN: 10139826

Smith LN, Smith ML (2005) Automatic machine vision calibration using statistical and neural network methods. Image Vis Comput 23(10):887–899, ISSN: 02628856

Smith ML (1999) Analysis of surface texture using photometric stereo acquisition and gradient space domain mapping. Image Vis Comput 17(14):1009–1019, ISSN: 02628856

Smith P, O'Doherty P, Luna C, McCarthy S (2004) Optical forensics for tracing counterfeit recorded media. Proc SPIE Int Soc Opt Eng 5616, art. no. 05, pp 40–46, ISSN: 0277786X

Smouse E (1985) Machine vision use is increasing in electronics. Robot World 3(4):20–22, ISSN: 07377908

Smyth B (1995) Self-learning machine vision. Ind Comput 14(6):12–15, ISSN: 10450203

Snyder J (1995) New machine vision technology focuses on performance, ease of use, lower costs. Chilton I&CS 68(12):25–28, ISSN: 07462395

Snyder CA, Cox JF (1989) Developing computer integrated manufacturing: Major issues and problem areas. Eng Costs Prod Econ 17(1–4):197–204, ISSN: 0167188X

Soini AJ (1994) Machine vision 1992–1996: Technology program to promote research and its utilization in industry. Proc SPIE Int Soc Opt Eng 2353:43–51, ISSN: 0277786X, ISBN: 0819416886

Soini AJ (1996) Machine vision 1992–1996, technology program in Finland. Int J Pattern Recognit Artif Intell 10(1):3–13, ISSN: 02180014

Sokolov SM, Treskunov AS (1992) Automatic vision system for final test of liquid crystal displays. Proc IEEE Int Conf Robot Autom 2:1578–1582, ISBN: 0818627204

Solinsky JC (1986a) Clever' imaging for the modern factory. Photon Spectra 20(9):6, Between p 61 and 70, ISSN: 07311230

Solinsky JC (1986b) Encoded lighting for 3-d robotic vision. Photon Spectra 20(12):57–59, ISSN: 07311230

Solinsky JC (1986c) Use of expert systems in machine vision recognition, pp 4.139–4.158, ISBN: 0872632296

Solvang B, Sziebig G, Korondi P (2008) Multilevel control of flexible manufacturing systems. Conference on human system interaction, HSI 2008, art. no. 4581541, pp 785–790, ISBN: 1424415438; 9781424415434

Some Perspectives on CAD/CAM in Mechanical Engineering (1984) American Society of Mechanical Engineers, Pressure Vessels and Piping Division (Publication) PVP, 87, 136 p, ISSN: 0277027X

Song F, Jutamulia S (2000a) Wavelength transform and its use in edge detection. Proc SPIE Int Soc Opt Eng 4221:xxii–xxxiii, ISSN: 0277786X

Song F, Jutamulia S (2000b) Wavelet transform and its use in edge detection. Proc SPIE Int Soc Opt Eng 4220:9–19, ISSN: 0277786X

Sorebo JH, Lorenz RD (2005a) Web inspection using gradient-indexed optics. IEEE Trans Ind Appl 41(6):1476–1482, ISSN: 00939994

Sorebo JH, Lorenz RD (2005b) Web inspection using gradient-indexed optics. IEEE conference record of annual pulp and paper industry technical conference, pp 295–299, ISSN: 01902172

Sorensen S, Stringham R (1999) Vision guided flexible feeding made easy. Ind Robot 26(2):99–104, ISSN: 0143991X

Spaven GP (1994) Automatic control of fibrous matter using computer vision and self-tuning knowledge base techniques. IEE Colloquium (Digest) 81:5/1–5/4, ISSN: 09633308

Spaven GP, Self AW, Pearce DF (1992) Fibertron II – An automatic wood fibre measurement and control system. IEE Conf Publ 359:65–70, ISSN: 05379989

Spelt PF, Knee HE, Glover CW (1991) Hybrid artificial intelligence architecture for diagnosis and decision-making in manufacturing. J Intel Manuf 2(5):261–268, ISSN: 09565515

Spitz L (1986) SMA Promotes automation and boosts machine vision. Electr Packaging Prod 26(1):82–86, ISSN: 00134945

Sprovieri J (2003) Doom or boom? Assembly 46(9):58–59, ISSN: 10508171

Srikanteswara S, Lu Q, King W, Drayer T, Conners R, Kline E, Araman P (1997) Real-time implementation of a color sorting system. Proc SPIE Int Soc Opt Eng 3205:170–179, ISSN: 0277786X

Stapleton T (1984) Applications of electro-optics in the factory. LIA (Laser Institute of America) 39:1–8, ISBN: 091203520X

Steiner D, Katz R (2007) Measurement techniques for the inspection of porosity flaws on machined surfaces. J Comput Inf Sci Eng 7(1):85–94, ISSN: 15309827

Sternberg SR (1984) Analyzing images by mathematical morphology. Manuf Eng 92(2):56–57, ISSN: 03610853

Sternberg SR (1985) Three-dimensional machine vision for bin-picking, pp 2.60–2.67, ISBN: 0872631737

Stojanovic R, Papadopoulos G, Georgoudakis M, Mitropulos P (2002) Vision system for finished fabric inspection. Proc SPIE Int Soc Opt Eng 4664:97–103, ISSN: 0277786X

Stovicek D (1992) Focusing on vision. Yesterday's science fiction, today's reality. Tool Prod 57(11):49–51, 53, ISSN: 00409243

Sturgill D, Detrick L (1986) Vision system exposes flaws in glass tubing. Prod Eng 33(6):7, Between p 20 and 32, ISSN: 01461737

Styron J, Shapiro J (2009) Seeing (infra)red for improved quality control. Mach Design 81(13):64–66, ISSN: 00249114

Su Z, Tian GY, Gao C (2006) A machine vision system for on-line removal of contaminants in wool. Mechatronics 16(5):243–247, ISSN: 09574158

Sun D-W (2006) Applications of computer vision for food quality evaluation: A review on recent research progress, CHISA. 17th international congress of chemical and process engineering, 19 p, ISBN: 8086059456; 9788086059457

Sun T-H, Tseng C-C, Chen M-S (2010) Electric contacts inspection using machine vision. Image Vis Comput 28(6):890–901, ISSN: 02628856

Supernault SL (1985) PC-based image processing. Robot Age 7(3):20–22, ISSN: 01971905

Sussman D (2005) Vision systems. Control Eng 52(8 Suppl), ISSN: 00108049

Svetkoff DJ, Smith DN, Doss BL (1986) Automatic inspection of component boards using 3d and greyscale vision, pp 57–64, ISBN: 0930815165

Swonger CW (1986) Machine vision. Robot Today 8(6):14–15, ISSN: 01936913

Tan KM, Liddy T, Anvar A, Lu T-F (2008) The advancement of an autonomous underwater vehicle (AUV) technology, 3rd IEEE conference on industrial electronics and applications, ICIEA 2008, art. no. 4582535, pp 336–341, ISBN: 9781424417186

Tang K, Williams' WW, Jwo W, Gong L (1999) Performance comparison between on-line sensors and control charts in manufacturing process monitoring. IIE Trans (Inst Ind Eng) 31(12):1181–1190, ISSN: 0740817X

Tantaswadi P, Vilainatre J, Tamaree N, Viraivan P (1999) Machine vision for automated visual inspection of cotton quality in textile industries using color isodiscrimination contour. Comput Ind Eng 37(1):347–350, ISSN: 03608352

Tao Y, Walker J (2000) Imaging and pattern recognition methods for feather sex separation of broiler chicks. Trans Am Soc Agric Eng 43(2):461–467, ISSN: 00012351

Tao L, Witty P, King T (1997) Machine vision in the inspection of patterned textile webs. IEE Colloquium (Digest) 41:9/1–9/5, ISSN: 09633308

Tarbox G, Gerhardt L (1989) Design and implementation of a hierarchical automated inspection system. Proc SPIE Int Soc Opt Eng 1197:13–24, ISSN: 0277786X

Tarn T-J, Song M, Xi N, Ghosh BK (1996) Multi-sensor fusion scheme for calibration-free stereo vision in a manufacturing workcell. IEEE international conference on multisensor fusion and integration for intelligent systems, pp 416–423

Taylor BR (1986) Automatic inspection in electronics manufacturing. Proc SPIE Int Soc Opt Eng 654:157–159, ISSN: 0277786X, ISBN: 0872526890

Techtextil 2003 Hits Frankfurt (2003) Nonwovens Ind 34(5):30–39, ISSN: 01634429

Thacker N (1995) Algorithm development to give eyesight to machines. New Electr 28(2):27–28, ISSN: 00479624

The View from Wall Street (2009) Photon Spectra 43(5):45–47, ISSN: 07311230

The Wonderful World Of Color (2007) Adv Imag 22(6):10–13, ISSN: 10420711

Theisen BL (2005) The 13th annual intelligent ground vehicle competition: Intelligent ground vehicles created by intelligent teams. Proc SPIE Int Soc Opt Eng 6006, art. no. 60060H, ISSN: 0277786X

Theisen BL (2007) The 15TH annual intelligent ground vehicle competition: Intelligent ground robots created by intelligent students. Proc SPIE Int Soc Opt Eng 6764, art. no. 676407, ISSN: 0277786X, ISBN: 9780819469243

Theisen BL (2009) The 16TH annual intelligent ground vehicle competition: Intelligent students creating intelligent vehicles. Proc SPIE Int Soc Opt Eng 7252, art. no. 725205, ISSN: 0277786X

Theisen BL, Maslach D (2004) The 12TH annual intelligent ground vehicle competition: Team approaches to intelligent vehicles. Proc SPIE Int Soc Opt Eng 5608, art. no. 46, pp 25–35, ISSN: 0277786X

Theisen BL, Nguyen D (2006) The 14TH annual intelligent ground vehicle competition: Intelligent teams creating intelligent ground robots. Proc SPIE Int Soc Opt Eng 6384, art. no. 63840N, ISSN: 0277786X, ISBN: 0819464821; 9780819464828

Thilmany J (2009) Accessible vsion. Mech Eng 131(7), ISSN: 00256501

Thompson SW (1984) Electrical connectors – Automated inspection of alignment using machine vision, p. 7; Falkman GA, Murray LA, Robots and vision: Where to?, pp 2.17–2.24, ISBN: 0872631400

Thomson CR (2006) Woodchip optimization – Dynamic on-line woodchip classification. Annual meeting of the pulp and paper technical association of Canada (PAPTAC), A, pp A185–A192, ISSN: 14947722, ISBN: 1897023170; 9781897023174

To HM, Shin DK, Ko SL (2007) On-line measurement of wrinkle using machine vision. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), 4706 LNCS (Part 2), pp 931–940, ISSN: 03029743, ISBN: 9783540744757

Tobias OJ, Seara R, Soares FAP, Bermudez JCM (1995) Automatic visual inspection using the cooccurrence approach. Midwest symposium on circuits and systems, 1, pp 154–157

Tong F, Xu X-M, Tso SK, Luk BL (2007) Identification of overlapped ultrasonic NDE echoes with adaptive deconvolution. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430713, pp 48–51, ISBN: 1424413583; 9781424413584

Toosizadeh S, Peiravi A (2005) Novel fuzzy test patterns and their application in the measurement of geometric characteristics of displays. Mach Vision Appl 16(5):273–281, ISSN: 09328092

Topical Conference on Path to Innovation: New Technologies in Decorating and Assembly 2008 (2008) Topical conference on path to innovation: New technologies in decorating and assembly 2008, 93 p, ISBN: 9781615670352

Torgerson E, Paul FW (1987) Vision guided robotic fabric manipulation for apparel manufacturing, pp 1196–1202, ISBN: 0818607874

Torgerson E, Paul FW (1988) Vision-guided robotic fabric manipulation for apparel manufacturing. IEEE Control Syst Mag 8(1):14–20, ISSN: 08880611

Trebilcock B (2003) I can see clearly now. Modern Mater Handl 58(4):25–27, ISSN: 00268038

Trivedi A, Liu PC-H (1997) Machine vision processing/selection in printed circuit board manufacturing. Proceedings of the annual southeastern symposium on system theory, pp 190–194

Trombly JE (1982a) Adding machine vision to assembly lines. Mach Design 54(25):78–81, ISSN: 00249114

Trombly JE (1982) Recent applications of computer aided vision in inspection and part sorting. Technical paper – Society of manufacturing engineers. MS, 12 p, ISSN: 01616382

Trombly JE (1982) Small part sorting and inspection using machine vision, Technical paper – Society of manufacturing engineers. MS, 13 p, ISSN: 01616382

Tsai D-M (1999) Machine vision approach for detecting and inspecting circular parts. Int J Adv Manuf Technol 15(3):217–221, ISSN: 02683768

Tsai D-M, Tzeng J-I (1997) Dimensional and angular measurements using least squares and neural networks. Int J Adv Manuf Technol 13(1):56–66, ISSN: 02683768

Tsai D-M, Wu S-K (2000) Automated surface inspection using Gabor filters. Int J Adv Manuf Technol 16(7):474–482, ISSN: 02683768

Tsai D-M, Tsai H-Y (2010) Low-contrast surface inspection of mura defects in liquid crystal displays using optical flow-based motion analysis. Machine vision and applications, pp 1–21, ISSN: 09328092

Tsai D-M, Chang C-C, Chao S-M (2010) Micro-crack inspection in heterogeneously textured solar wafers using anisotropic diffusion. Image Vis Comput 28(3):491–501, ISSN: 02628856

Tsang KF, Lee WC, Lam KL, Tung HY, Xuan K (2007) An integrated ZigBee automation system: An energy saving solution. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430752, pp 252–258, ISBN: 1424413583; 9781424413584

Tsang PWM, Bradbeer RS, Yeung LF, Lam KKY (2007) Facilitating inspection of underwater environment based on temporal filtering. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430737, pp 170–173, ISBN: 1424413583; 9781424413584

Tseng H-Y, Lin C-C (2007) A simulated annealing approach for curve fitting in automated manufacturing systems. J Manuf Technol Manag 18(2):202–216, ISSN: 1741038X

Tu Da-Wei, Lin C-X (2000) Spy quantitative inspection with a machine vision light sectioning method. Meas Sci Technol 11(8):1187–1192, ISSN: 09570233

Tucker JW (1989) Inside beverage can inspection. An application from start to finish. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–189, ISSN: 01616382

Tutching J (1984) Technological, market, and applications environment for machine vision systems 1984–1987: An analysis, study, and assessment of technological transition. Applications development, markets, and related factors, pp 1.1–1.9, ISBN: 0872631400

Udoka SJ (1991) Automated data capture techniques: A prerequisite for effective integrated manufacturing systems. Comput Ind Eng 21(1–4):217–221, ISSN: 03608352

Ulaş C, Demir S, Toker O, Fidanboylu K (2007) Rotation angle estimation algorithms for textures and their real-time implementation on the FU-smartCam, ISPA 2007 – Proceedings of the 5th international symposium on image and signal processing and analysis, art. no. 4383739, pp 469–475, ISBN: 9789531841160

Ulrich M, Steger C, Baumgartner A, Ebner H (2004) Recognition of laws of objects in pictures under real time requirements [Erkennung von zusammengesetzten Objekten in Bildern unter Echtzeit-Anforderungen]. ZFV Zeitschrift fur Geodasie, Geoinformation und Landmanagement 129(3):184–194, ISSN: 16188950

Ureña R, Rodríguez F, Berenguel M (2001) A machine vision system for seeds germination quality evaluation using fuzzy logic. Comput Electr Agric 32(1):1–20, ISSN: 01681699

Usamentiaga R, Molleda J, García DF, Bulnes FG (2009) Machine vision system for flatness control feedback. 2nd international conference on machine vision, ICMV 2009, art. no. 5381094, pp 105–110, ISBN: 9780769539447

Uusitalo J, Tuokko R (2007) Setting up task-optimal illumination automatically for inspection purposes. Proc SPIE Int Soc Opt Eng 6503, art. no. 65030K, ISSN: 0277786X, ISBN: 0819466166; 9780819466167

Vachtsevanos G, Daley W, Heck B, Yezzi A, Ding Y (2000) Fusion of visible and x-ray sensing modalities for the enhancement of bone detection in poultry products. Proc SPIE Int Soc Opt Eng 4203:102–110, ISSN: 0277786X

Vamos T (1983) AI as a tool for flexible manufacturing. Proceedings of the annual advanced control conference, pp 1–16

Vanderbrug G, Wilt D, Davis J (1983) Flexible/selective robotic keyboard assembly. Assembly Eng 26(7):32–35, ISSN: 00045063

Vardangalos G, Meliones A (1999) Artificial vision systems in clothing and carpet industries. Textile Horizons (AUG), pp 15–17, ISSN: 13536184

Vartiainen J, Sadovnikov A, Kamarainen J-K, Lensu L, Kälviäinen H (2008) Detection of irregularities in regular patterns. Mach Vision Appl 19(4):249–259, ISSN: 09328092

Velasquez JD, Nof SY (2008) Integration of machine-vision inspection information for best-matching of distributed components and suppliers. Comput Ind 59(1):69–81, ISSN: 01663615

Venables M (2007) Out of sight. Engineer 293(7728):34–36, ISSN: 00137758

Vienonen P, Asikainen A, Eronen J (2002) Color grading of beech parquet blocks by using spectral data. Forest Prod J 52(2):49–52, ISSN: 00157473

Vinogradov SL, Shubin VE (1995) Image processing photosensor for robots. Proc SPIE Int Soc Opt Eng 2349, pp 251–262, ISSN: 0277786X, ISBN: 0819416843

