You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

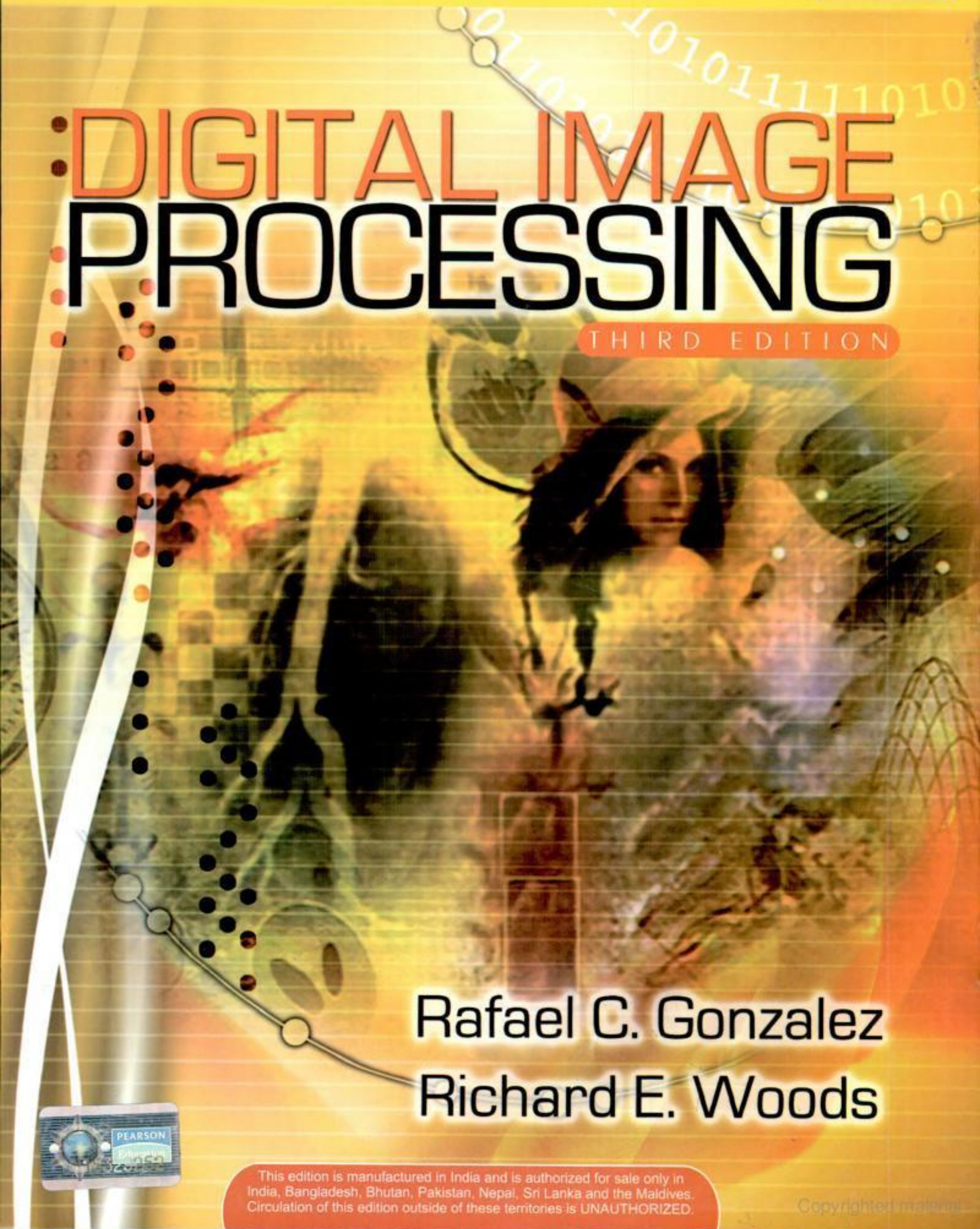
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

PEARSON Education

DIGITAL IMAGE PROCESSING

THIRD EDITION



Rafael C. Gonzalez
Richard E. Woods

Digital Image Processing

Third Edition

Rafael C. Gonzalez
University of Tennessee

Richard E. Woods
MedData Interactive

PEARSON
Education

Original Edition entitled *Digital Image Processing, Third Edition,* by Gonzalez, Rafael C.; Woods, Richard E., published by Pearson Education, Inc, publishing as Prentice Hall, Copyright © 2008

Indian edition published by Dorling Kindersley India Pvt. Ltd. Copyright © 2009

ISBN 978-81-317-2695-2

First Impression, 2009

This edition is manufactured in India and is authorized for sale only in India, Bangladesh, Bhutan, Pakistan, Nepal, Sri Lanka and the Maldives. Circulation of this edition outside of these territories is UNAUTHORIZED.

Published by Dorling Kindersley (India) Pvt. Ltd., licensees of Pearson Education in South Asia.

Head Office: 482, F.I.E., Patparganj, Delhi 110 092, India.
Registered Office: 14 Local Shopping Centre, Panchsheel Park, New Delhi 110 017, India.