Vira N (1995) Application of microcomputer in submicron level measurements and control of a positioning device. J Microcomput Appl 18(2):149–164, ISSN: 07457138

Vision turns 20 (2007) Adv Imag 22(9):32–35, ISSN: 10420711

Vizmanos JG, Fuentes LM, Gutierrez JA (1997) Splinter detection of half-cut peaches. Proc SPIE Int Soc Opt Eng 3208:295–299, ISSN: 0277786X

Walker RA, Frame GE, Fridge DA (1986) In-line machine vision inspection and control of thick-film printing, pp 1–10, ISBN: 0930815165

Waltz FM, Snyder MA, Batchelor BG (1989) Putting automated visual inspection systems to work on the factory floor. What's missing? Proc SPIE Int Soc Opt Eng 1197:78–86, ISSN: 0277786X

Wang S-PT (1992) In-line automated inspection streamlines IC manufacturing. Laser Focus (Littleton, MA) 28(11):155–159, ISSN: 07402511

Wang XJ, Butler C (1993) Use of a machine vision system in a flexible manufacturing cell incorporating an automated coordinate measuring machine. Proceedings of the institution of mechanical engineers, Part B. J Eng Manuf 207(B3):199–204, ISSN: 09544054

Wang H, Dong Y (2008) An improved image segmentation algorithm based on otsu method. Proc SPIE Int Soc Opt Eng 6625, art. no. 66250I, ISSN: 0277786X, ISBN: 9780819467676

Wang M-JJ, Wu W-Y, Hsu C-C (2002) Automated post bonding inspection by using machine vision techniques. Int J Prod Res 40(12):2835–2848, ISSN: 00207543

Wang N, Zhang N, Dowell FE, Pearson T (2005a) Determining vitreousness of durum wheat using transmitted and reflected images. Trans Am Soc Agric Eng 48(1):219–222, ISSN: 00012351

Wang Y, Meng R, Yu D, Xu Z (2005b) Classification and recognition system research about the glass bulb products on-line by machine vision. Yi Qi Yi Biao Xue Bao/Chinese J Sci Instrument 26(Suppl):636–637+645, ISSN: 02543087

Wang Z, Ji S, Zhang L, Huan Y, Yang G (2006) Research on automatic detecting technology of Razor's assembling quality based on the machine vision system. WSEAS Trans Comput 5(6):1319–1324, ISSN: 11092750

Wang P, Wu C, Liu D, Liu Y, Liu X (2007) Image texture analysis and detection of steel ball surface defect based on LabVIEW. Yi Qi Yi Biao Xue Bao/Chinese J Sci Instrument 28(Suppl 4):208–211, ISSN: 02543087

Wang X-Y, Zhu Z-F, Fang S-L (2007) Noncooperative detection and parameter estimation of underwater acoustic DSSS-BPSK signal. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430714, pp 52–56, ISBN: 1424413583; 9781424413584

Wang J-J, Wang Z, Mo X-T, Wang P, Liu M, Ren L, Zhu Y, Zhang F, Liu W-Y (2007a) Machine vision inspection system for film. Sensor Rev 27(3):212–216, ISSN: 02602288

Wang J-J, Li R-F, Mo Xt, Liu Wy, Liu M, Ren L, Zhu Y, Zhang F, Sang Sg (2009) Web film inspection system. Optik 120(13):630–635, ISSN: 00304026

Waterbury R (1985) Microcomputers move into manufacturing. Assembly Eng 28(9):22–26, ISSN: 00045063

Weeks AR, Gallagher A, Eriksson J (1999) Detection of oranges from a color image of an orange tree. Proc SPIE Int Soc Opt Eng 3808:346-357, ISSN: 0277786X

Wei Z, Zhang G (2005) Inspecting verticality of cylindrical workpieces via multi-vision sensors based on structured light. Opt Lasers Eng 43(10):1167–1178, ISSN: 01438166

Wei N, Flaschel E, Saalbach A, Twellmann T, Nattkemper TW (2005) Reagent-free automatic cell viability determination using neural networks based machine vision and dark-field microscopy in Saccharomyces cerevisiae. Annual international conference of the IEEE engineering in medicine and biology – Proceedings, 7 vols, art. no. 1615939, pp 6305–6308, ISSN: 05891019, ISBN: 0780387406; 9780780387409

Wei W, Xiaoyi H, Deqing W, Ru X, Haixin S (2007) Performance comparison of time synchronization algorithms for OFDM underwater communication system. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430725, pp 104–107, ISBN: 1424413583; 9781424413584

Weiß K (2007) Process-integrated fully automatic machine-vision for sorting of clay roofing tiles [Prozessintegrierte Vollautomatische Maschinelle Bildverarbeitung zum Sortieren von Dachziegeln]. ZI, Ziegelindustrie Int/Brick Tile Ind Int 5:69–71, ISSN: 03410552

Weiss SA (1994) For pharmaceutical manufacturers, photonics is a remedy for recalls. Photon Spectra 28(10):94–99, ISSN: 07311230

Wen Z, Tao Y (1998) Dual-wavelength imaging for online identification of stem-ends and calyxes. Proc SPIE Int Soc Opt Eng 3460:249–253, ISSN: 0277786X

Werth L (1985) Automated high-speed online inspection of 50-pin connectors using machine vision. Annual connectors and interconnection technology symposium proceedings, pp 366–378

Werth L (1985) Performance perspective. Prod Eng 32(8):6, Between pp 30 and 42, ISSN: 01461737

West J (1982) Machine vision in the real world of manufacturing. Electr Syst Technol Design/Comput Design 22(5):5, Betwen p 89 and 96, ISSN: 00104566

West PC (1983) Overview of machine vision in the electronics industry. Wescon conference record, 27, 5 p

West JK (1985) Visual fixturing for robotic assembly. Robot World 3(7–8):32–35, ISSN: 07377908

Whelan PF, Batchelor BG (1996) Automated packing systems – A systems engineering approach. IEEE Trans Syst Man Cybernet Part A Syst Humans 26(5):533–544, ISSN: 10834427

Whelan PF, Soille P, Drimbarean A (2001) Real-time registration of paper watermarks. Real-Time Imag 7(4):367–380, ISSN: 10072014

Whiteman JL, Negin M (1986) Automation of pre-seal inspection using machine vision, pp 37–41, ISBN: 0930815165

Whiteside BR, Spares R, Howell K, Martyn MT, Coates PD (2005a) In-process monitoring and 3d dimensional assessment of micromouldings. Annual technical conference – ANTEC, conference proceedings, 2, pp 83–87

Whiteside BR, Spares R, Howell K, Martyn MT, Coates PD (2005b) Micromoulding: Extreme process monitoring and inline product assessment, Plastics. Rubber Composites 34(9):380–386, ISSN: 14658011

Whiteside BR, Spares R, Coates PD (2006) In-process 3D assessment of micromoulding features. Proc SPIE Int Soc Opt Eng 6188, art. no. 61880Z, ISSN: 0277786X, ISBN: 0819462446; 9780819462442

Wijesoma SW, Wolfe DFH, Richards RJ (1993) Eye-to-hand coordinationfor vision-guided robot control applications. Int J Rob Res 12(1):65–78, ISSN: 02783649

Wilder J, Marsic I, Muzzio FJ, Tsai A, Weiner S, Wightman C (1995a) Machine-vision – Based evaluation of mixture percentages for powder blending processes. Proc SPIE Int Soc Opt Eng 2598:200–206, ISSN: 0277786X

Wilder J, Tsai A, Festa JM (1995b) Automatic plaque assay for the pharmaceutical industry using machine vision. Proc SPIE Int Soc Opt Eng 2597:47–58, ISSN: 0277786X, ISBN: 0819419613

Wilkie F (1986) Industry opens its eyes to machine vision. Process Eng Lond 67(1):41–43, ISSN: 03701859

Wilkinson DG (1986) Computers and manufacturing at the NEL. Comput Bull Lond 2(Part 4):3–5, ISSN: 00104531

Williams C (1995) High tech drain scan. Process and Control Engineering PACE, 48 (10), p. 28; Kukuljan ZV (1994) Advances in manufacturing technology: An impact statement for 21st century. Technical paper – society of manufacturing engineers. MS (MS94-192), pp MS94-222-1-11, ISSN: 01616382

Williams M (2002) Seeing the factory from anywhere. Quality 41(9):34–39, ISSN: 03609936

Wilson M (2002) The role of seam tracking in robotic welding and bonding. Ind Robot 29(2):132–137, ISSN: 0143991X

Winchester S (2003) Automation specialists use machine vision as a system development tool. IEE Comput Control Eng 14(3):36–39, ISSN: 09563385

Winter P, Sokhansanj S, Wood HC (1996) Machine vision methods for use in grain variety discrimination and quality analysis. Proc SPIE Int Soc Opt Eng 2907:230–240, ISSN: 0277786X, ISBN: 0819423092

Winzar J (1999) Machine vision reaches top gear. Environ Eng 12(1):21–22, ISSN: 09545824

Wollak E, King B (2009) Success through the lens. InTech 56(8), ISSN: 0192303X

Wong AKC (1991) Recent research on sensor and knowledge-based robotics. Electronic materials: Technology, here and now, pp 175–201, ISBN: 0938994581

Wong YS, Yuen WK, Lee KS, Bradley C (1998) Machine vision monitoring of tool wear. Proc SPIE Int Soc Opt Eng 3518:17–24, ISSN: 0277786X

Wood EI (1985) Successfully applied vision-based robotics. Robot World 3(6):48–50, ISSN: 07377908

Woods T (2005a) Foresight saga. Engineering 246(9):30–32, ISSN: 00137782

Woods T (2005b) Seeing it through. Engineering 246(1):31–32, ISSN: 00137782

Workshop on geometric reasoning (1988) Artif Intel 37(1–3):1–412, ISSN: 00043702

Wu Q, He T (2008) Indirectly online 3D position measurement based on machine vision using auxiliary gauge. Proc SPIE Int Soc Opt Eng 7130, art. no. 71301Q, ISSN: 0277786X

Wu W, Huang D, Wang F, Ge S (2005) Automatic recognition and classifier of online parts based on machine vision. Proc SPIE Int Soc Opt Eng 6040, art. no. 60400B, ISSN: 0277786X

Wu H-HP, Yang J-G, Hsu M-M, Wu C-J (2007a) Automatic measurement and grading of LED dies on wafer by machine vision. Proceedings of the 2007 4th IEEE international conference on mechatronics, ICM 2007, art. no. 4279980, ISBN: 142441184X; 9781424411849

Wu Y, Jiang Y-Q, Xiang Y, Li Y-Z, Shen D-J, Li D-M, Li B (2007b) Automated reading system of mercury thermometer using machine vision techniques. Bandaoti Guangdian/Semiconductor Optoelectronics 28(3):433–436, ISSN: 10015868

Wu F, Zhang X, Kuan Y, He Z (2008) An AOI algorithm for PCB based on feature extraction. Proceedings of the world congress on intelligent control and automation (WCICA), art. no. 4592931, pp 240–247, ISBN: 9781424421145

Wu H, Zhang X, Kuang Y, Lu S (2008) A real-time machine vision system for solder paste inspection. IEEE/ASME international conference on advanced intelligent mechatronics, AIM, art. no. 4601660, pp 205–210, ISBN: 9781424424955

X400 Jaguar headliners assembled JIT using vision systems (2004) Assembly Autom 24(1):104–107, ISSN: 01445154

Xia D, Kedian W (2007) Analysis of dynamic images in machine vision and its application study in motion control. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430727, pp 112–117, ISBN: 1424413583; 9781424413584

Xie F, Pearson T, Dowell FE, Zhang N (2004) Detecting vitreous wheat kernels using reflectance and transmittance image analysis. Cereal Chem 81(5):594–597, ISSN: 00090352

Xie SQ, Cheng D, Wong S, Haemmerle E (2008) Three-dimensional object recognition system for enhancing the intelligence of a KUKA robot. Int J Adv Manuf Technol 38(7–8):822–839, ISSN: 02683768

Xiong Y (2000) Neuro-vision for 3D machine vision in intelligent manufacturing system. Proc SPIE Int Soc Opt Eng 4020:38–49, ISSN: 0277786X

Xu X, Zhang X (2007) A remote acoustic monitoring system for offshore aquaculture fish cage. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430721, pp 86–90, ISBN: 1424413583; 9781424413584

Xu L, Dong J-Y, Cai C-B, Chen Z (2007) Multi-focus image fusing based on non-negative matrix factorization. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430726, pp 108–111, ISBN: 1424413583; 9781424413584

Xu E, Zhang X, Han J, Wu C (2008) HALCON application for shape-based matching. 3rd IEEE conference on industrial electronics and applications, ICIEA 2008, art. no. 4582953, pp 2431–2434, ISBN: 9781424417186

Xu X, Cheng F, Ying Y-B (2009) Application and recent development of research on near-infrared spectroscopy for meat quality evaluation. Guang Pu Xue Yu Guang Pu Fen Xi/Spectroscopy Spectr Anal 29(7):1876–1880, ISSN: 10000593

Yagiz Y, Balaban MO, Kristinsson HG, Welt BA, Marshall MR (2009) Comparison of Minolta colorimeter and machine vision system in measuring colour of irradiated Atlantic salmon. J Sci Food Agric 89(4):728–730, ISSN: 00225142

Yan J, De Sam Lazaro A (2003) Reverse engineering of sheet metal parts using machine vision. Proceedings of the ASME design engineering technical conference, 1 B, pp 1085–1095; Theisen BL, Lane GR (2003) The 11TH Annual intelligent ground vehicle competition: Team approaches to intelligent driving and machine vision. Proc SPIE Int Soc Opt Eng 5267:60–69, ISSN: 0277786X

Yang X (2009) Laser processing robot and its industrial applications. Zhongguo Jiguang/Chinese J Lasers 36(11):2780–2798, ISSN: 02587025

Yang L, Dickinson J, Wu QMJ, Lang S (2007) A fruit recognition method for automatic harvesting. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430734, pp 152–157, ISBN: 1424413583; 9781424413584

Yarberry M (2000) Higher-resolution displays create new inspection challenges. EE: Eval Eng 39(4):2, ISSN: 01490370

Yazdchi M, Yazdi M, Mahyari AG (2009) Steel surface defect detection using texture segmentation based on multifractal dimension. Proceedings – 2009 international conference on digital image processing, ICDIP 2009, art. no. 5190595, pp 346–350, ISBN: 9780769535654

Ye S, Zou D, Wang C, Zhang E (1992) Inspecting parts of car-body by visual gaging system. Proceedings of the international instrumentation symposium, pp 561–565, ISSN: 02777576, ISBN: 1556173784

Yeh C, Perng D-B (2001) Establishing a demerit count reference standard for the classification and grading of leather hides. Int J Adv Manuf Technol 18(10):731–738, ISSN: 02683768

Yeh C-P, Hammond H (2002) An industry based student project: Implementing a machine vision systems for robotic application. ASEE annual conference proceedings, pp 3843–3851, ISSN: 01901052

Yella S, Dougherty MS, Gupta NK (2006) Artificial intelligence techniques for the automatic interpretation of data from non-destructive testing. Insight: Non-Destructive Test Condition Monitor 48(1):10–20, ISSN: 13542575

Yencharis L (1996) Let the good times roll: The imaging industry at Advanced Imaging's 10th anniversary. Adv Imag 11(5):4, ISSN: 10420711

Yencharis L (1999) Machine vision: Dependable markets beyond the ups and downs of semiconductors? Adv Imag 14(1):2, ISSN: 10420711

Yi LW, Jian YZ, Hui HC (2007) Real-time infrared imaging system based on FPGA. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430723, pp 97–99, ISBN: 1424413583; 9781424413584

Yongan Z, Zhongke S (2007) Research on fault diagnosis of temperature control system in aircraft electronic equipment cabin. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430748, pp 229–233, ISBN: 1424413583; 9781424413584

Yongyue Y, Guang Z, Peng W (2009) Reading system for 2D code based on machine vision, ICEMI 2009. Proceedings of 9th international conference on electronic measurement and instruments, art. no. 5274010, pp 4462–4465, ISBN: 9781424438624

York AB (1994) Illumination systems for web scan type machine vision applications: A comparative discussion highlighting recent advances. Technical paper – Society of manufacturing engineers. MS, pp 1–13, ISSN: 01616382

Yoshimi B, Hong T-H, Herman M, Nashman M, Rippey WG (1997) Integrated vision and touch sensing for CMMs. Technical paper – Society of manufacturing engineers. AD (160), pp 1–20, ISSN: 03618765

You Y, Zhou H, Sun P, Hou S (2007) Precise position detection technique for permanent magnet linear synchronous motors. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430718, pp 73–77, ISBN: 1424413583; 9781424413584

Young E (1995a) Use of linescan cameras and a DSP processing system for high-speed wood inspection. Proc SPIE Int Soc Opt Eng 2597:259–264, ISSN: 0277786X, ISBN: 0819419613

Young RA (1995b) Bridging the gap between vision and commercial applications. Proc SPIE Int Soc Opt Eng 2411:2–14, ISSN: 0277786X, ISBN: 0819417580

Young K (2004) Automotive industry still leads the way – But where is it going? Ind Robot 31(3):244–245, ISSN: 0143991X

Young SD, Michalsky DL (1989) Robotic canopy polishing system. Technical paper – Society of manufacturing engineers. MS, pp var paging MS89–134, ISSN: 01616382

Yu W, Shi H (2006) A fast RLE-based reconstruction technique for real-time robot tracking. Wuhan Ligong Daxue Xuebao/J Wuhan Univ Technol 28(Suppl 1):582–585, ISSN: 16714431

Yu X, Zhao Z, Wang D (2008) Selection of light source for on-line nep detection system. Zhongguo Jiguang/ Chinese J Lasers 35(5):788–791, ISSN: 02587025

Yuan S, Tian D, Zeng Y (2006) 3D inspection on wafer solder bumps using binary grating projection in integrated circuit manufacturing. IEICE Trans Electr E89-C 5:602–606, ISSN: 09168524

Yue X, Han L, Yu J (2005) Application of machine vision recognition technology in piston assembly. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 36(2):88–92, ISSN: 10001298

Zang RA (1984) Applied robotics: A videotaped course, pp 667–676, ISBN: 3540150153

Zens RG Jr (2005) Guided by vision. Assembly 48(10):52–58, ISSN: 10508171

Zha Y, Liu T-G, Du D, Jiang J-F (2006) Part automatic assembling system based on machine vision. Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/J Tianjin Univ Sci Technol 39(6):722–726, ISSN: 04932137

Zhang JB, Weston RH (1994) Reference architecture for open and integrated automatic optical inspection systems. Int J Prod Res 32(7):1521–1543, ISSN: 00207543

Zhang M, Ludas LI, Morgan MT, Krutz GW, Precetti CJ (1999) Applications of color machine vision in the agricultural and food industries. Proc SPIE Int Soc Opt Eng 3543:208–219, ISSN: 0277786X

Zhang S, Liu Y, Liu W, Ran D, Wang J (2002) Non-contact measurement of revolving body straightness. Proceedings of the second international symposium on instrumentation science and technology, pp 4/132–4/136, ISBN: 7560317685

Zhang HW, Zhang GX, Shi Y (2004) Machine vision and data reconstruction for measuring free form surface. Mater Sci Forum, 471–472:508–512, ISSN: 02555476

Zhang H, Zhang G, Liu S, Qiu Z, Yu F (2006) Design of security inspection system of measuring machine for contour of prototype workpiece. Jixie Gongcheng Xuebao/Chinese J Mech Eng 42(2):212–215, ISSN: 05776686

Zhang X, Yang L (2007) Influence analysis of reflector shape with respect to the fibre optic based dynamic measurement of lubricant film for slide bearing. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430742, pp 197–201, ISBN: 1424413583; 9781424413584

Zhang X, Wang Y, Dai W (2007) An approach for fuzzy-PI control of mobile robot with wheels. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430743, pp 202–205, ISBN: 1424413583; 9781424413584

Zhang Z, He B, Dai M, Shi J (2007) A high-precision vision measurement method based on dimension characteristics of sequential partial images. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430735, pp 158–164, ISBN: 1424413583; 9781424413584

Zhang L, Liu P, Liu Y-L, Yu F-H (2009) Real-time multicore parallel image sharpness evaluation algorithm for high resolution CCD/CMOS based digital microscope autofocus imaging system. Proc SPIE Int Soc Opt Eng 7384, art. no. 738415, ISSN: 0277786X, ISBN: 9780819476654

Zhao D, Li S (2005) A 3D image processing method for manufacturing process automation. Comput Ind 56(8–9):975–985, ISSN: 01663615

Zhao C, Cheng J-T (2006) Design of image sampling and processing system on 3D measuring machine. Jilin Daxue Xuebao (Gongxueban)/J Jilin Univ (Eng Technol Edn) 36(Suppl 1):56–59, ISSN: 16715497

Zhao D, Chen JJ, Wang X, Zhang SX (1997) Region segmentation based on 3D geometry from range images. Proc SPIE Int Soc Opt Eng 3205:70–81, ISSN: 0277786X

Zhao D, Li S, Deng J (1998) 3-D processing of range image data for vision applications in manufacturing. Proc SPIE Int Soc Opt Eng 3521:27–32, ISSN: 0277786X

Zhao H, Cheng J, Jin J (2009a) NI vision based automatic optical inspection (AOI) for surface mount devices: Devices and method. International conference on applied superconductivity and electromagnetic devices, ASEMD 2009, art. no. 5306622, pp 356–360, ISBN: 9781424436873

Zhao X, Burks TF, Qin J, Ritenour MA (2009) Digital microscopic imaging for citrus peel disease classification using color texture features. Appl. Eng. Agric. 25(5):769–776, ISSN: 08838542

Zheng Z, Dong J, Chen Z (2007) Process control for the tea baking. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430707, pp 18–22, ISBN: 1424413583; 9781424413584

Zhou P, Zhao C, Zheng W, Wang J, Sun Z, Wen Y (2010) Fast method of egg image edge detecting based on dichotomy. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 26(1):238–242, ISSN: 10026819

Zhou Q, Korhonen P, Laitinen J, Sjövall S (2006) Automatic dextrous microhandling based on a 6-DOF microgripper. J Micromechatron 3(3):359–387, ISSN: 13892258

Zhu D, Beex AAL (1994) Robust spatial autoregressive modeling for hardwood log inspection. J Vis Commun Image Represent 5(1):41–51, ISSN: 10473203

Zhu D, Conners RW, Araman PA (1992) Three-dimensional image segmentation and recognition in an intelligent vision system. Proc SPIE Int Soc Opt Eng 1607:286–297, ISSN: 0277786X, ISBN: 0819407445

Zhu H, Liu W-Y, Wang J-T, Hao Y-J (2004) Application of a novel segmentation algorithm to robot vision of fruit industry. Guangdian Gongcheng/Opto-Electr Eng 31(2):54–58, ISSN: 1003501X

Zhu S, Fang J, Zhou R, Zhao J, Yu W (2008) A new noncontact flatness measuring system of large 2-D flat workpiece. IEEE Trans Instrum Measure 57(12):2891–2904, ISSN: 00189456

Zhu X, Wang Y, Xie Z, Liu C, Chu H (2009) Surface cleanliness level detection by imaging method for precision optical elements. Xinan Jiaotong Daxue Xuebao/J Southwest Jiaotong Univ 44(6):958–962, ISSN: 02582724

Zuech N (1982) Machine vision and quality control. LIA (Laser Inst Am) 33:21–23, ISBN: 0912035021

Zuech N (1984) Projections for the machine vision market. Robot World 2((7-8):43, ISSN: 07377908

Zuech N (1988) Vision moves in on manufacturing. Photon Spectra 22(4):151–152, ISSN: 07311230

Zuech N (1991) Introduction to color based machine vision. Sensors (Peterborough, NH) 8(5):37–39, ISSN: 07469462

Zuech N (2000a) Looking at machine vision as a quality-assurance data acquisition system. Adv Imag 15(9):26, 28–29, 62, ISSN: 10420711

Zuech N (2000b) When you're hot, you're hot: Machine vision and the semiconductor roller coaster. Adv Imag 15(3):2, ISSN: 10420711

Zuech N (2001) The changing character of machine vision. Sensors (Peterborough, NH), 18(6):86–91, ISSN: 07469462

Zuech N (2002) Highlights of some new applications of machine vision. Adv Imag 17(7):16–17, +31, ISSN: 10420711

Applications: Natural Objects and Scenes

Abdullah MZ, Guan LC, Mohd Azemi BMN (2001) Stepwise discriminant analysis for colour grading of oil palm using machine vision system. Food and bioproducts processing: Transactions of the institution of chemical engineers, Part C, 79(4), pp 223–231, ISSN: 09603085

Abdullah MZ, Guan LC, Mohamed AMD, Noor MAM (2002) Color vision system for ripeness inspection of oil palm elaeis guineensis. J Food Proces Preserv 26(3):213–235, ISSN: 01458892

Abdullah MZ, Guan LC, Lim KC, Karim AA (2004) The applications of computer vision system and tomographic radar imaging for assessing physical properties of food. J Food Eng 61(1 SPEC):125–135, ISSN: 02608774

Abdullah MZ, Fathinul-Syahir AS, Mohd-Azemi BMN (2005) Automated inspection system for colour and shape grading of starfruit (*Averrhoa carambola* L.) using machine vision sensor. Trans Inst Meas Control 27(2):65–87, ISSN: 01423312

Abdullah MZ, Mohamad-Saleh J, Fathinul-Syahir AS, Mohd-Azemi BMN (2006) Discrimination and classification of fresh-cut starfruits (*Averrhoa carambola* L.) using automated machine vision system. J Food Eng 76(4):506–523, ISSN: 02608774

Ahmad IS, Reid JF, Paulsen MR, Sinclair JB (1999) Color classifier for symptomatic soybean seeds using image processing. Plant Dis 83(4):320–327, ISSN: 01912917

Aleixos N, Blasco J, Navarrón F, Moltó E (2002) Multispectral inspection of citrus in real-time using machine vision and digital signal processors. Comput Electron Agric 33(2):121–137, ISSN: 01681699

American Society of Agriculture and Biological Engineers Annual International Meeting (2008), 12, 7784 p, ISBN: 9781605605364

Andres NS, Marimuthu RP, Eom Y-K, Jang B-C (2005) Development of a machine vision system for automotive part inspection. Proc SPIE Int Soc Opt Eng 6041, art. no. 60412J, ISSN: 0277786X

Annamalai P, Lee WS, Burks TF (2004) Color vision system for estimating citrus yield in real-time. ASAE annual international meeting 2004, pp 3993–4004; Kondo N, Qiao J, Shibusawa S, Sasao A, Morimoto E (2004) Mobile type sweet pepper grading robot. Proceedings of the international conference on automation technology for off-road equipment, ATOE 2004, art. no. 14D, pp 480–486, ISBN: 189276945X

Applications of artificial neural networks and genetic algorithms to agricultural systems (1997) Comput Electron Agric 18(2–3):71–224, ISSN: 01681699

Armstrong CJ, Price BL, Barrett WA (2007) Interactive segmentation of image volumes with live surface. Comput Graph, 31(2):212–229, ISSN: 00978493

ASABE – 7th world congress on computers in agriculture and natural resources 2009, WCCA 2009, 512 p, ISBN: 9781615673902

ASABE Annual International Meeting (2006) ASABE annual international meeting, 875 p; Zhang C, Cheng Y-H (2006) A quadric image segmentation for the feature extraction of tire surface wear. Proceedings – ISDA 2006: Sixth international conference on intelligent systems design and applications, 2, art. no. 4021707, pp 457–462, ISBN: 0769525288; 9780769525280

Asadollahi H, Kamarposhty MS, Teymoori MM (2009) Classification and evaluation of tomato images using several classifier. International association of computer science and information technology – Spring conference, IACSIT-SC 2009, art. no. 5169397, pp 471–474, ISBN: 9780769536538

Ashaghathra S, Weckler P, Solie J, Stone M, Wayadande A (2007) Identifying pecan weevils through image processing techniques based on template matching. ASABE annual international meeting. Technical papers, 7 book, 14 p

Åstrand B, Baerveldt A-J (2002) An agricultural mobile robot with vision-based perception for mechanical weed control. Autonom Robots 13(1):21–35, ISSN: 09295593

Aziz MZ, Shafik MS, Mertsching B, Munir A (2005) Color segmentation for visual attention of mobile robots. Proceedings – IEEE 2005 international conference on emerging technologies, ICET 2005, art. no. 1558865, pp 115–120, ISBN: 0780392477; 9780780392472

Balaban MO (2008) Quantifying nonhomogeneous colors in agricultural materials part I: Method development. J Food Sci 73(9):S431–S437, ISSN: 00221147

Balaban MO, Aparicio J, Zotarelli M, Sims C (2008) Quantifying nonhomogeneous colors in agricultural materials. Part II: Comparison of machine vision and sensory panel evaluations. J Food Sci 73(9): S438–S442, ISSN: 00221147

Baranyai L, Regen C, Geyer M, Zude M (2009) Application of Monte Carlo simulation in estimation of anisotropy of fruit tissue. 5th international technical symposium on food processing, monitoring technology in bioprocesses and food quality management, pp 1475–1478; Mathanker SK, Weckler PR, Wang N, Bowser T, Maness NO (2009) Adaptive thresholding of pecan X-ray images using water flow models and feature extraction. American society of agricultural and biological engineers annual international meeting 2009, 4, pp 2464–2472, ISBN: 9781615673629

Batchelor BG, Whelan PF (1995) Real-time colour recognition in symbolic programming for machine vision systems. Mach Vision Appl 8(6):385–398, ISSN: 09328092

Beckwith R, Greenfield S (2009) Context aware education for agriculture, 7th Annual IEEE international conference on pervasive computing and communications, PerCom 2009, art. no. 4912883, ISBN: 9781424433049

Ben-Hanan U, Peleg K, Gutman P-O (1992) Classification of fruits by a boltzmann perceptron neural network. Automatica 28(5):961–968, ISSN: 00051098

Bennedsen BS, Peterson DL (2005) Performance of a system for apple surface defect identification in near-infrared images. Biosyst Eng 90((4):419–431, ISSN: 15375110

Bennedsen BS, Peterson DL, Tabb A (2005) Identifying defects in images of rotating apples. Comput Electron Agric 48(2):92–102, ISSN: 01681699

Benson ER, Reid JF, Zhang Q (2003) Machine vision-based guidance system for an agricultural small-grain harvester. Trans Am Soc Agric Eng 46(4):1255–1264, ISSN: 00012351

Benxue M, Yibin Y (2006) Fragrant pear sexuality recognition with machine vision. Proc SPIE Int Soc Opt Eng 6381, art. no. 63811A, ISSN: 0277786X, ISBN: 0819464791; 9780819464798

Berlage AG, Cooper TM, Carone RA (1984) Seed sorting by machine vision. Agric Eng 65(10):14–17, ISSN: 00021458

Berlage AG, Cooper TM, Aristazabal JF (1988) Machine vision identification of diploid and tetraploid ryegrass seed. Trans Am Soc Agric Eng 31(1):24–27, ISSN: 00012351

Berlage AG, Churchill DB, Cooper TM, Bisland DM (1989) The application of new technologies to seed conditioning. J Agric Eng Res 42(3):193–202, ISSN: 00218634

Beving JE (1992) Application of "off-the-shelf" hardware for machine vision. Proc SPIE Int Soc Opt

Eng 1778:122–127, ISSN: 0277786X, ISBN: 0819409502

Billingsley J (2007) Vision applications in agriculture. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430754, pp 6–7, ISBN: 1424413583; 9781424413584

Billingsley J, Dunn M (2005) Unusual vision – Machine vision applications at the NCEA. Sensor Rev. 25(3):202–208, ISSN: 02602288

Blasco J, Aleixos N, Roger JM, Rabatel G, Moltó E (2002) Biosyst Eng 83(2):149–157, ISSN: 15375110

Blasco J, Aleixos N, Moltó E (2003) Machine vision system for automatic quality grading of fruit. Biosyst Eng 85(4):415–423, ISSN: 15375110

Blasco J, Aleixos N, Gómez J, Moltó E (2007a) Citrus sorting by identification of the most common defects using multispectral computer vision. J Food Eng 83(3):384–393, ISSN: 02608774

Blasco J, Aleixos N, Moltó E (2007b) Computer vision detection of peel defects in citrus by means of a region oriented segmentation algorithm. J Food Eng 81(3):535–543, ISSN: 02608774

Blasco J, Aleixos N, Cubero S, Gómez-Sanchís J, Moltó E (2009) Automatic sorting of satsuma (*Citrus unshiu*) segments using computer vision and morphological features. Comput Electr Agric 66(1):1–8, ISSN: 01681699

Blasco J, Gómez-Sanchís J, Gutierrez A, Chueca P, Argilés R, Moltó E (2009) Automatic sex detection of individuals of *Ceratitis capitata* by means of computer vision in a biofactory. Pest Manag Sci 65(1):99–104, ISSN: 1526498X

Bossu J, Gée C, Truchetet F (2007) Development of a machine vision system for a real time precision sprayer. Proc SPIE Int Soc Opt Eng 6356, art. no. 63560K, ISSN: 0277786X, ISBN: 0819464511; 9780819464514

Bradley C, Wei S, Zhang YF, Loh HT (1999) Generation of polyhedral models from machine vision data. Proc SPIE Int Soc Opt Eng 3832:26–37, ISSN: 0277786X

Brosnan T, Sun D-W (2004) Improving quality inspection of food products by computer vision – A review. J Food Eng 61(1 Spec.):3–16, ISSN: 02608774

Branson K, Robie AA, Bender J, Perona P, Dickinson MH (2009) High-throughput ethomics in large groups of Drosophila. Nat Meth 6(6):451–457, ISSN: 15487091

Bubel A, Sylvain J-F, Martin F (2009) Digital TAcy: Proof of concept. Proc SPIE Int Soc Opt Eng 7386, art. no. 73861G, ISSN: 0277786X, ISBN: 9780819476685

Bulanon DM, Kataoka T, Ota Y, Hiroma T (2002) A segmentation algorithm for the automatic recognition of Fuji apples at harvest. Biosyst Eng 83(4):405–412

Bulanon DM, Kataoka T, Ota Y, Hiroma T (2003) An algorithm for the recognition of different fruit varieties. Int Agric Eng J 12(1–2):79–94, ISSN: 08582114

Bulanon DM, Burks TF, Alchanatis V (2007) Study on fruit visibility for robotic harvesting. ASABE annual international meeting. Technical papers, 8 book, 12 p

Bulanon DM, Burks TF, Alchanatis V (2008) Analysis of the thermal temporal variation in the citrus canopy. American society of agricultural and biological engineers annual international meeting 2008, 1, pp 237–246, ISBN: 9781605605364