Printed in India by India Binding House.

Contents

Preface

When something can be read without effort,
great effort has gone into its writing.
Enrique Jardiel Poncela

This edition of *Digital Image Processing* is a major revision of the book. As in the 1977 and 1987 editions by Gonzalez and Wintz, and the 1992 and 2002 editions by Gonzalez and Woods, this fifth-generation edition was prepared with students and instructors in mind. The principal objectives of the book continue to be to provide an introduction to basic concepts and methodologies for digital image processing, and to develop a foundation that can be used as the basis for further study and research in this field. To achieve these objectives, we focused again on material that we believe is fundamental and whose scope of application is not limited to the solution of specialized problems. The mathematical complexity of the book remains at a level well within the grasp of college seniors and first-year graduate students who have introductory preparation in mathematical analysis, vectors, matrices, probability, statistics, linear systems, and computer programming. The book Web site provides tutorials to support readers needing a review of this background material.

One of the principal reasons this book has been the world leader in its field for more than 30 years is the level of attention we pay to the changing educational needs of our readers. The present edition is based on the most extensive survey we have ever conducted. The survey involved faculty, students, and independent readers of the book in 134 institutions from 32 countries. The major findings of the survey indicated a need for:

- A more comprehensive introduction early in the book to the mathematical tools used in image processing.
- An expanded explanation of histogram processing techniques.
- Stating complex algorithms in step-by-step summaries.
- An expanded explanation of spatial correlation and convolution.
- An introduction to fuzzy set theory and its application to image processing.
- A revision of the material dealing with the frequency domain, starting with basic principles and showing how the discrete Fourier transform follows from data sampling.
- Coverage of computed tomography (CT).
- Clarification of basic concepts in the wavelets chapter.
- A revision of the data compression chapter to include more video compression techniques, updated standards, and watermarking.
- Expansion of the chapter on morphology to include morphological reconstruction and a revision of gray-scale morphology.

- Expansion of the coverage on image segmentation to include more advanced edge detection techniques such as Canny's algorithm, and a more comprehensive treatment of image thresholding.
- An update of the chapter dealing with image representation and description.
- Streamlining the material dealing with structural object recognition.

The new and reorganized material that resulted in the present edition is our attempt at providing a reasonable degree of balance between rigor, clarity of presentation, and the findings of the market survey, while at the same time keeping the length of the book at a manageable level. The major changes in this edition of the book are as follows.

Chapter 1: A few figures were updated and part of the text was rewritten to correspond to changes in later chapters.

Chapter 2: Approximately 50% of this chapter was revised to include new images and clearer explanations. Major revisions include a new section on image interpolation and a comprehensive new section summarizing the principal mathematical tools used in the book. Instead of presenting "dry" mathematical concepts one after the other, however, we took this opportunity to bring into Chapter 2 a number of image processing applications that were scattered throughout the book. For example, image averaging and image subtraction were moved to this chapter to illustrate arithmetic operations. This follows a trend we began in the second edition of the book to move as many applications as possible early in the discussion not only as illustrations, but also as motivation for students. After finishing the newly organized Chapter 2, a reader will have a basic understanding of how digital images are manipulated and processed. This is a solid platform upon which the rest of the book is built.

Chapter 3: Major revisions of this chapter include a detailed discussion of spatial correlation and convolution, and their application to image filtering using spatial masks. We also found a consistent theme in the market survey asking for numerical examples to illustrate histogram equalization and specification, so we added several such examples to illustrate the mechanics of these processing tools. Coverage of fuzzy sets and their application to image processing was also requested frequently in the survey. We included in this chapter a new section on the foundation of fuzzy set theory, and its application to intensity transformations and spatial filtering, two of the principal uses of this theory in image processing.

Chapter 4: The topic we heard most about in comments and suggestions during the past four years dealt with the changes we made in Chapter 4 from the first to the second edition. Our objective in making those changes was to simplify the presentation of the Fourier transform and the frequency domain. Evidently, we went too far, and numerous users of the book complained that the new material was too superficial. We corrected that problem in the present edition. The material now begins with the Fourier transform of one continuous variable and proceeds to derive the discrete Fourier transform starting with basic concepts of sampling and convolution. A by-product of the flow of this

material is an intuitive derivation of the sampling theorem and its implications. The 1-D material is then extended to 2-D, where we give a number of examples to illustrate the effects of sampling on digital images, including aliasing and moiré patterns. The 2-D discrete Fourier transform is then illustrated and a number of important properties are derived and summarized. These concepts are then used as the basis for filtering in the frequency domain. Finally, we discuss implementation issues such as transform decomposition and the derivation of a fast Fourier transform algorithm. At the end of this chapter, the reader will have progressed from sampling of 1-D functions through a clear derivation of the foundation of the discrete Fourier transform and some of its most important uses in digital image processing.

Chapter 5: The major revision in this chapter was the addition of a section dealing with image reconstruction from projections, with a focus on computed tomography (CT). Coverage of CT starts with an intuitive example of the underlying principles of image reconstruction from projections and the various imaging modalities used in practice. We then derive the Radon transform and the Fourier slice theorem and use them as the basis for formulating the concept of filtered backprojections. Both parallel- and fan-beam reconstruction are discussed and illustrated using several examples. Inclusion of this material was long overdue and represents an important addition to the book.

Chapter 6: Revisions to this chapter were limited to clarifications and a few corrections in notation. No new concepts were added.

Chapter 7: We received numerous comments regarding the fact that the transition from previous chapters into wavelets was proving difficult for beginners. Several of the foundation sections were rewritten in an effort to make the material clearer.

Chapter 8: This chapter was rewritten completely to bring it up to date. New coding techniques, expanded coverage of video, a revision of the section on standards, and an introduction to image watermarking are among the major changes. The new organization will make it easier for beginning students to follow the material.

Chapter 9: The major changes in this chapter are the inclusion of a new section on morphological reconstruction and a complete revision of the section on gray-scale morphology. The inclusion of morphological reconstruction for both binary and gray-scale images made it possible to develop more complex and useful morphological algorithms than before.

Chapter 10: This chapter also underwent a major revision. The organization is as before, but the new material includes greater emphasis on basic principles as well as discussion of more advanced segmentation techniques. Edge models are discussed and illustrated in more detail, as are properties of the gradient. The Marr-Hildreth and Canny edge detectors are included to illustrate more advanced edge detection techniques. The section on thresholding was rewritten also to include Otsu's method, an optimum thresholding technique whose popularity has increased significantly over the past few years. We introduced this approach in favor of optimum thresholding based on the Bayes classification rule, not only because it is easier to understand and implement, but also

because it is used considerably more in practice. The Bayes approach was moved to Chapter 12, where the Bayes decision rule is discussed in more detail. We also added a discussion on how to use edge information to improve thresholding and several new adaptive thresholding examples. Except for minor clarifications, the sections on morphological watersheds and the use of motion for segmentation are as in the previous edition.

Chapter 11: The principal changes in this chapter are the inclusion of a boundary-following algorithm, a detailed derivation of an algorithm to fit a minimum-perimeter polygon to a digital boundary, and a new section on co-occurrence matrices for texture description. Numerous examples in Sections 11.2 and 11.3 are new, as are all the examples in Section 11.4.

Chapter 12: Changes in this chapter include a new section on matching by correlation and a new example on using the Bayes classifier to recognize regions of interest in multispectral images. The section on structural classification now limits discussion only to string matching.

All the revisions just mentioned resulted in over 400 new images, over 200 new line drawings and tables, and more than 80 new homework problems. Where appropriate, complex processing procedures were summarized in the form of step-by-step algorithm formats. The references at the end of all chapters were updated also.

The book Web site, established during the launch of the second edition, has been a success, attracting more than 20,000 visitors each month. The site was redesigned and upgraded to correspond to the launch of this edition. For more details on features and content, see *The Book Web Site*, following the *Acknowledgments*.

This edition of *Digital Image Processing* is a reflection of how the educational needs of our readers have changed since 2002. As is usual in a project such as this, progress in the field continues after work on the manuscript stops. One of the reasons why this book has been so well accepted since it first appeared in 1977 is its continued emphasis on fundamental concepts—an approach that, among other things, attempts to provide a measure of stability in a rapidly evolving body of knowledge. We have tried to follow the same principle in preparing this edition of the book.

R. C. G.
R. E. W.

Acknowledgments

We are indebted to a number of individuals in academic circles as well as in industry and government who have contributed to this edition of the book. Their contributions have been important in so many different ways that we find it difficult to acknowledge them in any other way but alphabetically. In particular, we wish to extend our appreciation to our colleagues Mongi A. Abidi, Steven L. Eddins, Yongmin Kim, Bryan Morse, Andrew Oldroyd, Ali M. Reza, Edgardo Felipe Riveron, Jose Ruiz Shulcloper, and Cameron H. G. Wright for their many suggestions on how to improve the presentation and/or the scope of coverage in the book.

Numerous individuals and organizations provided us with valuable assistance during the writing of this edition. Again, we list them alphabetically. We are particularly indebted to Courtney Esposito and Naomi Fernandes at The Mathworks for providing us with MATLAB software and support that were important in our ability to create or clarify many of the examples and experimental results included in this edition of the book. A significant percentage of the new images used in this edition (and in some cases their history and interpretation) were obtained through the efforts of individuals whose contributions are sincerely appreciated. In particular, we wish to acknowledge the efforts of Serge Beucher, Melissa D. Binde, James Blankenship, Uwe Boos, Ernesto Bribiesca, Michael E. Casey, Michael W. Davidson, Susan L. Forsburg, Thomas R. Gest, Lalit Gupta, Daniel A. Hammer, Zhong He, Roger Heady, Juan A. Herrera, John M. Hudak, Michael Hurwitz, Chris J. Johannsen, Rhonda Knighton, Don P. Mitchell, Ashley Mohamed, A. Morris, Curtis C. Ober, Joseph E. Pascente, David. R. Pickens, Michael Robinson, Barrett A. Schaefer, Michael Shaffer, Pete Sites, Sally Stowe, Craig Watson, David K. Wehe, and Robert A. West. We also wish to acknowledge other individuals and organizations cited in the captions of numerous figures throughout the book for their permission to use that material.