Bulanon DM, Burks TF, Alchanatis V (2009) Fruit visibility analysis for robotic citrus harvesting. Trans ASABE 52(1):277–283, ISSN: 00012351

Bulanon DM, Burks TF, Alchanatis V (2009) Improving fruit detection for robotic fruit harvesting. Acta Horti 824:329–336, ISSN: 05677572

Bulanon DM, Kataoka T, Okamoto H, Hata S (2004) Development of a real-time machine vision system for the apple harvesting robot. Proceedings of the SICE annual conference, art. no. FPI-12-5, pp 2531–2534

Burks TF, Shearer SA, Green JD, Heath JR (2002) Influence of weed maturity levels on species classification using machine vision. Weed Sci 50(6):802–811, ISSN: 00431745

Burks T, Villegas F, Hannan M, Flood S, Sivaraman B, Subramanian V, Sikes J (2005) Engineering and horticultural aspects of robotic fruit harvesting: Opportunities and constraints. HortTechnology 15(1):79–87, ISSN: 10630198

Buzera M, Groza V, Prostean G, Prostean O (2008) Techniques of analysing the colour of produces for automatic classification. 12th international conference on intelligent engineering systems – Proceedings, INES 2008, art. no. 4481296, pp 209–214, ISBN: 9781424420834

Cai J, Zhou X, Li Y, Fan J (2008) Recognition of mature oranges in natural scene based on machine vision. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 24(1):175–178, ISSN: 10026819

Cao Q, Wang K, Yang Y, Shi X (2009) Identifying the navigation route based on TMS320DM642 for agriculture visual robot. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 40(7):171–175, ISSN: 10001298

Cardarelli AJ, Tao Yang, Bernhardt JL, Lee FN (1999) Nondestructive quantification of internal damage in rough rice caused by insects and fungus. Proc SPIE Int Soc Opt Eng 3543:111–118, ISSN: 0277786X

Cavazzoni James, Ling PP (1999) Coupling sensing to crop models for closed-loop plant production in advanced life support systems. Proc SPIE Int Soc Opt Eng 3543:32–43, ISSN: 0277786X

Chang W-H, Chen S, Lin S-C, Huang P-Y, Chen Y-Y (1994) Vision based fruit sorting system using measures of fuzziness and degree of matching. Proceedings of the IEEE international conference on systems, man and cybernetics, 3, pp 2600–2604, ISSN: 08843627

Cheng F, Ying Y (2004a) Analyzing characteristics of hybrid rice seed. Proc SPIE Int Soc Opt Eng 5271:180–187, ISSN: 0277786X

Cheng F, Ying Y (2004b) Machine vision system for inspecting characteristics of hybrid rice seed. Proc SPIE Int Soc Opt Eng 5271:328–335, ISSN: 0277786X

Cheng F, Ying Y-B (2004c) Machine vision inspection of rice seed based on Hough transform. J Zhejiang Univ Sci 5(6):663–667, ISSN: 10093095

Cheng F, Ying YB (2004) Variety recognition of rice seeds using image analysis and artificial neural network. Proc SPIE Int Soc Opt Eng 5587, art. no. 10, pp 71–81, ISSN: 0277786X

Cheng F, Ying YB (2004) Image recognition of diseased rice seeds based on color feature. Proc SPIE Int Soc Opt Eng 5587, art. no. 26, pp 224–231, ISSN: 0277786X

Cheng F, Ying YB (2005) Image recognition of clipped stigma traces in rice seeds. Proc SPIE Int Soc Opt Eng 5996, art. no. 59960Z, ISSN: 0277786X

Cheng F, Ying YB (2005) Machine vision system for quality inspection of bulk rice seeds. Proc SPIE Int Soc Opt Eng 5996, art. no. 599616, ISSN: 0277786X

Cheng F, Ying YB (2006) Detection algorithm for multiple rice seeds images. Proc SPIE Int Soc Opt Eng 6381, art. no. 63810Z, ISSN: 0277786X, ISBN: 0819464791; 9780819464798

Cheng X, Tao Y, Chen Y-R, Luo Y (2003) NIR/MIR dual-sensor machine vision system for online apple stem-end/calyx recognition. Trans Am Soc Agric Eng 46(2):551–558, ISSN: 00012351

Cheng F, Ying YB, Liu ZY (2005) Rice seeds information system based on artificial neural network. Proc SPIE Int Soc Opt Eng 5999, art. no. 599910, ISSN: 0277786X

Cheng F, Ying YB, Li YB (2006) Detection of defects in rice seeds using machine vision. Trans ASABE 49(6):1929–1934, ISSN: 00012351

Cheng F, Zheng S, Ying Y (2007) Determine quality of rice seed using rapid techniques. Proc SPIE Int Soc Opt Eng 6761, art. no. 676115, ISSN: 0277786X, ISBN: 9780819469212

Chi Y-T, Ling PP (2004) Fast fruit identification for robotic tomato picker. ASAE Annual International Meeting 2004, p. 4159; Chong VK, Kondo N, Ninomiya K, Monta M, Namba K (2004) Comparison on eggplant fruit grading between nir-color camera and color camera. Proceedings of the international conference on automation technology for off-road equipment, ATOE 2004, art. no. 11B, pp 387–393, ISBN: 189276945X

Chi Y-T, Chien C-F, Lin T-T (2003) Leaf shape modeling and analysis using geometric descriptors derived from bezier curves. Trans Am Soc Agric Eng 46(1):175–185, ISSN: 00012351

Chien C-F, Lin T-T (2000) Non-destructive measurement of vegetable seedling leaf area using elliptical hough transform. ASAE annual intenational meeting, technical papers: Engineering solutions for a new century, 1, pp 193–209; Ying Y, Jing H, Tao Y (2000) Application of machine vision in identifying size and surface defect of huanghua pear, 2000 ASAE annual intenational meeting, technical papers: Engineering solutions for a new century, 1, pp 1443–1455; Ying Y, Jing H, Tao Y, Jin J, Ibarra J, Chen Z (2000) Application of machine vision in inspecting stem and shape of fruits. Proc SPIE Int Soc Opt Eng 4203, pp 122–130, ISSN: 0277786X

Chinchuluun R, Lee WS, Ehsani R (2007) Citrus yield mapping system on a canopy shake and catch harvester. ASABE annual international meeting. Technical papers, 7 book, 13 p,

Chinchuluun R, Lee WS, Ehsani R (2009) Machine vision system for determining citrus count and size on a canopy shake and catch harvester. Appl Eng Agric 25(4):451–458, ISSN: 08838542

Cho SI, Chang SJ, Kim YY, An KJ (2002) Development of a three-degrees-of-freedom robot for harvesting lettuce using machine vision and fuzzy logic control. Biosyst Eng 82(2):143–149, ISSN: 15375110

Chong VK, Kondo N, Ninomiya K, Monta M, Namba K (2005) Defects detection of eggplant fruit using multi spectral images. Proceedings of the SICE annual conference, pp 2381–2384

Chong VK, Kondo N, Ninomiya K, Nishi T, Monta M, Namba K, Zhang Q (2008) Features extraction for eggplant fruit grading system using machine vision. Appl Eng Agric 24(5):675–684, ISSN: 08838542

Chtioui Y, Panigrahi S, Backer LF (2003) Self-organizing map combined with a fuzzy clustering for color image segmentation of edible beans. Trans Am Soc Agric Eng 46(3):831–838, ISSN: 00012351

Churchill DB, Berlage AG, Bilsland DM, Cooper TM (1989) Decision-support system development for conditioning seeds with indent cylinder. Trans Am Soc Agric Eng 32(4):1395–1398, ISSN: 00012351

Churchill DB, Cooper DM, Bilsland DM (1991) Rotating table for measuring seed physical properties. Trans Am Soc Agric Eng 34(4):1842–1845, ISSN: 00012351

Churchill DB, Bilsland DM, Cooper TM (1992) Comparison of machine vision with human measurement of seed dimensions. Trans Am Soc Agric Eng 35(1):61–64, ISSN: 00012351

Churchill DB, Bilsland DM, Cooper TM (1993) Separation of mixed lots of tall fescue and ryegrass seed using machine vision. Trans Am Soc Agric Eng 36(5):1383–1386, ISSN: 00012351

Cooper TM, Berlage AG (1985) Machine vision for monitoring seed conditioning, Paper – American society of agricultural engineers, 8 p, ISSN: 01450166

Cooper TM, Berlage AG (1986) Integrating database and machine vision seed measuring process. Paper – American society of agricultural engineers, 11 p, ISSN: 01450166

Daley WDR, Stewart J (2009) Proactive detection of bones in poultry processing. Proc SPIE Int Soc Opt Eng 7315, art. no. 73150B, ISSN: 0277786X, ISBN: 9780819475817

Daley WD, Doll TJ, McWhorter SW, Wasilewski AA (1999) Machine vision algorithm generation using human visual models. Proc SPIE Int Soc Opt Eng 3543:65–72, ISSN: 0277786X

Davies ER (2009) The application of machine vision to food and agriculture: A review. Imag Sci J 57(4):197–217, ISSN: 13682199

Day W (2005) Engineering precision into variable biological systems. Ann Appl Biol 146(2):155–162, ISSN: 00034746

De Sena DG Jr, Pinto FDADC, De Queiroz DM, Santos NT, Khoury JK Jr (2008) Machine vision techniques and multivariate classifiers for nitrogen fertilization doses discrimination in wheat [Discriminação entre estágios nutricionais na cultura do trigo com técnicas de visão artificial e medidor portátil de clorofila]. Engenharia Agricola 28(1):187–195, ISSN: 01006916

Demuynck O, Cedeño CP, Moore AL (2009) Industrial machine vision system for fast and precise 3D object localization. Proceedings of the 9th WSEAS international conference on signal processing, computational geometry and artificial vision, ISCGAV '09, pp 160–164, ISBN: 9789604741083

D'Esnon AG (1984) Robotic harvesting of apples, asae publication, pp 112–113, ISBN: 0916150607

Diaz R, Gil L, Serrano C, Blasco M, Moltó E, Blasco J (2004) Comparison of three algorithms in the classification of table olives by means of computer vision. J Food Eng 61(1 SPEC):101–107, ISSN: 02608774

Dilawari G, Jones C (2007) Estimating quality of canola seed using a flatbed scanner. ASABE annual international meeting. Technical papers, 7 book, 10 p

Dilawari G, Jones C (2008) Estimating quality of canola seed using a flatbed scanner. American society of agricultural and biological engineers annual international meeting 2008, 6, pp 3236–3246, ISBN: 9781605605364

Ding F, Chen Y-R, Chao K, Chan DE (2006a) Two-color mixing for classifying agricultural products for safety and quality. Appl Opt 45(4):668–677, ISSN: 00036935

Ding F, Chen Y-R, Chao K, Kim MS (2006b) Three-color mixing for classifying agricultural products for safety and quality. Appl Opt 45(15):3516–3526, ISSN: 00036935

Dohi M (1996) Development of multipurpose robot for vegetable production. Jap Agric Res Quart 30(4): 227–232, ISSN: 00213551

Downey D, Crowe TG, Giles DK, Slaughter DC (2006) Direct nozzle injection of pesticide concentrate into continuous flow for intermittent spray applications. Trans ASABE 49(4):865–873, ISSN: 00012351

Du WY (2001) Motion tracking of a part on a vibratory feeder. IEEE/ASME international conference on advanced intelligent mechatronics, AIM, 1, pp 75–80; Mehl PM, Chao K, Kim M, Chen Y-R (2001) Detection of contamination on selected apple cultivars using reflectance hyperspectral and multispectral analysis. Proc SPIE Int Soc Opt Eng 4206, pp 201–213, ISSN: 0277786X

Dunn M, Billingsley J, Bell D (2006b) SR-06-119 vision based macadamia yield assessment. Sensor Rev 26(4):312–317, ISSN: 02602288

Edan Y, Miles GE, Flash T, Wolf I, Grinspun F, Peiper UM (1996) Robotic melon harvester. Service Robot 2(1):10–15, ISSN: 13563378

Eddy PR, Smith AM, Coburn C, Peddle DR, Blackshaw RE (2005) Weed detection through segmentation of ground-based hyperspectral remote sensing imagery. Proceedings of the 26th canadian symposium on remote sensing, pp 283–293; Optical Sensors and Sensing Systems for Natural Resources and Food Safety and Quality (2005) Proc SPIE Int Soc Opt Eng 5996:516, ISSN: 0277786X

Eifert JD, Sanglay GC, Lee D-J, Sumner SS, Pierson MD (2006) Prediction of raw produce surface area from weight measurement. J Food Eng 74(4):552–556, ISSN: 02608774

El-Faki MS, Zhang N, Peterson DE (2000a) Factors affecting color-based weed detection. Trans Am Soc Agric Eng 43(4):1001–1009, ISSN: 00012351

El-Faki MS, Zhang N, Peterson DE (2000b) Weed detection using color machine vision. Trans Am Soc Agric Eng 43(6):1969–1978, ISSN: 00012351

ElMasry G, Nassar A, Wang N, Vigneault C (2008) Spectral methods for measuring quality changes of fresh fruits and vegetables. Stewart Postharvest Rev 4(4), art. no. 3, ISSN: 17459656

Fennimore SA, Tourte L, Rachuy JS, Smith RF, George C (2010) Evaluation and economics of a machine-vision guided cultivation program in broccoli and lettuce. Weed Technol 24(1):33–38, ISSN: 0890037X

Fernández C, Suardíaz J, Jiménez C, Navarro PJ, Toledo A, Iborra A (2002) On-line quality control assessment of the production process of preserved orange segments. IECON Proc (Ind Electron Conf) 4:2635–2639

Forbes KA, Tattersfield GM (1999) Estimating fruit volume from digital images. IEEE AFRICON Conf 1:107–112

Fox JS, Weldon Jr EJ, Ang Jr MH (1985) Machine vision techniques for producing sugarcane seedeyes, pp 653–655, ISBN: 0818606339

Gao H, Zhu F, Cai J (2010) A review of non-destructive detection for fruit quality. IFIP Adv Info Commun Technol 317:133–140, ISSN: 18684238, ISBN: 9783642122194

Gat N, Subramanian S, Barhen J, Toomarian N (1997) Spectral imaging applications: Remote sensing, environmental monitoring, medicine, military operations, factory automation, and manufacturing. Proc SPIE Int Soc Opt Eng 2962:63–77, ISSN: 0277786X

Ghazali KH, Razali S, Mustafa MM, Hussain A (2008) Machine vision system for automatic weeding strategy in oil palm plantation using image filtering technique. 3rd international conference on information and communication technologies: From theory to applications, ICTTA, art. no. 4530075, ISBN: 1424417511; 9781424417513

Ghazanfari A, Irudayaraj J, Kusalik A (1996) Grading pistachio nuts using a neural network approach. Trans Am Soc Agric Eng 39(6):2319–2324, ISSN: 00012351

Ghazanfari A, Irudayaraj J, Kusalik A, Romaniuk M (1997) Machine vision grading of pistachio nuts using fourier descriptors. J Agric Eng Res 68(3):247–252, ISSN: 00218634

Ghazanfari A, Wulfsohn D, Irudayaraj J (1998) Machine vision grading of pistachio nuts using gray-level histogram. Can Agric Eng 40(1):61–66, ISSN: 0045432X

Giles DK, Slaughter DC (1997) Precision band spraying with machine-vision guidance and adjustable yaw nozzles. Trans Am Soc Agric Eng 40(1):29–36, ISSN: 00012351

Gómez-Sanchis J, Gómez-Chova L, Aleixos N, Camps-Valls G, Montesinos-Herrero C, Moltó E, Blasco J (2008a) Hyperspectral system for early detection of rottenness caused by *Penicillium digitatum* in mandarins. J Food Eng 89(1):80–86, ISSN: 02608774

Gómez-Sanchis J, Moltó E, Camps-Valls G, Gómez-Chova L, Aleixos N, Blasco J (2008b) Automatic correction of the effects of the light source on spherical objects. An application to the analysis of hyperspectral images of citrus fruits. J Food Eng 85(2):191–200, ISSN: 02608774

Graetzel CF, Fry SN, Nelson BJ (2006) A 6000 Hz computer vision system for real-time wing beat analysis of drosophila. Proceedings of the First IEEE/RAS-EMBS international conference on biomedical robotics and biomechatronics, BioRob 2006, art. no. 1639099, pp 278–283, ISBN: 1424400406; 9781424400409

Granitto PM, Navone HD, Verdes PF, Ceccatto HA (2002) Weed seeds identification by machine vision. Comput Electr Agric 33(2):91–103, ISSN: 01681699

Granitto PM, Verdes PF, Ceccatto HA (2005) Large-scale investigation of weed seed identification by machine vision. Comput Electron Agric 47(1):15–24, ISSN: 01681699

Gu H, Lu Y, Lou J, Zhang W (2006) Recognition and location of fruit objects based on machine vision. Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics), 4282 LNCS, pp 785–795, ISSN: 03029743, ISBN: 3540497765; 9783540497769

Gui J, Ying Y, Rao X (2004) Real time fruits size inspection based on machine vision. Proc SPIE Int Soc Opt Eng 5587, art. no. 31, pp 262–269, ISSN: 0277786X

Gui J, Ying Y, Rao X (2005) A novel fruit shape classification method based on multi-scale analysis. Proc SPIE Int Soc Opt Eng 5996, art. no. 59961F, ISSN: 0277786X

Gui J, Ying Y (2006) Fruit shape detection based on multi-scale level set framework. ASABE annual international meeting, 11 p; Vijayarekha K, Govindaraj R (2006) Citrus fruit external defect classification using wavelet packet transform features and ANN. Proceedings of the IEEE international conference on industrial technology, art. no. 4237968, pp 2872–2877, ISBN: 1424407265; 9781424407262