Special thanks go to Vince O'Brien, Rose Kernan, Scott Disanno, Michael McDonald, Joe Ruddick, Heather Scott, and Alice Dworkin, at Prentice Hall. Their creativity, assistance, and patience during the production of this book are truly appreciated.

R.C.G.
R.E.W.

The Book Web Site

www.prenhall.com/gonzalezwoods
or its mirror site,
www.imageprocessingplace.com

Digital Image Processing is a completely self-contained book. However, the companion Web site offers additional support in a number of important areas.

For the Student or Independent Reader the site contains

- Reviews in areas such as probability, statistics, vectors, and matrices.
- Complete solutions to selected problems.
- Computer projects.
- A Tutorials section containing dozens of tutorials on most of the topics discussed in the book.
- A database containing all the images in the book.

For the Instructor the site contains

- An *Instructor's Manual* with complete solutions to all the problems in the book, as well as course and laboratory teaching guidelines. The manual is available free of charge to instructors who have adopted the book for classroom use.
- Classroom presentation materials in PowerPoint format.
- Material removed from previous editions, downloadable in convenient PDF format.
- Numerous links to other educational resources.

For the Practitioner the site contains additional specialized topics such as

- Links to commercial sites.
- Selected new references.
- Links to commercial image databases.

The Web site is an ideal tool for keeping the book current between editions by including new topics, digital images, and other relevant material that has appeared after the book was published. Although considerable care was taken in the production of the book, the Web site is also a convenient repository for any errors that may be discovered between printings. References to the book Web site are designated in the book by the following icon:

About the Authors

Rafael C. Gonzalez

R. C. Gonzalez received the B.S.E.E. degree from the University of Miami in 1965 and the M.E. and Ph.D. degrees in electrical engineering from the University of Florida, Gainesville, in 1967 and 1970, respectively. He joined the Electrical and Computer Engineering Department at the University of Tennessee, Knoxville (UTK) in 1970, where he became Associate Professor in 1973, Professor in 1978, and Distinguished Service Professor in 1984. He served as Chairman of the department from 1994 through 1997. He is currently a Professor Emeritus at UTK.

Gonzalez is the founder of the Image & Pattern Analysis Laboratory and the Robotics & Computer Vision Laboratory at the University of Tennessee. He also founded Perceptics Corporation in 1982 and was its president until 1992. The last three years of this period were spent under a full-time employment contract with Westinghouse Corporation, who acquired the company in 1989.

Under his direction, Perceptics became highly successful in image processing, computer vision, and laser disk storage technology. In its initial ten years, Perceptics introduced a series of innovative products, including: The world's first commercially available computer vision system for automatically reading license plates on moving vehicles; a series of large-scale image processing and archiving systems used by the U.S. Navy at six different manufacturing sites throughout the country to inspect the rocket motors of missiles in the Trident II Submarine Program; the market-leading family of imaging boards for advanced Macintosh computers; and a line of trillion-byte laser disk products.

He is a frequent consultant to industry and government in the areas of pattern recognition, image processing, and machine learning. His academic honors for work in these fields include the 1977 UTK College of Engineering Faculty Achievement Award; the 1978 UTK Chancellor's Research Scholar Award; the 1980 Magnavox Engineering Professor Award; and the 1980 M.E. Brooks Distinguished Professor Award. In 1981, he became an IBM Professor at the University of Tennessee, and in 1984, he was named a Distinguished Service Professor there. He was awarded a Distinguished Alumnus Award by the University of Miami in 1985, the Phi Kappa Phi Scholar Award in 1986, and the University of Tennessee's Nathan W. Dougherty Award for Excellence in Engineering in 1992.

Honors for industrial accomplishment include the 1987 IEEE Outstanding Engineer Award for Commercial Development in Tennessee; the 1988 Albert Rose Nat'l Award for Excellence in Commercial Image Processing; the 1989 B. Otto Wheeley Award for Excellence in Technology Transfer; the 1989 Coopers and Lybrand Entrepreneur of the Year Award; the 1992 IEEE Region 3 Outstanding Engineer Award; and the 1993 Automated Imaging Association National Award for Technology Development.

Gonzalez is author or co-author of over 100 technical articles, two edited books, and four textbooks in the fields of pattern recognition, image processing, and robotics. His books are used in over 1000 universities and research institutions throughout the world. He is listed in the prestigious Marquis *Who's Who in America*, Marquis *Who's Who in Engineering*, Marquis *Who's Who in the World*, and in 10 other national and international biographical citations. He is the co-holder of two U.S. Patents, and has been an associate editor of the *IEEE Transactions on Systems, Man and Cybernetics*, and the *International Journal of Computer and Information Sciences*. He is a member of numerous professional and honorary societies, including Tau Beta Pi, Phi Kappa Phi, Eta Kappa Nu, and Sigma Xi. He is a Fellow of the IEEE.

Richard E. Woods

Richard E. Woods earned his B.S., M.S., and Ph.D. degrees in Electrical Engineering from the University of Tennessee, Knoxville. His professional experiences range from entrepreneurial to the more traditional academic, consulting, governmental, and industrial pursuits. Most recently, he founded MedData Interactive, a high technology company specializing in the development of handheld computer systems for medical applications. He was also a founder and Vice President of Perceptics Corporation, where he was responsible for the development of many of the company's quantitative image analysis and autonomous decision-making products.

Prior to Perceptics and MedData, Dr. Woods was an Assistant Professor of Electrical Engineering and Computer Science at the University of Tennessee and prior to that, a computer applications engineer at Union Carbide Corporation. As a consultant, he has been involved in the development of a number of special-purpose digital processors for a variety of space and military agencies, including NASA, the Ballistic Missile Systems Command, and the Oak Ridge National Laboratory.

Dr. Woods has published numerous articles related to digital signal processing and is a member of several professional societies, including Tau Beta Pi, Phi Kappa Phi, and the IEEE. In 1986, he was recognized as a Distinguished Engineering Alumnus of the University of Tennessee.

1 Introduction

One picture is worth more than ten thousand words.

Anonymous

Preview

Interest in digital image processing methods stems from two principal application areas: improvement of pictorial information for human interpretation; and processing of image data for storage, transmission, and representation for autonomous machine perception. This chapter has several objectives: (1) to define the scope of the field that we call image processing; (2) to give a historical perspective of the origins of this field; (3) to give you an idea of the state of the art in image processing by examining some of the principal areas in which it is applied; (4) to discuss briefly the principal approaches used in digital image processing; (5) to give an overview of the components contained in a typical, general-purpose image processing system; and (6) to provide direction to the books and other literature where image processing work normally is reported.

1.1 What Is Digital Image Processing?

An image may be defined as a two-dimensional function, $f(x, y)$, where x and y are *spatial* (plane) coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the *intensity* or *gray level* of the image at that point. When x, y, and the intensity values of f are all finite, discrete quantities, we call the image a *digital image*. The field of *digital image processing* refers to processing digital images by means of a digital computer. Note that a digital image is composed of a finite number of elements, each of which has a particular location

and value. These elements are called *picture elements*, *image elements*, *pels*, and *pixels*. *Pixel* is the term used most widely to denote the elements of a digital image. We consider these definitions in more formal terms in Chapter 2.

Vision is the most advanced of our senses, so it is not surprising that images play the single most important role in human perception. However, unlike humans, who are limited to the visual band of the electromagnetic (EM) spectrum, imaging machines cover almost the entire EM spectrum, ranging from gamma to radio waves. They can operate on images generated by sources that humans are not accustomed to associating with images. These include ultrasound, electron microscopy, and computer-generated images. Thus, digital image processing encompasses a wide and varied field of applications.

There is no general agreement among authors regarding where image processing stops and other related areas, such as image analysis and computer vision, start. Sometimes a distinction is made by defining image processing as a discipline in which both the input and output of a process are images. We believe this to be a limiting and somewhat artificial boundary. For example, under this definition, even the trivial task of computing the average intensity of an image (which yields a single number) would not be considered an image processing operation. On the other hand, there are fields such as computer vision whose ultimate goal is to use computers to emulate human vision, including learning and being able to make inferences and take actions based on visual inputs. This area itself is a branch of artificial intelligence (AI) whose objective is to emulate human intelligence. The field of AI is in its earliest stages of infancy in terms of development, with progress having been much slower than originally anticipated. The area of image analysis (also called image understanding) is in between image processing and computer vision.

There are no clear-cut boundaries in the continuum from image processing at one end to computer vision at the other. However, one useful paradigm is to consider three types of computerized processes in this continuum: low-, mid-, and high-level processes. Low-level processes involve primitive operations such as image preprocessing to reduce noise, contrast enhancement, and image sharpening. A low-level process is characterized by the fact that both its inputs and outputs are images. Mid-level processing on images involves tasks such as segmentation (partitioning an image into regions or objects), description of those objects to reduce them to a form suitable for computer processing, and classification (recognition) of individual objects. A mid-level process is characterized by the fact that its inputs generally are images, but its outputs are attributes extracted from those images (e.g., edges, contours, and the identity of individual objects). Finally, higher-level processing involves "making sense" of an ensemble of recognized objects, as in image analysis, and, at the far end of the continuum, performing the cognitive functions normally associated with vision.

Based on the preceding comments, we see that a logical place of overlap between image processing and image analysis is the area of recognition of individual regions or objects in an image. Thus, what we call in this book *digital image processing* encompasses processes whose inputs and outputs are images

and, in addition, encompasses processes that extract attributes from images, up to and including the recognition of individual objects. As an illustration to clarify these concepts, consider the area of automated analysis of text. The processes of acquiring an image of the area containing the text, preprocessing that image, extracting (segmenting) the individual characters, describing the characters in a form suitable for computer processing, and recognizing those individual characters are in the scope of what we call digital image processing in this book. Making sense of the content of the page may be viewed as being in the domain of image analysis and even computer vision, depending on the level of complexity implied by the statement "making sense." As will become evident shortly, digital image processing, as we have defined it, is used successfully in a broad range of areas of exceptional social and economic value. The concepts developed in the following chapters are the foundation for the methods used in those application areas.