Gui J, Ma B, Rao X, Ying Y (2007a) Analysis and selection of the methods for fruit image denoise. Proc SPIE Int Soc Opt Eng 6761, art. no. 676117, ISSN: 0277786X, ISBN: 9780819469212

Gui J-S, Rao X-Q, Ying Y-B (2007b) Fruit shape detection by level set. J Zhejiang Univ Sci A 8(8):1232–1236, ISSN: 1673565X

Guo F, Cao Q (2004) Study on color image processing based intelligent fruit sorting system. Proceedings of the World Congress on Intelligent Control and Automation (WCICA), 6, pp 4802–4805; Ducournau S, Feutry A, Plainchault P, Revollon P, Vigouroux B, Wagner MH (2004) An image acquisition system for automated monitoring of the germination rate of sunflower seeds. Comput Electron Agric 44(3):189–202, ISSN: 01681699

Guo F, Cao Q, Xie G, Zhou J (2005) OHTA color space based method for fruit contour detection. Nongye Jixie Xuebao/Trans Chinese Soc Agric Machin 36(11):113–116, ISSN: 10001298

Guo F, Cao Q, Cui Y, Masateru N (2008a) Fruit location and stem detection method for strawberry harvesting robot. Nongye Gongcheng Xuebao/

Trans Chinese Soc Agric Eng 24(10):89–94, ISSN: 10026819

Guo F, Cao Q, Masateru N (2008b) Fruit detachment and classification method for strawberry harvesting robot. Int J Adv Rob Syst 5(1):41–48, ISSN: 17298806

Gutman P-O, Peleg K, Ben-Hanan U (1994) Classification by varying features with an erring sensor. Automatica 30(12):1943–1948, ISSN: 00051098

Guyer D, Yang X (2000) Use of genetic artificial neural networks and spectral imaging for defect detection on cherries. Comput Electron Agric 29(3):179–194, ISSN: 01681699

Hackett JK, Shah M (1989) Segmentation using intensity and range data. Opt Eng 28(6):667–674, ISSN: 00913286

Hague T, Marchant JA, Tillett ND (2000) Ground based sensing systems for autonomous agricultural vehicles. Comput Electron Agric 25(1-2):11–28, ISSN: 01681699

Hahn F, Avila A, Romanchik E (2006) Algorithm for contour detection of vegetable cactus pear cladodes prior cutting. Acta Hortic 728:297–303, ISSN: 05677572

Han Z-Z, Zhao Y-G (2009) A method of detecting peanut cultivars and quality based on the appearance characteristic recognition. 2nd international conference on information and computing science, ICIC 2009, 2, art. no. 5168997, pp 21–24, ISBN: 9780769536347

Han S, Zhang Q, Ni B, Reid JF (2004) A guidance directrix approach to vision-based vehicle guidance systems. Comput Electron Agric 43(3):179–195, ISSN: 01681699

Hannan MW, Burks TF (2004) Current developments in automated citrus harvesting. ASAE annual international meeting 2004, pp 4187–4196; Shrestha DS, Steward BL, Thorp KR, Li B (2004) A rapid video frame correspondence algorithm for agricultural video field surveying. ASAE annual international meeting 2004, pp 4027–4040; Figueiredo GF, Dawson MD, Benson ER, Van Wicklen GL, Gedamu, N (2004) Advancement in whole house machine vision based poultry behavior analysis. ASAE annual international meeting 2004, pp 4161–4172; Jafari A, Mohtasebi SS, Jahromi HE, Omid M (2004) Color feature extraction by means of discriminant analysis for weed segmentation. ASAE annual international meeting 2004, pp 4041–4052; Powell N, Boyette M (2004) Agricultural robotics using a zero turning radius platform. ASAE annual international meeting 2004, pp 4223–4228; ASAE annual international meeting (2004) ASAE annual international meeting 2004, 8347 p; Kim G, Lee K, Choi K, Son J, Choi D, Kang S (2004) Defect and ripeness inspection of citrus using NIR transmission spectrum. Key Eng Mater 270–273(II):1008–1013, ISSN: 10139826

Hannan MW, Burks TF, Bulanon DM (2007) A real-time machine vision algorithm for robotic citrus harvesting. ASABE annual international meeting. Technical papers, 8

Harrell RC, Slaughter DC, Adsit PD (1989) A fruit-tracking system for robotic harvesting. Machine Vision Appl 2(2):69–80, ISSN: 09328092

Harrell RC, Adsit PD, Munilla RD, Slaughter DC (1990) Robotic picking of citrus. Robotica 8(pt 4):269–278, ISSN: 02635747

Hayashi S, Ganno K, Ishii Y, Tanaka I (2002) Robotic harvesting system for eggplants. Jap Agric Res Quart 36(3):163–168, ISSN: 00213551

Hayashi S, Shigematsu K, Yamamoto S, Kobayashi K, Kohno Y, Kamata J, Kurita M (2010) Evaluation of a strawberry-harvesting robot in a field test. Biosyst Eng 105(2):160–171, ISSN: 15375110

He WB, Beck MS, Martin WJ (1991) Processing of living plant images for automatic selection and transfer. Comput Electron Agric 6(2):107–122, ISSN: 01681699

Heinemann PH, Pathare NP, Morrow CT (1996) An automated inspection station for machine-vision grading of potatoes. Machine Vision Appl 9(1):14–19, ISSN: 09328092

Hendrawan Y, Murase H (2009) Machine vision-based precision irrigation system for Sunagoke moss production. American society of agricultural and biological engineers annual international meeting 2009, 3, pp 1864–1875, ISBN: 9781615673629

Hočevar M, Sirok B, Jejčič V, Godeša T, Lešnik M, Stajnko D (2010) Design and testing of an automated system for targeted spraying in orchards [konzeption und erprobung eines automatisierten systems für die gezielte applikation von pflanzenschutzmitteln in obstanlagen]. J Plant Dis Protect 117(2):71–79, ISSN: 18613829

Hodgson RM (1992) Digital image processing – A developing technology for enhancing productivity. National conference publication – Institution of engineers, Australia (92 pt 15):273–278, ISSN: 03136922

Hodgson RM, McNeill SJ (1986) Design of image processing systems for real-time inspection applications. IEE Conf Publ 265:126–129, ISBN: 0852963325

Hong T, Li Z, Wu C, Liu M, Qiao J, Wang N (2007) Review of hyperspectral image technology for non-destructive inspection of fruit quality. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 23(11):280–285, ISSN: 10026819

Howarth MS, Stanwood PC (1993a) Measurement of seedling growth rate by machine vision. Proc SPIE Int Soc Opt Eng 1836:185–194, ISSN: 0277786X, ISBN: 0819410373

Howarth MS, Stanwood PC (1993b) Measurement of seedling growth rate by machine vision. Trans Am Soc Agric Eng 36(3):959–963, ISSN: 00012351

Hsiao M-L (1989) Mathematical morphology on gradient space surface tessellation. Proc SPIE Int Soc Opt Eng 1199(pt 1):203–210, ISSN: 0277786X. ISBN: 0819402389

Hsieh C-L, Weng S-F (2005) Prediction of physical properties of orchid seedlings 'Phalaenopsis Sogo Vivien F819' in a flask by digital imaging. Can Biosyst Eng/Le Genie des biosystems au Canada 47:3.23–3.32, ISSN: 14929058

Huang K-Y, Lin T-C (2000) Identifying the geometric characteristics of big plant phalaenopsis with machine vision. ASAE annual intenational meeting. Technical papers: Engineering solutions for a new century, 1, pp 211–225; Cufí X, Muñoz X, Freixenet J, Martí J (2000) A concurrent region growing algorithm guided by circumscribed contours. Proc Int Conf Pattern Recognit 15(1):432–435, ISSN: 10514651

Huang K-Y, Lin T-C, Tsai J-N (2002) Disease detection and classification in phalaenopsis seedlings using machine vision. Int Agric Eng J 11(1):11–30, ISSN: 08582114

Humburg DS, Reid JF (1991) Field performance of machine vision for the selective harvest of asparagus, SAE technical paper series, pp 1–12, ISSN: 01487191

Jain KR, Modi CK, Pithadiya KJ (2009) Non-destructive quality evaluation in spice industry with specific reference to *Cuminum cyminum* L (cumin) seeds. Innovative technologies in intelligent systems and industrial applications, CITISIA 2009, art. no. 5224191, pp 311–316, ISBN: 9781424428878

Jain KR, Modi CK, Patel JJ (2009) Non-destructive quality evaluation in spice industry with specific reference to *Cuminum cyminum* L (Cumin) seeds. 2nd international conference on emerging trends in engineering and technology, ICETET 2009, art. no. 5395509, pp 458–463, ISBN: 9780769538846

Jain KR, Modi CK, Pithadiya KJ (2009) Non-destructive quality evaluation in spice industry with specific reference to *Foeniculum vulgare* (Fennel) seeds. Proceedings of the 9th IASTED international conference on visualization, imaging, and image processing, VIIP 2009, pp 80–85, ISBN: 9780889868007

Jakeway LA, Hewetson WF (1989) Sugarcane transplanting – An alternative planting method. Appl Eng Agric 5(3):291–296, ISSN: 08838542

Jakeway LA, Fox JS, Weldon EJ, Holly DJ (1985) Machine vision and rf absorption detection techniques for robotic production of sugarcane seedpieces, ASAE Publication, pp 312–320, ISBN: 0916150674

Jayas DS (1999) An automated seed presentation device for use in machine vision identification of grain. Can Biosyst Eng/Le Genie des biosystems au Canada 41(2):113–118, ISSN: 14929058

Ji B, Zhu W, Liu B, Ma C, Li X (2009a) Review of recent machine-vision technologies in agriculture. 2nd international symposium on knowledge acquisition and modeling, KAM 2009, 3, art. no. 5362294, pp 330–334, ISBN: 9780769538884

Ji Y, Liu G, Shen W (2009b) A method based on machine vision to obtain a guidance directrix. Guangxue Xuebao/Acta Optica Sinica 29(12):3362–3366, ISSN: 02532239

Jia J, Krutz GW, Gibson HG (1991) Evaluation of machine vision algorithms for locating corn plants. SAE (Society of Automotive Engineers) Trans 100(Sect. 2):248–252, ISSN: 0096736X , ISBN: 1560912731

Jiang L, Zhu B, Cheng X, Luo Y, Tao Y (2009) 3D surface reconstruction and analysis in automated apple stem-end/calyx identification. Trans ASABE 52(5):1775–1784, ISSN: 00012351

Jiangsheng G, Yibin Y, Xiuqin R (2006) A new algorithm for fruit shape classification based on level set. Proc SPIE Int Soc Opt Eng 6384, art. no. 63840X, ISSN: 0277786X, ISBN: 0819464821; 9780819464828

Jiangsheng G, Xiuqin R, Yibin Y (2007) Fruit shape classification using support vector machine. Proc SPIE Int Soc Opt Eng 6764, art. no. 67640Z, ISSN: 0277786X, ISBN: 9780819469243

Kaewapichai W, Kaewtrakulpong P, Prateepasen A (2006) A real-time automatic inspection system for pattavia pineapples. Key Eng Mater 321–323 II:1186–1191, ISSN: 10139826

Kang S, Lee K, Lee H-Y (2006) Nondestructive cucumber quality evaluation system using machine vision. Key Eng Mater 321–323 II:1205–1208, ISSN: 10139826

Kavdir I, Guyer DE (2000) Artificial neural networks, machine vision and surface reflectance spectra for apple defect detection. ASAE annual intenational meeting, technical papers: Engineering solutions for a new century, 2, pp 937–953

Kavdir I, Guyer DE (2003) Apple grading using fuzzy logic [Bulanik mantik kullanarak elma siniflama]. Turkish J Agric Forestry 27(6):375–382, ISSN: 1300011X

Iida K, Suguri M, Umeda M, Matsui T (2000) Estimation of nitrogen content using machine vision in a paddy field. ASAE annual intenational meeting, technical papers: Engineering solutions for a new century, 1, pp 157–176

Khojastehnazhand M, Omid M, Tabatabaeefar A (2009) Determination of orange volume and surface area using image processing technique. Int Agrophys 23(3):237–242, ISSN: 02368722

Khoshroo A, Keyhani A, Rafiee S, Zoroofi RA, Zamani Z (2009) Pomegranate quality evaluation using machine vision. Acta Horti 818:347–352, ISSN: 05677572

Kim S-M, Lee Y-C, Lee S-C (2006) Vision based automatic inspection system for nuts welded on the support hinge. SICE-ICASE international joint conference, art. no. 4109209, pp 1508–1512, ISBN: 8995003855; 9788995003855

Kohno Y, Kondo N, Taniwaki S, Namba K, Nishi T, Kurita M (2009) Precision citrus production concept based on information from mobile citrus fruit grading robot, field-server, and satellite. Acta Hortic 824:237–244, ISSN: 05677572

Kondo N (2009) Robotization in fruit grading system. Sens Instrum Food Qual Saf 3(1):81–87, ISSN: 19327587

Kondo N (2010) Automation on fruit and vegetable grading system and food traceability. Trends Food Sci Technol 21(3):145–152, ISSN: 09242244

Kondo N, Nishitsuji Y, Ling PP, Ting KC (1996) Visual feedback guided robotic cherry tomato harvesting. Trans Am Soc Agric Eng 39(6):2331–2338, ISSN: 00012351

Kondo N, Ahmad U, Monta M, Murase H (2000) Machine vision based quality evaluation of Iyokan orange fruit using neural networks. Comput Electron Agric 29(1–2):135–147, ISSN: 01681699

Kondo N, Taniwaki S, Tanihara K, Yata K, Monta M, Kurita M, Tsutumi M (2007) An end-effector and manipulator control for tomato cluster harvesting robot. ASABE annual international meeting. Technical papers, 7 book, 15 p

Kondo N, Yamamoto K, Yata K, Kurita M (2008) A machine vision for tomato cluster harvesting robot. American society of agricultural and biological engineers annual international meeting 2008, 5, pp 3111–3120, ISBN: 9781605605364

Kondo N, Ling PP, Kurita M, Falzea PD, Nishizu T, Kuramoto M, Ogawa Y, Minami Y (2008) A double image acquisition system with visible and UV LEDs for citrus fruit. American society of agricultural and biological engineers – Food processing automation conference 2008, pp 159–169, ISBN: 9781605607900

Kumhala F, Kroulik M, Kviz Z, Masek J, Prosek V (2008) Sugar beets and potatoes throughput measurement by capacitive sensor. Conference: Agricultural engineering – land-technik 2008: Landtechnik regional und International, pp 199–204, ISBN: 9783180920450

Labbe JP (2003) Food security: Machine vision systems inspect fruit. Biophoton Int 10(2):20–24, ISSN: 10818693

Lamm RD, Slaughter DC, Giles DK (2002) Precision weed control system for cotton. Trans Am Soc Agric Eng 45(1):231–238, ISSN: 00012351

Laykin S, Alchanatis V, Fallik E, Edan Y (2002) Image-processing algorithms for tomato classification. Trans Am Soc Agric Eng 45(3):851–858, ISSN: 00012351

Lee D-J (2000) Color space conversion for linear color grading. Proc SPIE Int Soc Opt Eng 4197:358–366, ISSN: 0277786X

Lee D-J, Lane RM, Chang G-H (2001) Three-dimensional reconstruction for high-speed volume measurement. Proc SPIE Int Soc Opt Eng 4189:258–267, ISSN: 0277786X

Lee DJ, Eifert J, Westover B (2002) Surface area and volume measurement using radial projections. Proc SPIE Int Soc Opt Eng 4794:92–100, ISSN: 0277786X

Lee DJ, Xu X, Eifert J, Zhan P (2006) Area and volume measurements of objects with irregular shapes using multiple silhouettes. Optic Eng 45(2), art. no. 027202, ISSN: 00913286

Lee DJ, Chang Y, Archibald JK, Greco CG (2008a) Color quantization and image analysis for automated fruit quality evaluation. 4th IEEE conference on automation science and engineering, CASE 2008, art. no. 4626418, pp 194–199, ISBN: 9781424420230

Lee D-J, Archibald JK, Chang Y-C, Greco CR (2008c) Robust color space conversion and color distribution analysis techniques for date maturity evaluation. J Food Eng 88(3):364–372, ISSN: 02608774

Lee WS, Ehsani R, Shankar R (2008) Trash detection system for a citrus canopy shake and catch harvester using machine vision. American society of agricultural and biological engineers annual international meeting 2008, 7, pp 4262–4273, ISBN: 9781605605364

Lee D-J, Schoenberger R, Archibald J, McCollum S (2008e) Development of a machine vision system for automatic date grading using digital reflective near-infrared imaging. J Food Eng 86(3):388–398, ISSN: 02608774

Lee WS, Chinchuluun R, Ehsani R (2009) Citrus fruit identification using machine vision for a canopy shake and catch harvester. Acta Hortic 824:217–222, ISSN: 05677572

Leemans V, Destain M-F (2004) A real-time grading method of apples based on features extracted from defects. J Food Eng 61(1 SPEC):83–89, ISSN: 02608774

Leemans V, Destain M-F (2006) Application of the hough transform for seed row localisation using machine vision. Biosyst Eng 94(3):325–336, ISSN: 15375110

Leemans V, Destain M-F (2007) A computer-vision based precision seed drill guidance assistance. Comput Electron Agric 59(1-2):1–12, ISSN: 01681699