1.2 The Origins of Digital Image Processing

One of the first applications of digital images was in the newspaper industry, when pictures were first sent by submarine cable between London and New York. Introduction of the Bartlane cable picture transmission system in the early 1920s reduced the time required to transport a picture across the Atlantic from more than a week to less than three hours. Specialized printing equipment coded pictures for cable transmission and then reconstructed them at the receiving end. Figure 1.1 was transmitted in this way and reproduced on a telegraph printer fitted with typefaces simulating a halftone pattern.

Some of the initial problems in improving the visual quality of these early digital pictures were related to the selection of printing procedures and the distribution of intensity levels. The printing method used to obtain Fig. 1.1 was abandoned toward the end of 1921 in favor of a technique based on photographic reproduction made from tapes perforated at the telegraph receiving terminal. Figure 1.2 shows an image obtained using this method. The improvements over Fig. 1.1 are evident, both in tonal quality and in resolution.

FIGURE 1.1 A digital picture produced in 1921 from a coded tape by a telegraph printer with special type faces. (McFarlane.[†])

[†]References in the Bibliography at the end of the book are listed in alphabetical order by authors' last names.

FIGURE 1.2 A digital picture made in 1922 from a tape punched after the signals had crossed the Atlantic twice. (McFarlane.)

The early Bartlane systems were capable of coding images in five distinct levels of gray. This capability was increased to 15 levels in 1929. Figure 1.3 is typical of the type of images that could be obtained using the 15-tone equipment. During this period, introduction of a system for developing a film plate via light beams that were modulated by the coded picture tape improved the reproduction process considerably.

Although the examples just cited involve digital images, they are not considered digital image processing results in the context of our definition because computers were not involved in their creation. Thus, the history of digital image processing is intimately tied to the development of the digital computer. In fact, digital images require so much storage and computational power that progress in the field of digital image processing has been dependent on the development of digital computers and of supporting technologies that include data storage, display, and transmission.

The idea of a computer goes back to the invention of the abacus in Asia Minor, more than 5000 years ago. More recently, there were developments in the past two centuries that are the foundation of what we call a computer today. However, the basis for what we call a *modern* digital computer dates back to only the 1940s with the introduction by John von Neumann of two key concepts: (1) a memory to hold a stored program and data, and (2) conditional branching. These two ideas are the foundation of a central processing unit (CPU), which is at the heart of computers today. Starting with von Neumann, there were a series of key advances that led to computers powerful enough to

FIGURE 1.3 Unretouched cable picture of Generals Pershing and Foch, transmitted in 1929 from London to New York by 15-tone equipment. (McFarlane.)

be used for digital image processing. Briefly, these advances may be summarized as follows: (1) the invention of the transistor at Bell Laboratories in 1948; (2) the development in the 1950s and 1960s of the high-level programming languages COBOL (Common Business-Oriented Language) and FORTRAN (Formula Translator); (3) the invention of the integrated circuit (IC) at Texas Instruments in 1958; (4) the development of operating systems in the early 1960s; (5) the development of the microprocessor (a single chip consisting of the central processing unit, memory, and input and output controls) by Intel in the early 1970s; (6) introduction by IBM of the personal computer in 1981; and (7) progressive miniaturization of components, starting with large scale integration (LI) in the late 1970s, then very large scale integration (VLSI) in the 1980s, to the present use of ultra large scale integration (ULSI). Concurrent with these advances were developments in the areas of mass storage and display systems, both of which are fundamental requirements for digital image processing.

The first computers powerful enough to carry out meaningful image processing tasks appeared in the early 1960s. The birth of what we call digital image processing today can be traced to the availability of those machines and to the onset of the space program during that period. It took the combination of those two developments to bring into focus the potential of digital image processing concepts. Work on using computer techniques for improving images from a space probe began at the Jet Propulsion Laboratory (Pasadena, California) in 1964 when pictures of the moon transmitted by *Ranger 7* were processed by a computer to correct various types of image distortion inherent in the on-board television camera. Figure 1.4 shows the first image of the moon taken by *Ranger 7* on July 31, 1964 at 9:09 A.M. Eastern Daylight Time (EDT), about 17 minutes before impacting the lunar surface (the markers, called *reseau* marks, are used for geometric corrections, as discussed in Chapter 2). This also is the first image of the moon taken by a U.S. spacecraft. The imaging lessons learned with *Ranger 7* served as the basis for improved methods used to enhance and restore images from the Surveyor missions to the moon, the Mariner series of flyby missions to Mars, the Apollo manned flights to the moon, and others.

FIGURE 1.4 The first picture of the moon by a U.S. spacecraft. *Ranger 7* took this image on July 31, 1964 at 9:09 A.M. EDT, about 17 minutes before impacting the lunar surface. (Courtesy of NASA.)