Leemans V, Magein H, Destain M-F (1998) Defects segmentation on 'Golden Delicious' apples by using colour machine vision. Comput Electron Agric 20(2):117–130, ISSN: 01681699

Leemans V, Magein H, Destain M-F (2002) On-line fruit grading according to their external quality using machine vision. Biosyst Eng 83(4):397–404, ISSN: 15375110

Lefcout AM, Kim MS (2006) Technique for normalizing intensity histograms of images when the approximate size of the target is known: Detection of feces on apples using fluorescence imaging. Comput Electron Agric 50(2):135–147, ISSN: 01681699

Lefcourt AM, Narayanan P, Tasch U, Rostamian R, Kim MS, Chen Y-R (2006) Development of video technology to analyze dynamics of inertia-based apple orientation. Proc SPIE Int Soc Opt Eng 6381, art. no. 63810Q, ISSN: 0277786X, ISBN: 0819464791; 9780819464798

Lefcout AM, Kim MS, Chen Y-R, Kang S (2006) Systematic approach for using hyperspectral imaging data to develop multispectral imagining systems: Detection of feces on apples. Comput Electron Agric 54(1):22–35, ISSN: 01681699

Lefcourt AM, Narayanan P, Tasch U, Rostamian R, Kim MS, Chen Y-R (2008) Algorithms for parameterization of dynamics of inertia-based apple orientation. Appl Eng Agric 24(1):123–129, ISSN: 08838542

Li Q, Lee K-M (2005) Effects of color characterization on computational efficiency of feature detection with live-object handling applications. IEEE/ASME international conference on advanced intelligent mechatronics, AIM, 1, art. no. MB4-04, pp 225–230

Li W, Lin J (2006) Seeding precision test based on machine vision. Computers in agriculture and natural resources – Proceedings of the 4th world congress, pp 375–380, ISBN: 1892769557; 9781892769558

Liming X, Yanchao Z (2010) Automated strawberry grading system based on image processing. Comput Electron Agric 71(Suppl 1):S32–S39, ISSN: 01681699

Li Q, Wang M, Gu W (2002) Computer vision based system for apple surface defect detection. Comput Electron Agric 36(2-3):215–223, ISSN: 01681699

Liu Z-Y, Cheng F, Ying Y-B, Rao X-Q (2005) Identification of rice seed varieties using neural network. J Zhejiang Univ Sci 6B(11):1095–1100, ISSN: 10093095

Li M-X, Mao H-P, Zhang Y-C (2008) Fusion algorithm for multi-sensor images based on lifting wavelet transform and fractal theory. Proceedings of the 2007 international conference on wavelet analysis and pattern recognition, ICWAPR '07, 4, art. no. 4421726, pp 1692–1695, ISBN: 1424410665; 9781424410668

Li J, Liao G, Xiao F (2008) Rapeseed seeds colour recognition by machine vision. Proceedings of the 27th Chinese control conference, CCC, art. no. 4604918, pp 146–149, ISBN: 9787900719706

Li M, Mao H, Zhang Y (2007b) Fusion algorithm for multi-sensor images based on lifting wavelet transform and fractal theory. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 38(10):91–93, +121, ISSN: 10001298

Li Z, Hong T-S, Wang W-X, Song S-R (2007) Automatic detection of growing orange fruits by machine vision, book, 11 p. ASABE annual international meeting. Technical papers, 8 book, 8 p

Li C, Cao Q, Guo F (2009a) A method for color classification of fruits based on machine vision. WSEAS Trans Syst 8(2):312–321, ISSN: 11092777

Li C-Y, Cao Q-X, Guo F (2009b) Method for fruits and vegetables classification based on dominant color histogram. Shanghai Jiaotong Daxue Xuebao/J Shanghai Jiaotong Univ 43(12):1898–1903, ISSN: 10062467

Li G, Wei X, Li F, Yan S, Liu G (2009c) Design and implementation of weighing module for fruit integrative grader. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 25(2):96–100, ISSN: 10026819

Li H, Li W, Xu X (2007d) Method for identification of wrinkled black melon seeds using photometric stereo. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 23(5):159–163, ISSN: 10026819

Lin T-T, Hsiung Y-K, Hong G-L, Chang H-K, Lu F-M (2008) Development of a virtual reality GIS using stereo vision. Comput Electr Agric 63(1):38–48, ISSN: 01681699

Linder KD, Baumgart C, Creager J, Heinen B, Troupe T, Meyer D, Carr J, Quint J (1998) Automated detection of Karnal bunt teliospores. Proc SPIE Int Soc Opt Eng 3306:80–87, ISSN: 0277786X

Ling P, Ehsani R, Ting KC, Ramalingam N, Klingman MH, Chi Y-T, Draper C (2005a) Sensing and end-effector for a robotic tomato harvester. Resour Eng Technol Sustainable World 12(7):13–14, ISSN: 10763333

Ling Y, Wang Y, Sun M, Zhang X (2005b) Application of watershed algorithm to paddy image segmentation. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 36(3):95–98, ISSN: 10001298

Lino ACL, Sanches J, Dal Fabbro IM (2008) Image processing techniques for lemons and tomatoes classification. Bragantia 67(3):785–789, ISSN: 00068705

Liu S, Wang K (2009) Detection of damaged cottonseeds using machine vision. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 40(12):186–189, ISSN: 10001298

Liu Y, Ouyang A, Wu J, Ying Y (2005) An automatic method for identifying different variety of rice seeds using machine vision technology. Proc SPIE Int Soc Opt Eng 5996, art. no. 59961H, ISSN: 0277786X

Liu H-C, Chen Z-J, Feng Y (2006) Real-time weed recognition method based on color and morphological features. Guangdian Gongcheng/Opto-Electr Eng 33(7):96–100, ISSN: 1003501X

Liu Q-M, Wang L-S, Chen X-W, Li G-F (2006) Ball nut detection by machine vision. Jilin Daxue Xuebao (Gongxueban)/J Jilin Univ (Eng Technol Edn) 36(4):534–538, ISSN: 16715497

Liu P, Tu K, Pan L, Zhan G, Jia Y, Mei W (2009a) Persimmon's surface defect recognition based on machine vision fuzzy clustering. Guangxue Xuebao/Acta Optica Sinica 29(Suppl 2):138–144, ISSN: 02532239

Liu Y, Zhang J, Richards M, Pham B, Roe P, Clarke A (2009b) Towards continuous surveillance of fruit flies using sensor networks and machine vision. Proceedings – 5th international conference on wireless communications, networking and mobile computing, WiCOM 2009, art. no. 5303034, ISBN: 9781424436934

Liu Z, Liu G, Qiao J (2010) Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 41(2):171–175, ISSN: 10001298

Lootens P, Van Waes J, Carlier L (2007) Description of the morphology of roots of *Chicorium intybus* L. partim by means of image analysis: Comparison of elliptic Fourier descriptors and classical parameters. Comput Electron Agric 58((2):164–173, ISSN: 01681699

Lu N, Tredgold A, Lu N, Tredgold A, Fielding ER (1995) Use of machine vision and fuzzy sets to classify soft fruit. Proc SPIE Int Soc Opt Eng 2620: 663–669, ISSN: 0277786X, ISBN: 0819420123

MacArthur DK, Schueller JK, Lee WS, Crane CD, MacArthur EZ, Parsons LR, (2006) Remotely-piloted helicopter citrus yield map estimation. ASABE Annual International Meeting, 12 p; Kondo N, Namba K, Nishiwaki K, Ling PP, Monta M (2006) An illumination system for machine vision inspection of agricultural products. ASABE annual international meeting, 10 p; Gui J, Ying Y, Rao X (2006) Fruit edge detection based on gradient vector flow. Proc SPIE Int Soc Opt Eng 6381, art. no. 63810O, ISSN: 0277786X, ISBN: 0819464791; 9780819464798

Mao H, Li M (2009) Tomato target matching based on multi-sensors machine vision information fusion. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 25(10):142–147, ISSN: 10026819

Mao W, Wang Y, Zhang X, Wang Y (2005) Real-time algorithm based on machine vision to segment weeds at seedling. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 36(1):83–86, ISSN: 10001298

Marchant JA, Onyango CM (2003) Comparison of a Bayesian classifier with a multilayer feed-forward neural network using the example of plant/weed/soil discrimination. Comput Electron Agric 39(1):3–22, ISSN: 01681699

Mathanker SK, Weckler PR, Taylor RK (2008) Canola-weed identification for machine vision based patch spraying. American society of agricultural and biological engineers annual international meeting 2008, 12, pp 7199–7209, ISBN: 9781605605364

Mehl PM, Chao K, Kim M, Chen YR (2002) Detection of defects on selected apple cultivars using hyperspectral and multispectral image analysis. Appl Eng Agric 18((2):219–226, ISSN: 08838542

Meyer GE, Neto JC (2008) Verification of color vegetation indices for automated crop imaging applications. Comput Electron Agric 63((2):282–293, ISSN: 01681699

Meyer GE, Hindman TW, Jones DD, Mortensen DA (2004a) Digital camera operation and fuzzy logic classification of uniform plant, soil, and residue color images. Appl Eng Agric 20(4):519–529, ISSN: 08838542

Meyer GE, Neto JC, Jones DD, Hindman TW (2004b) Intensified fuzzy clusters for classifying plant, soil, and residue regions of interest from color images. Comput Electron Agric 42(3):161–180, ISSN: 01681699

Miller WM, Drouillard GP (2001) Multiple feature analysis for machine vision grading of Florida citrus. Appl Eng Agric 17((5):627–633, ISSN: 08838542

Mishra A, Matouš K, Mishra KB, Nedbal L (2009) Towards discrimination of plant species by machine vision: Advanced statistical analysis of chlorophyll fluorescence transients. J Fluoresc 19(5):905–913, ISSN: 10530509

Moreda GP, Ortiz-Cañavate J, García-Ramos FJ, Ruiz-Altisent M (2009) Non-destructive technologies for fruit and vegetable size determination – A review. J Food Eng 92(2):119–136, ISSN: 02608774

Muñoz X, Cufí X, Freixenet J, Martí J (2000) A new approach to segmentation based on fusing circumscribed contours, region growing and clustering. IEEE Int Conf Image Proces 1:800–803

Muracciole V, Plainchault P, Bertrand D, Mannino MR (2007) development of an automated device for sorting seeds: Application on sunflower seeds, ICINCO 2007 – 4th International conference on informatics in control, automation and robotics. Proceedings, 1 RA, pp 311–318; Reese DY, Lefcourt AM, Kim MS, Lo YM (2007) Whole surface image reconstruction for machine vision inspection of fruit. Proc SPIE Int Soc Opt Eng 6761, art. no. 67610P, ISSN: 0277786X, ISBN: 9780819469212

Muracciole V, Plainchault P, Mannino M-R, Bertrand D, Vigouroux B (2007) Methodology for creating dedicated machine and algorithm on sunflower counting. Proc SPIE Int Soc Opt Eng 6761, art. no. 67610X, ISSN: 0277786X, ISBN: 9780819469212

Nagata M, Cao Q (1998) Study on grade judgment of fruit vegetables using machine vision. Jap Agric Res Quart 32(4):257–265, ISSN: 00213551

Nagata M, Cao Q, Bato PM, Shrestha BP, Kinoshita O (1997) Basic study on strawberry sorting system in Japan. Paper Am Soc Agric Eng 1:9, ISSN: 01450166

Narayanan P, Lefcourt AM, Tasen U, Rostamian R, Kim MS (2007) Tests of the ability to orient apples using their inertial properties. ASABE annual international meeting. Technical papers, 13 book, 12 p

Narayanan P, Lefcourt AM, Tasch U, Rostamian R, Grinblat A, Kim MS (2008a) Theoretical analysis of stability of axially symmetric rotating objects with regard to orienting apples. Trans ASABE 51(4):1353–1364, ISSN: 00012351

Narayanan P, Lefcourt AM, Tasch U, Rostamian R, Kim MS (2008b) Orientation of apples using their inertial properties. Trans ASABE 51(6):2073–2081, ISSN: 00012351

Neto JC, Meyer GE, Jones DD, Samal AK (2006) Plant species identification using Elliptic Fourier leaf shape analysis. Comput Electron Agric 50(2):121–134, ISSN: 01681699

Ni B, Paulsen MR, Reid JF (1998) Size grading of corn kernels with machine vision. Appl Eng Agric 14(5):567–571, ISSN: 08838542

Nickols FMJ, Le Vasan M (2007) The pedagogy of creating a mechatronic product integrated with English communication skills for teaching design and innovation to engineering undergraduates. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430751, pp 246–251, ISBN: 1424413583; 9781424413584

Öhman M, Miettinen M, Kannas K, Jutila J, Visala A, Forsman P (2008) Tree measurement and simultaneous localization and mapping system for forest harvesters. Springer Tracts Adv Robot 42:369–378, ISSN: 16107438, ISBN: 9783540754039

Okamoto T (1996) Robotization of orchid protocorm transplanting in tissue culture. Japan Agric Res Quart 30(4):213–220, ISSN: 00213551

Okamoto H, Lee WS (2009) Green citrus detection using hyperspectral imaging. Comput Electron Agric 66(2):201–208, ISSN: 01681699

Ooms D, Destain M-F (2010) Study of the potentialities of machine vision used for optical selection of immature seeds of industrial chicory (*Cichorium intybus* L.) [Étude des potentialités de la vision artificielle pour la reconnaissance optique des semences immatures de chicorée industrielle (*Cichorium intybus* L.)]. Biotechnol Agronom Soc Environ 14((1):253–263, ISSN: 13706233

Optical Inspection and Metrology for Non-Optics Industries (2009) Proc SPIE Int Soc Opt Eng 7432:358, ISSN: 0277786X, ISBN: 9780819477224

Optics in Agriculture (1991) Proc SPIE Int Soc Opt Eng 1379:255, ISSN: 0277786X

Optics in Agriculture, Forestry, and Biological Processing II (1996) Proc SPIE Int Soc Opt Eng 2907:284 p, ISSN: 0277786X, ISBN: 0819423092

Pan L, Liu R, Peng S, Chai Y, Yang SX (2007) An wireless electronic nose network for odours around livestock farms. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430745, pp 211–216, ISBN: 1424413583; 9781424413584

Park B, Lawrence KC, Windham WR, Buhr RJ (2002a) Hyperspectral imaging for detecting fecal and ingesta contaminants on poultry carcasses. Trans Am Soc Agric Eng 45((6):2017–2026, ISSN: 00012351

Park B, Lawrence KC, Windham WR, Chen Y-R, Chao K (2002b) Discriminant analysis of dual-wavelength spectral images for classifying poultry carcasses. Comput Electron Agric 33(3):219–231, ISSN: 01681699

Park B, Lawrence KC, Windham WR, Smith DP (2002c) Assessment of hyperspectral imaging system for poultry safety inspection. Proc SPIE Int Soc Opt Eng 4879:269–279, ISSN: 0277786X

Park B, Lawrence KC, Windham WR, Smith DP (2005) Detection of cecal contaminants in visceral cavity of broiler carcasses using hyperspectral imaging. Appl Eng Agric 21(4):627–635, ISSN: 08838542

Park B, Kise M, Lawrence KC, Windham WR, Yoon SC (2008) Portable multispectral imaging instrument for food industry. American society of agricultural and biological engineers – Food processing automation conference 2008, pp 1–8, ISBN: 9781605607900

Paulsen MR, Eckhoff SR, Obaldo L, Jones E, Eustace D, Ye B, Liu J (2002) Measurement and removal of garlic in wheat. Appl Eng Agric 18(3):313–324, ISSN: 08838542

Pearson T (1995) Machine vision system for automated detection of stained pistachio nuts. Proc SPIE Int Soc Opt Eng 2345:95–103, ISSN: 0277786X, ISBN: 0819416789

Pearson T (1996) Machine vision system for automated detection of stained pistachio nuts. LWT Food Sci Technol 29(3):203–209, ISSN: 00236438

Pearson TC, Slaughter DC (1996) Machine vision detection of early split pistachio nuts. Trans Am Soc Agric Eng 39(3):1203–1207, ISSN: 00012351

Pearson TC, Schatzki TF (1998) Machine vision system for automated detection of aflatoxin-contaminated pistachios. J Agric Food Chem 46(6):2248–2252, ISSN: 00218561

Pearson T, Toyofuku N (2000) Automated sorting of pistachio nuts with closed shells. Appl Eng Agric 16(1):91–94, ISSN: 08838542

Pearson TC, Doster MA, Michailides TJ (2001) Automated detection of pistachio defects by machine vision. Appl Eng Agric 17(5):729–732, ISSN: 08838542

Peirs A, Scheerlinck N, De Baerdemaeker J, Nicolaï BM (2003) Starch index determination of apple fruit by means of a hyperspectral near infrared reflectance imaging system. J Near Infrared Spectrosc 11(5):379–389, ISSN: 09670335

Peleg K, Ben-Hanan U (1994) Adaptive sorting by prototype populations. Pattern Recognit Lett 15(2):111–123, ISSN: 01678655

Pellerin C (1995) Machine vision in experimental poultry inspection. Sensor Rev 15(4):23–24, ISSN: 02602288

Penman DW (2001) Determination of stem and calyx location on apples using automatic visual inspection. Comput Electron Agric 33(1):7–18, ISSN: 01681699

Picus M, Peleg K (2000) Adaptive classification – A case study on sorting dates. J Agric Eng Res 76(4): 409–418, ISSN: 00218634

Pla F, Juste F (1995) A thinning-based algorithm to characterize fruit stems from profile images. Comput Electron Agric 13((4):301–314, ISSN: 01681699

Pla F, Sanchez JS, Sanchiz JM (1999) On-line machine vision system for fast fruit colour sorting using low-cost architecture. Proc SPIE Int Soc Opt Eng 3836:244–251, ISSN: 0277786X

Pla F, Sanchiz JM, Sánchez JS (2001) An integral automation of industrial fruit and vegetable sorting by machine vision. IEEE symposium on emerging technologies and factory automation, ETFA, 2, pp 541–546

Polder G, Van Der Heijden GWAM, Young IT (2002) Spectral image analysis for measuring ripeness of tomatoes. Trans Am Soc Agric Eng 45(4):1155–1161, ISSN: 00012351

Proceedings of 1998 Precision Agriculture and Biological Quality (1999) Proc SPIE Int Soc Opt Eng 3543, ISSN: 0277786X

Proceedings of the 1997 ASAE Annual International Meeting. Part 1 (of 3) (1997) Paper – American society of agricultural engineers, 1, pp var paging, ISSN: 01450166

Pydipati R, Burks TF, Lee WS (2005) Statistical and neural network classifiers for citrus disease detection using machine vision. Trans Am Soc Agric Eng 48(5):2007–2014, ISSN: 00012351

Pydipati R, Burks TF, Lee WS (2006) Identification of citrus disease using color texture features and discriminant analysis. Comput Electr Agric 52(1–2): 49–59, ISSN: 01681699

Qiao J, Sasao A, Shibusawa S, Kondo N (2004) Mobile fruit grading robot-concept and prototype. ASAE annual international meeting 2004, pp 4173–4185; Bulanon DM, Kataoka T, Okamoto H, Hata S (2004) Determining the 3-D location of the apple fruit during harvest. Proceedings of the international conference on automation technology for off-road equipment, ATOE 2004, art. no. 02D, pp 91–97, ISBN: 189276945X

Qin J, Burks TF, Kim DG, Bulanon DM (2008) Classification of citrus peel diseases using color texture feature analysis. American society of agricultural and biological engineers – Food processing automation conference 2008, pp 170–178, ISBN: 9781605607900

Qingbing Z, Chengliang L, Yubin M, Shengwei F, Shiping W (2008) A machine vision system for continuous field measurement of grape fruit diameter. Proceedings – 2nd international symposium on intelligent information technology application, IITA 2008, 2, art. no. 4739925, pp 1064–1068, ISBN: 9780769534978

Qiu B, Liu B, Wu C, Shi C, Li H (2005) Applications of near-infrared image processing in agricultrual engineering. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 21(5):102–106, ISSN: 10026819

Qiu D, Zhang H, Liu X, Liu Y (2007) Design of detection system for agriculture field pests based on machine vision. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 38(1):120–122, ISSN: 10001298

Rao X, Ying Y (2004) A line-scanned based digit image description method and its application in fruit quality inspection. Proc SPIE Int Soc Opt Eng 5587, art. no. 09, pp 63–70, ISSN: 0277786X

Rao X, Ying Y (2005) A method of size inspection for fruit with machine vision. Proc SPIE Int Soc Opt Eng 5996, art. no. 59961C, ISSN: 0277786X

Rao X-Q, Ying Y-B (2009) Grading a fruit by its surface color. Zhejiang Daxue Xuebao (Gongxue Ban)/J Zhejiang Univ (Eng Sci) 43((5):869–871, ISSN: 1008973X

Reese D, Lefcourt AM, Kim MS, Martin Lo Y (2009) Using parabolic mirrors for complete imaging of apple surfaces. Bioresource Technol 100(19): 4499–4506, ISSN: 09608524

Regunathan M, Lee WS (2005) Citrus fruit identification and size determination using machine vision and ultrasonic sensors. ASAE annual international meeting, 12 p; Rao X, Ying Y (2005) Color model for fruit quality inspection with machine vision. Proc SPIE Int Soc Opt Eng 5996, art. no. 59960S, ISSN: 0277786X

Rehkugler GE, Throop JA (1985) Apple sorting with machine vision. Paper Am Soc Agric Eng 32 p, ISSN: 01450166

Rehkugler GE, Throop JA (1986) Apple sorting with machine vision. Trans Am Soc Agric Eng 29(5):1388–1397, ISSN: 00012351

Rigney MP, Kranzler GA (1995) Machine vision system for measuring conifer seedling morphology. Proc SPIE Int Soc Opt Eng 2345:26–35, ISSN: 0277786X, ISBN: 0819416789

Robinson T (2008) The evolution towards more competitive apple orchard systems in the USA. Acta Horti 772:491–500, ISSN: 05677572, ISBN: 9789066055315

Safren O, Alchanatis V, Ostrovsky V, Levi O (2007) Detection of green apples in hyperspectral images of apple-tree foliage using machine vision. Trans ASABE 50(6):2303–2313, ISSN: 00012351

Sanz-Uribe JR, Ramos-Giraldo PJ, Oliveros-Tascón CE (2008) Algorithm to identify maturation stages of coffee fruits. Proceedings – Advances in electrical and electronics engineering – IAENG special edition of the world congress on engineering and computer science 2008, WCECS 2008, art. no. 5233181, pp 167–174, ISBN: 9780769535555

Savakar DG, Anami BS (2009) Recognition and classification of food grains, fruits and flowers using machine vision. Int J Food Eng 5(4), art. no. 14, ISSN: 15563758

Searcy SW, Reid JF (1989) Machines see red...and so much more. Agric Eng 70(7):10–15, ISSN: 00021458

Seitzler T (1996) Color machine vision: Why its time has finally come. Adv Imag 11(4):56–X9, ISSN: 10420711

Sena DG Jr, Pinto FAC, Queiroz DM, Viana PA (2003) Fall armyworm damaged maize plant identification using digital images. Biosyst Eng 85(4): 449–454, ISSN: 15375110

Shahin MA, Symons SJ (2003) Lentil type identification using machine vision. Can Biosyst Eng/Le Genie des biosystems au Canada 45:3.5–3.11, ISSN: 14929058

Shahin MA, Symons SJ, Meng AX (2004) Seed sizing with image analysis. ASAE annual international meeting 2004, pp 4407–4415; Ninomiya K, Kondo N, Chong VK, Monta M (2004) Machine vision systems of eggplant grading system. Proceedings of the international conference on automation technology for off-road equipment, ATOE 2004, art. no. 11D, pp 399–404, ISBN: 189276945X

Shatadal P, Jayas DS, Hehn JL, Bulley NR (1995) Seed classification using machine vision. Can Agric Eng 37(3):163–168, ISSN: 0045432X

Shi M, Zhang YF, Loh HT, Bradley C, Wong YS (2006) Triangular mesh generation employing a boundary expansion technique. Int J Adv Manuf Technol 30(1-2):54–60, ISSN: 02683768

Shi L, Wen Y, Xie X (2009) Classifier based on cloud model and its application. Proceedings – 2009 international conference on computational intelligence and software engineering, CiSE 2009, art. no. 5364494, ISBN: 9781424445073

Shiigi T, Kondo N, Chong VK, Tsuzuki S, Okada S, Nobara T (2008) Operator detection method for the construction of virtual low-noise space using spread spectrum acoustic waves. American society of agricultural and biological engineers annual international meeting 2008, 1, pp 171–181, ISBN: 9781605605364

Shiigi T, Kondo N, Kurita M, Ninomiya K, Rajendra P, Kamata J, Hayashi S, Kobayashi K, Shigematsu K, Kohno Y (2008) Strawberry harvesting robot for fruits grown on table top culture. American society of agricultural and biological engineers annual international meeting 2008, 5, pp 3139–3148, ISBN: 9781605605364

Shou Z, Wang Q, Gui J, Wang Y (2009) Fruit shape detection by optimizing Chan-Vese model. Proc SPIE Int Soc Opt Eng 7495, art. no. 74950W, ISSN: 0277786X, ISBN: 9780819478061

Shrestha DS, Steward BL (2003) Automatic corn plant population measurement using machine vision. Trans Am Soc Agric Eng 46(2):559–565, ISSN: 00012351

Shrestha DS, Steward BL (2005) Shape and size analysis of corn plant canopies for plant population and spacing sensing. Appl Eng Agric 21(2):295–303, ISSN: 08838542

Si Y, Liu G, Gao R (2009a) Segmentation algorithm for green apples recognition based on k-means algorithm. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 40(Suppl 1):100–104, ISSN: 10001298

Si Y, Qiao J, Liu G, Liu Z, Gao R (2009b) Recognition and shape features extraction of apples based on machine vision. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 40(8):161–165, +73, ISSN: 10001298

Siddiqi MH, Ahmad I, Sulaiman SB (2009) Edge link detector based weed classifier. Proceedings – 2009 international conference on digital image processing, ICDIP 2009, art. no. 5190618, pp 255–259, ISBN: 9780769535654

Simon JC, Dickinson MH (2010) A new chamber for studying the behavior of Drosophila. PLoS ONE, 5(1), art. no. e8793, pp 1–11, ISSN: 19326203

Sippel M (2008) The missing link in inspection technology. Assembly 51(5):52–55, ISSN: 10508171

Slaughter DC, Giles DK, Downey D (2008a) Autonomous robotic weed control systems: A review. Comput Electron Agric 61(1):63–78, ISSN: 01681699

Slaughter DC, Giles DK, Fennimore SA, Smith RF (2008b) Multispectral machine vision identification of lettuce and weed seedlings for automated weed control. Weed Technol 22(2):378–384, ISSN: 0890037X

Slaughter DC, Obenland DM, Thompson JF, Arpaia ML, Margosan DA (2008c) Non-destructive freeze damage detection in oranges using machine vision and ultraviolet fluorescence. Postharvest Biol Technol 48(3):341–346, ISSN: 09255214

Søgaard HT, Olsen HJ (2003) Determination of crop rows by image analysis without segmentation. Comput Electron Agric 38(2):141–158, ISSN: 01681699

Søgaard HT, Lund I (2007) Application accuracy of a machine vision-controlled robotic micro-dosing system. Biosyst Eng 96(3):315–322, ISSN: 15375110

Solis-Sánchez LO, García-Escalante JJ, Castañeda-Miranda R, Torres-Pacheco I, Guevara-González R (2009) Machine vision algorithm for whiteflies (*Bemisia tabaci* Genn.) scouting under greenhouse environment. J Appl Entomol 133(7):546–552, ISSN: 09312048

Southall B, Hague T, Marchant JA, Buxton BF (2002) An autonomous crop treatment robot: Part I. A Kalman filter model for localization and crop/weed classification. Int J Rob Res 21(1):61–74, ISSN: 02783649

Spelt PF, Knee HE, Glover CW (1991) Hybrid artificial intelligence architecture for diagnosis and decision-making in manufacturing. J Intel Manuf 2(5): 261–268, ISSN: 09565515

Staab ES, Slaughter DC, Zhang Y, Giles DK (2009) Hyperspectral imaging system for precision weed control in processing tomato. American society of agricultural and biological engineers annual international meeting 2009, 7, pp 4644–4656, ISBN: 9781615673629

Steenhoek L, Precetti C (2000) Vision sizing of seed corn. ASAE annual intenational meeting. Technical papers: Engineering solutions for a new century, 1, pp 1125–1134; Lin T-T, Chi Y-T, Liao W-C (2000) Leaf boundary extraction and geometric modeling of vegetable seedlings. ASAE annual intenational meeting, technical papers: Engineering solutions for a new century, 1, pp 177–192; Aleixos N, Blasco J, Moltó E, Navarrón F (2000) Assessment of citrus fruit quality using a real-time machine vision system. Proc Int Conf Pattern Recognit 15(1):482–485, ISSN: 10514651

Steward BL, Tian LF, Nettleton D, Tang L (2004) Reduced-dimension clustering for vegetation segmentation. Trans Am Soc Agric Eng 47(2):609–616, ISSN: 00012351

Sun D-W (2006) Applications of computer vision for food quality evaluation: A review on recent research progress, CHISA 2006 – 17th international congress of chemical and process engineering, 19 p, ISBN: 8086059456; 9788086059457

Sun W, Bradley C, Zhang YF, Loh HT (2001) Cloud data modelling employing a unified, non-redundant triangular mesh. CAD Comput Aided Design 33(2):183–193, ISSN: 00104485

Sun H, Sun M, Wang Y (2006) Status and trend of research on non-destructive measurement of plant growth based on machine vision. Nongye Jixie Xuebao/ Trans Chinese Soc Agric Mach 37(10):181–185, ISSN: 10001298

Sun M, Si J, An D, Wei Y (2007) Nondestructive measurement of tomato seedlings during their growth based on machine vision. Proceedings – second international symposium on plant growth modeling, simulation, visualization and applications, PMA 2006, art. no. 4548379, pp 255–258, ISBN: 0769528511; 9780769528519

Sun T-H, Tseng C-C, Chen M-S (2010) Electric contacts inspection using machine vision. Image Vision Comput 28(6):890–901, ISSN: 02628856

Syahrir W, Suryanti A, Connsynn C (2009) Color grading in tomato maturity estimator using image processing technique. Proceedings – 2009 2nd IEEE international conference on computer science and information technology, ICCSIT 2009, art. no. 5234497, pp 276–280, ISBN: 9781424445196

Symons SJ, Van Schepdael L, Dexter JE (2003) Measurement of hard vitreous kernels in durum wheat by machine vision. Cereal Chem 80(5):511–517, ISSN: 00090352

Tamura H, Atoda O, Sakai K (2000) Texture analysis method to distinguish clover stocks from grass thicket. Proceedings of the 4th Asia–Pacific conference on control and measurement, pp 309–314, ISBN: 7801346955

Tanaka M, Suzuki T, Seryu M, Hattori K, Kyura N (1982) Flexible handling system for small parts. Technical paper – Society of manufacturing engineers. MS, 9 p, ISSN: 01616382

Tang L, Tian LF (2008a) Plant identification in mosaicked crop row images for automatic emerged corn plant spacing measurement. Trans ASABE 51(6): 2181–2191, ISSN: 00012351

Tang L, Tian LF (2008b) Real-time crop row image reconstruction for automatic emerged corn plant spacing measurement. Trans ASABE 51(3):1079–1087, ISSN: 00012351

Tao Y (1998) Closed-loop search method for on-line automatic calibration of multi-camera inspection systems. Trans Am Soc Agric Eng 41(5):1549–1555, ISSN: 00012351

Tao Y, Wen Z (1999) An adaptive spherical image transform for high-speed fruit defect detection. Trans Am Soc Agric Eng 42(1):241–246, ISSN: 00012351

Tao Y, Heinemann PH, Varghese Z, Morrow CT, Sommer HJ III (1995) Machine vision for color inspection of potatoes and apples. Trans Am Soc Agric Eng 38(5):1555–1561, ISSN: 00012351

Tao Y, Shao J, Skeeles K, Chen YR (2000) Detection of splenomegaly in poultry carcasses by UV and color imaging. Trans Am Soc Agric Eng 43(2):469–474, ISSN: 00012351

Taouil K, Chtourou Z, Kamoun L (2008) Machine vision based quality monitoring in olive oil conditioning. 1st international workshops on image processing theory, tools and applications, IPTA 2008, art. no. 4743743, ISBN: 9781424433223

Tellaeche A, Burgos-Artizzu XP, Pajares G, Ribeiro A (2008a) 2008. Pattern Recognit 41(2):521–530, ISSN: 00313203

Tellaeche A, BurgosArtizzu XP, Pajares G, Ribeiro A, Fernández-Quintanilla C (2008b) A new vision-based approach to differential spraying in precision agriculture. Comput Electron Agric 60(2):144–155, ISSN: 01681699

Tellaeche A, Pajares G, Burgos-Artizzu XP, Ribeiro A (2011) A computer vision approach for weeds identification through Support Vector Machines. Appl Soft Comput J, ISSN: 15684946

Thompson BT (2001) Measurement of stem content for control purposes in a green leaf threshing plant. Proc SPIE Int Soc Opt Eng 4578:153–164, ISSN: 0277786X

Tian L, Slaughter DC, Norris RF (1997) Outdoor field machine vision identification of tomato seedlings for automated weed control. Trans Am Soc Agric Eng 40(6):1761–1768, ISSN: 00012351

Tillett ND, Hague T, Grundy AC, Dedousis AP (2008) Mechanical within-row weed control for transplanted crops using computer vision. Biosyst Eng 99(2):171–178, ISSN: 15375110

Timm EJ, Gilliland PV, Brown GK, Affeldt HA Jr (1991) Potential methods for detecting pits in tart cherries. Appl Eng Agric 7(1):103–109, ISSN: 08838542

Tyystjärvi E, Koski A, Keränen M, Nevalainen O (1999) The Kautsky curve is a built-in barcode. Biophys J 77(2):1159–1167, ISSN: 00063495

Unay D, Gosselin B (2005) Artificial neural network-based segmentation and apple grading by machine vision. Proceedings – International conference on image processing, ICIP, 2, art. no. 1530134, pp 630–633, ISSN: 15224880, ISBN: 0780391349; 9780780391345

Unay D, Gosselin B (2006) Automatic defect segmentation of 'Jonagold' apples on multi-spectral images: A comparative study. Postharvest Biol Technol 42(3):271–279, ISSN: 09255214

Unay D, Gosselin B (2007) Stem and calyx recognition on 'Jonagold' apples by pattern recognition. J Food Eng 78(2):597–605, ISSN: 02608774

Upchurch BL, Throop JA (1993) Considerations for implementing machine vision for detecting watercore in apples. Proc SPIE Int Soc Opt Eng 1836:291–297, ISSN: 0277786X, ISBN: 0819410373