In parallel with space applications, digital image processing techniques began in the late 1960s and early 1970s to be used in medical imaging, remote Earth resources observations, and astronomy. The invention in the early 1970s of computerized axial tomography (CAT), also called computerized tomography (CT) for short, is one of the most important events in the application of image processing in medical diagnosis. Computerized axial tomography is a process in which a ring of detectors encircles an object (or patient) and an X-ray source, concentric with the detector ring, rotates about the object. The X-rays pass through the object and are collected at the opposite end by the corresponding detectors in the ring. As the source rotates, this procedure is repeated. Tomography consists of algorithms that use the sensed data to construct an image that represents a "slice" through the object. Motion of the object in a direction perpendicular to the ring of detectors produces a set of such slices, which constitute a three-dimensional (3-D) rendition of the inside of the object. Tomography was invented independently by Sir Godfrey N. Hounsfield and Professor Allan M. Cormack, who shared the 1979 Nobel Prize in Medicine for their invention. It is interesting to note that X-rays were discovered in 1895 by Wilhelm Conrad Roentgen, for which he received the 1901 Nobel Prize for Physics. These two inventions, nearly 100 years apart, led to some of the most important applications of image processing today.

From the 1960s until the present, the field of image processing has grown vigorously. In addition to applications in medicine and the space program, digital image processing techniques now are used in a broad range of applications. Computer procedures are used to enhance the contrast or code the intensity levels into color for easier interpretation of X-rays and other images used in industry, medicine, and the biological sciences. Geographers use the same or similar techniques to study pollution patterns from aerial and satellite imagery. Image enhancement and restoration procedures are used to process degraded images of unrecoverable objects or experimental results too expensive to duplicate. In archeology, image processing methods have successfully restored blurred pictures that were the only available records of rare artifacts lost or damaged after being photographed. In physics and related fields, computer techniques routinely enhance images of experiments in areas such as high-energy plasmas and electron microscopy. Similarly successful applications of image processing concepts can be found in astronomy, biology, nuclear medicine, law enforcement, defense, and industry.

These examples illustrate processing results intended for human interpretation. The second major area of application of digital image processing techniques mentioned at the beginning of this chapter is in solving problems dealing with machine perception. In this case, interest is on procedures for extracting from an image information in a form suitable for computer processing. Often, this information bears little resemblance to visual features that humans use in interpreting the content of an image. Examples of the type of information used in machine perception are statistical moments, Fourier transform coefficients, and multidimensional distance measures. Typical problems in machine perception that routinely utilize image processing techniques are automatic character recognition, industrial machine vision for product assembly and inspection,

military recognizance, automatic processing of fingerprints, screening of X-rays and blood samples, and machine processing of aerial and satellite imagery for weather prediction and environmental assessment. The continuing decline in the ratio of computer price to performance and the expansion of networking and communication bandwidth via the World Wide Web and the Internet have created unprecedented opportunities for continued growth of digital image processing. Some of these application areas are illustrated in the following section.

1.3 Examples of Fields that Use Digital Image Processing

Today, there is almost no area of technical endeavor that is not impacted in some way by digital image processing. We can cover only a few of these applications in the context and space of the current discussion. However, limited as it is, the material presented in this section will leave no doubt in your mind regarding the breadth and importance of digital image processing. We show in this section numerous areas of application, each of which routinely utilizes the digital image processing techniques developed in the following chapters. Many of the images shown in this section are used later in one or more of the examples given in the book. All images shown are digital.

The areas of application of digital image processing are so varied that some form of organization is desirable in attempting to capture the breadth of this field. One of the simplest ways to develop a basic understanding of the extent of image processing applications is to categorize images according to their source (e.g., visual, X-ray, and so on). The principal energy source for images in use today is the electromagnetic energy spectrum. Other important sources of energy include acoustic, ultrasonic, and electronic (in the form of electron beams used in electron microscopy). Synthetic images, used for modeling and visualization, are generated by computer. In this section we discuss briefly how images are generated in these various categories and the areas in which they are applied. Methods for converting images into digital form are discussed in the next chapter.

Images based on radiation from the EM spectrum are the most familiar, especially images in the X-ray and visual bands of the spectrum. Electromagnetic waves can be conceptualized as propagating sinusoidal waves of varying wavelengths, or they can be thought of as a stream of massless particles, each traveling in a wavelike pattern and moving at the speed of light. Each massless particle contains a certain amount (or bundle) of energy. Each bundle of energy is called a *photon*. If spectral bands are grouped according to energy per photon, we obtain the spectrum shown in Fig. 1.5, ranging from gamma rays (highest energy) at one end to radio waves (lowest energy) at the other.