Upchurch BL, Throop JA (1994) Effects of storage duration on detecting watercore in apples using machine vision. Trans Am Soc Agric Eng 37((2):483–486, ISSN: 00012351

Ureña R, Rodríguez F, Berenguel M (2001) A machine vision system for seeds germination quality evaluation using fuzzy logic. Comput Electr Agric 32(1):1–20, ISSN: 01681699

Vijayarekha K (2008) Multivariate image analysis for defect identification of apple fruit images. Proceedings – 34th annual conference of the IEEE industrial electronics society, IECON 2008, art. no. 4758175, pp 1499–1503, ISBN: 9781424417667

Wan Y-N (2002) Kernel handling performance of an automatic grain quality inspection system. Trans Am Soc Agric Eng 45(2):369–377, ISSN: 00012351

Wang J (1999) Intelligent measurement of the colour of food products. Proc SPIE Int Soc Opt Eng 3543:2–9, ISSN: 0277786X

Wang TY, Nguang SK (2007) Low cost sensor for volume and surface area computation of axi-symmetric agricultural products. J Food Eng 79(3):870–877, ISSN: 02608774

Wang Z, Heinemann PH, Sommer HJ III, Walker PN, Morrow CT, Heuser C (1998) Identification and separation of micropropagated sugarcane shoots based on the hough transform. Trans Am Soc Agric Eng 41(5):1535–1541, ISSN: 00012351

Wang R, Ji S, Xu Y, Li B (2001) General design for corn fertilizing intelligent machine vehicle system based on machine vision. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 17(2):151–153, ISSN: 10026819

Wang Y, Guo J, Zhao X, An A (2005) Performance detection and analysis of a machine vision based metering mechanism of drill. Nongye Jixie Xuebao/Trans Chin Soc Agric Mach 36(11):50–54+49, ISSN: 10001298

Wang N, Elmasry G, Sevakarampalayam S, Qiao J (2007) Spectral imaging techniques for food quality evaluation. Stewart Postharvest Rev 3(1), ISSN: 17459656

Wang Y, Shen M, Ji C (2007f) Using color data and shape properties for cotton fruit recognition. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 38(11):77–79, +87, ISSN: 10001298

Wang M, Wei J, Yuan J, Xu K (2008a) A research for intelligent cotton picking robot based on machine vision. Proceedings of the 2008 IEEE international conference on information and automation, ICIA 2008, art. no. 4608107, pp 800–803, ISBN: 9781424421848

Wang Z, Fu J, Meng H, Yang W (2008b) Small defect extracting based on region growing algorithm and grey relational analysis. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 39(12):166–169, ISSN: 10001298

Weeks AR, Gallagher A, Eriksson J (1999) Detection of oranges from a color image of an orange tree. Proc SPIE Int Soc Opt Eng 3808:346–357, ISSN: 0277786X

Wei X, Zhou X, Li F, Li G (2007) Optimum design of fruits roller conveyor of machine vision grader. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 38(9):98–102, ISSN: 10001298

Wei X, Hu X, Zhou X, Zhang J (2008) Design of a synchronization control system for fruit machine vision grader. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 39(11):99–103, ISSN: 10001298

Wei X, Hu X, Zhou X, Li G (2009) Acquisition method of dynamic image sequence of fruit flow for fruit machine vision grader. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 40(1):171–175, ISSN: 10001298

Wen Z, Tao Y (1997a) Intensity compensation for on-line detection of defects on fruit. Proc SPIE Int Soc Opt Eng 3164:474–481, ISSN: 0277786X

Wen Z, Tao Y (1997b) Adaptive spherical transform of fruit images for high-speed defect detection. Paper Am Soc Agric Eng 1:5, ISSN: 01450166

Wen Z, Tao Y (1998) Brightness-invariant image segmentation for on-line fruit defect detection. Opt Eng 37(11):2948–2952, ISSN: 00913286

Wen Z, Tao Y (1998) Fuzzy-based determination of model and parameters of dual-wavelength vision system for on-line apple sorting. Opt Eng 37(1):293–299, ISSN: 00913286

Wen Z, Tao Y (1998) Dual-wavelength imaging for on-line identification of stem-ends and calyxes. Proc SPIE Int Soc Opt Eng 3460:249–253, ISSN: 0277786X

Wen Z, Tao Y (1999) Building a rule-based machine-vision system for defect inspection on apple sorting and packing lines. Expert Syst Appl 16(3):307–313, ISSN: 09574174

Wen Z, Tao Y (2000) Dual-camera NIR/MIR imaging for stem-end/calyx identification in apple defect sorting. Trans Am Soc Agric Eng 43(2):449–452, ISSN: 00012351

Wen C, Guyer DE, Li W (2009) Local feature-based identification and classification for orchard insects. Biosyst Eng 104(3):299–307, ISSN: 15375110

Wenhua M, Baoping J, Jicheng Z, Xiaochao Z, Xiaoan H (2009) Apple location method for the apple harvesting robot. Proceedings of the 2009. 2nd international congress on image and signal processing, CISP'09, art. no. 5305224, ISBN: 9781424441310

Werth L (1986) Machine vision implementation in agri-applications: The general purpose approach, pp 63–77, ISBN: 091615078X

Wilhoit JH, Kutz LJ, Vandiver WA (1995) Machine vision system for quality control assessment of bareroot pine seedlings. Proc SPIE Int Soc Opt Eng 2345:36–49, ISSN: 0277786X, ISBN: 0819416789

Windham WR, Smith DP, Park B, Lawrence KC, Feldner PW (2003) Algorithm development with visible/near-infrared spectra for detection of poultry feces and ingesta. Trans Am Soc Agric Eng 46(6): 1733–1738, ISSN: 00012351

Winter P, Sokhansanj S, Wood HC (1996) Machine vision methods for use in grain variety discrimination and quality analysis. Proc SPIE Int Soc Opt Eng 2907:230–240, ISSN: 0277786X, ISBN: 0819423092

Wooten JR, White JG, Thomasson JA, Thompson PG (2000) Yield and quality monitor for sweetpotato with machine vision. ASAE annual intenational meeting, technical papers: Engineering solutions for a new century, 1, pp 2979–2996

Wu J-H, Liu Y-D, Ouyang A-G (2005b) Research on real time identification of seed variety by machine vision technology. Chinese J Sensors Actuators 18(4): 742–744, ISSN: 10041699

Wyeth G (1998a) Training a vision guided mobile robot. Autonom Robots 5(3-4):381–394, ISSN: 09295593

Wyeth G (1998b) Training a vision guided mobile robot. Mach Learn 31(1-3):201–222, ISSN: 08856125

Xiaobo Z, Jiewen Z, Yanxiao L (2007) Apple color grading based on organization feature parameters. Pattern Recognit Lett 28(15):2046–2053, ISSN: 01678655

Xie W, Paulsen MR (2001) Machine vision detection of tetrazolium staining in corn. Trans Am Soc Agric Eng 44(2):421–428, ISSN: 00012351

Xie F, Pearson T, Dowell FE, Zhang N (2004) Detecting vitreous wheat kernels using reflectance and transmittance image analysis. Cereal Chem 81(5):594–597, ISSN: 00090352

Xie L-J, Ying Y-B, Yu H-Y, Fu X-P (2007) Application of near infrared spectroscopy technique to nondestructive measurement of vegetable quality. Guang Pu Xue Yu Guang Pu Fen Xi/Spectrosc Spectr Anal 27((6):1131–1135, ISSN: 10000593

Xin H, Shao J (1997) Application of machine vision to swine environmental control. IEEE/ASME international conference on advanced intelligent mechatronics, AIM, p. 14

Xiuqin R, Yibin Y, YiKe C, Haibo H (2006) Laser scatter feature of surface defect on apples. Proc SPIE Int Soc Opt Eng 6381, art. no. 638113, ISSN: 0277786X, ISBN: 0819464791; 9780819464798

Xu H, Ying Y (2004a) Citrus fruit recognition using color image analysis. Proc SPIE Int Soc Opt Eng 5608, art. no. 43, pp 321–328, ISSN: 0277786X

Xu H, Ying Y (2004b) Detection citrus in a tree canopy using infrared thermal imaging. Proc SPIE Int Soc Opt Eng 5271:321–327, ISSN: 0277786X

Xu H-R, Ying Y-B (2004c) Application of infrared thermal imaging in identification of citrus on trees. Hongwai Yu Haomibo Xuebao/J Infrared Millimeter Waves 23(5):353–356, ISSN: 10019014

Xu X, Zhang X (2007) A remote acoustic monitoring system for offshore aquaculture fish cage. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430721, pp 86–90, ISBN: 1424413583; 9781424413584

Xu H, Ye Z, Ying Y (2005) Identification of citrus fruit in a tree canopy using color information. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 21(5):98–101, ISSN: 10026819

Xun Y, Zhang J, Li W, Cai W (2006) Automatic system of seeds refined grading based on machine vision. Proceedings of the world congress on intelligent control and automation (WCICA), 2, art. no. 1713883, pp 9686–9689, ISBN: 1424403324; 9781424403325

Xun Y, Zhang J, Li W, Cai W (2007) Multi-objective dynamic detection of seeds based on machine vision. Prog Nat Sci 17(2):217–221, ISSN: 10020071

Yang Q (1994) An approach to apple surface feature detection by machine vision. Comput Electron Agric 11(2-3):249–264, ISSN: 01681699

Yang Q (1996) Apple stem and calyx identification with machine vision. J Agric Eng Res 63(3):229–236, ISSN: 00218634

Yang L, Dickinson J, Wu QMJ, Lang S (2007) A fruit recognition method for automatic harvesting. Proceedings 14th international conference on mechatronics and machine vision in practice, M2VIP2007, art. no. 4430734, pp 152–157, ISBN: 1424413583; 9781424413584

Yang C-C, Chao K, Kim MS (2009) Automatic inspection using machine vision for food safety. ASABE – 7th world congress on computers in agriculture and natural resources 2009, WCCA 2009, pp 63–72, ISBN: 9781615673902

Yencharis Len (1999b) Machine vision: Dependable markets beyond the ups and downs of semiconductors? Adv Imag 14(1):2, ISSN: 10420711

Yin J, Mao H, Hu Y, Wang X, Chen S (2006a) An automatic segmentation method for multi-tomatoes image under complicated natural background. Proc SPIE Int Soc Opt Eng 6411, art. no. 641118, ISSN: 0277786X, ISBN: 0819465186; 9780819465184

Yin J, Mao H, Wang X, Chen S, Zhang J (2006b) Automatic segmentation method for multi-tomato images under various growth conditions. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 22(10):149–153, ISSN: 10026819

Ying Y, Fu F (2004) Color transformation model of fruit image in process of non-destructive quality inspection based on machine vision. Nongye Jixie Xuebao/ Trans Chinese Soc Agric Mach 35(1):85, ISSN: 10001298

Ying Y, Jing H, Tao Y, Zhang N (2003) Detecting stem and shape of pears using fourier transformation and an artificial neural network. Trans Am Soc Agric Eng 46(1):157–162, ISSN: 00012351

Ying Y-B, Gui J-S, Rao X-Q (2007) Fruit shape classification based on Zernike moments. Jiangsu Daxue Xuebao (Ziran Kexue Ban)/J Jiangsu Univ (Nat Sci Edn) 28(1):1–3, +67, ISSN: 16717775

Yonekawa S, Sakai N, Kitani O (1994) Three-dimensional measurement of loam clods using machine vision. Trans Am Soc Agric Eng 37(3):1003–1009

Yu X-L, Zhao Z-M (2010) Spectral navigation technology and its application in positioning the fruits of fruit trees. Guang Pu Xue Yu Guang Pu Fen Xi/Spectrosc Spectr Anal 30(3):770–773, ISSN: 10000593

Yuan T, Li W, Tan Y, Yang Q, Gao F, Ren Y (2009a) Information acquisition for cucumber harvesting robot in greenhouse. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 40(10):151–155, ISSN: 10001298

Yuan T, Zhang J, Li W, Ren Y (2009b) Feature acquisition of cucumber fruit in unstructured environment using machine vision. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 40(8):170–174, +218, ISSN: 10001298

Zadeh A, Schrieber M (2002) Color systems see what grayscales miss. Mach Design 73(Suppl):S30–S34, ISSN: 00249114

Zaman QUZ, Zhang F, Schumann AW, Percival DC (2009) Bare spots mapping in wild blueberry fields using digital photography. American society of agricultural and biological engineers annual international meeting 2009, 1, pp 450–457, ISBN: 9781615673629

Zayas IY, Flinn PW (1998) Detection of insects in bulk wheat samples with machine vision. Trans Am Soc Agric Eng 41(3):883–888, ISSN: 00012351

Zhang W, Du S (2005) Hough transform applying in vision navigation of farming machine. Yi Qi Yi Biao Xue Bao/Chinese J Sci Instrument 26(Suppl): 706–707, ISSN: 02543087

Zhang F, Ying Y (2006) A 2D CMAC neural net algorithm for a positioning system of automated agriculture vehicle. Proc SPIE Int Soc Opt Eng 6384, art. no. 63840Y, ISSN: 0277786X, ISBN: 0819464821; 9780819464828

Zhang H, Cheng F (2007) Determination of rice seed vigor using digital image processing technology. ASABE Annual International Meeting, Technical Papers, 7 book, 7 p

Zhang M, Ludas LI, Morgan MT, Krutz GW, Precetti CJ (1999) Applications of color machine vision in the agricultural and food industries. Proc SPIE Int Soc Opt Eng 3543:208–219, ISSN: 0277786X

Zhang Y, Qiao X, Wang C, Zhang X, Tian H (2005a) Development of plant leaf area measurement instrument based on virtual instrument technology. Nongye Gongcheng Xuebao/Trans Chinese Soc Agric Eng 21(Suppl 2):200–203, ISSN: 10026819

Zhang F-M, Ying Y-B, Jiang H-Y, Shin B-S (2005b) Correlation analysis-based image segmentation approach for automatic agriculture vehicle. J Zhejiang Univ Sci 6(10):1158–1162, ISSN: 10093095

Zhang Y, Li M, Qiao J, Liu G (2008) Segmentation algorithm for apple recognition using image features and artificial neural network. Guangxue Xuebao/ Acta Optica Sinica 28(11):2104–2108, ISSN: 02532239

Zhao J, Liu S, Zou X, Shi J, Yin X (2008) Recognition of defect Chinese dates by machine vision and support vector machine, Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 39(3):113–115+147, ISSN: 10001298

Zhang F, Zaman Q, Schumann AW, Percival DC, Nams D, Esau T (2009) Detecting weeds in wild blueberry field based on color images. American society of agricultural and biological engineers annual international meeting 2009, 5, pp 2779–2786, ISBN: 9781615673629

Zhang Y, Li M, Liu G, Qiao J (2009) Separating adjoined apples based on machine vision and information fusion. Nongye Jixie Xuebao/Trans Chinese Soc Agric Mach 40(11):180–183, ISSN: 10001298

Zhang Y, Staab ES, Slaughter DC, Giles DK, Downey D (2009) Precision automated weed control using hyperspectral vision identification and heated oil. American society of agricultural and biological engineers annual international meeting 2009, 6, pp 3567–3583, ISBN: 9781615673629

Zhao X, Burks TF, Qin J, Ritenour MA (2009) Digital microscopic imaging for citrus peel disease classification using color texture features. Appl Eng Agric 25(5):769–776, ISSN: 08838542

Zhao Y, Wang D, Qian D (2009) Machine vision based image analysis for the estimation of pear external quality. 2nd international conference on intelligent computing technology and automation, ICICTA 2009, 1, art. no. 5287572, pp 629–632, ISBN: 9780769538044

Zhen Z, Qixin C, Yang Y, Nianjiong X (2009) A feature extraction method for galvanized steel sheet powdering rates classification. Imag Sci J 57(2):94–100, ISSN: 13682199

Zhou L, Chalana V, Kim Y (1998) PC-based machine vision system for real-time computer-aided potato inspection. Int J Imag Syst Technol 9(6):423–433, ISSN: 08999457

Zhu H, Liu W (2006) Fruit image segmentation based on a novel watershed algorithm. Proc SPIE Int Soc Opt Eng 6027 II, art. no. 602734, ISSN: 0277786X

Zhu H, Liu W-Y, Wang J-T, Hao Y-J (2004) Application of a novel segmentation algorithm to robot vision of fruit industry. Guangdian Gongcheng/Opto-Electr Eng 31(2):54–58, ISSN: 1003501X

Zhu H, Liu W, Wang J (2005) Implementation of a novel watershed algorithm, MAPE2005: IEEE 2005 international symposium on microwave, antenna, propagation and EMC technologies for wireless communications. Proceedings, 2, art. no. 1618125, pp 1150–1153, ISBN: 0780391284; 9780780391284

Zhu B, Jiang L, Tao Y (2007) Automated 3D surface reconstruction and analysis of apple near-infrared data for the application of apple stem-end/calyx identification. ASABE annual international meeting. Technical papers, 7 book, 14 p

Zou X, Zhao J (2005) Apple quality assessment by fusion three sensors. Proceedings of IEEE sensors, 2005, art. no. 1597717, pp 389–392, ISBN: 0780390563; 9780780390560

Zuech N (1991) Introduction to color based machine vision. Sensors (Peterborough, NH) 8(5):37–39, ISSN: 07469462

Zwiggelaar R, Yang Q, Garcia-Pardo E, Bull CR (1996) Use of spectral information and machine vision for bruise detection on peaches and apricots. J Agric Eng Res 63(4):323–332, ISSN: 00218634

Index