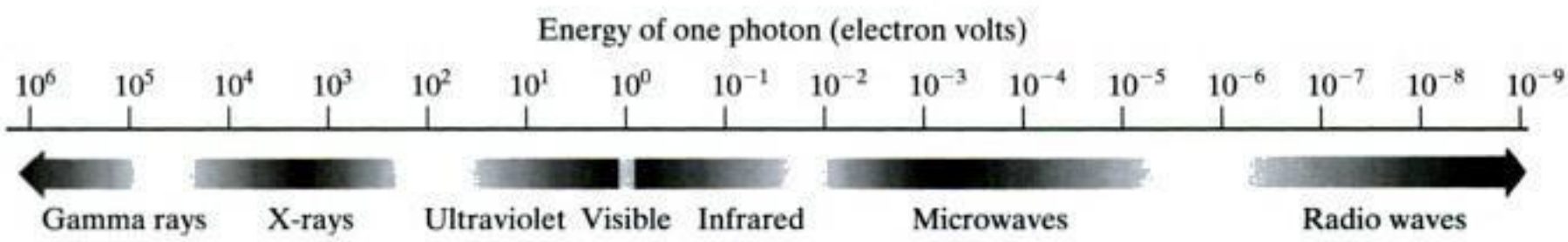
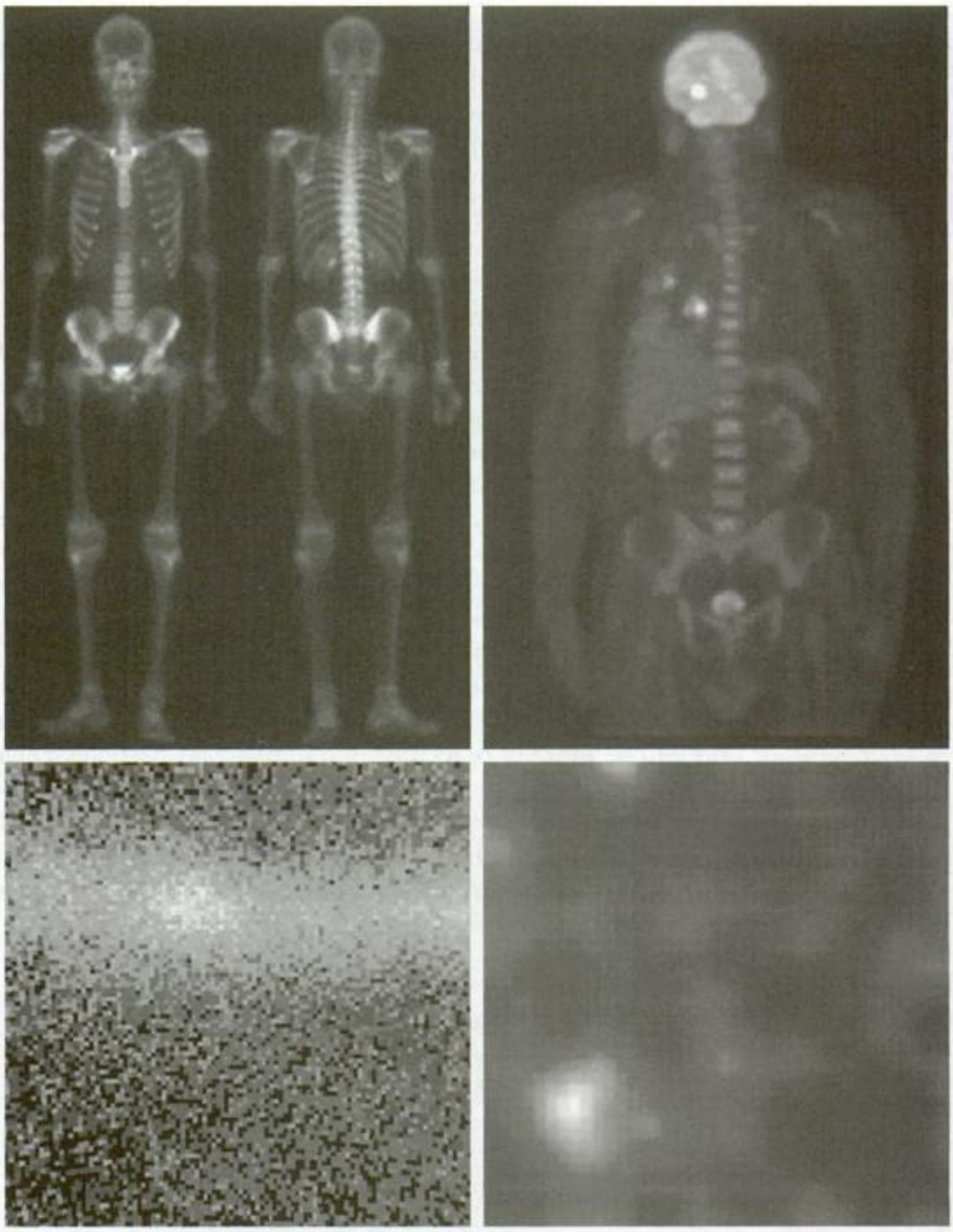


FIGURE 1.5 The electromagnetic spectrum arranged according to energy per photon.

The bands are shown shaded to convey the fact that bands of the EM spectrum are not distinct but rather transition smoothly from one to the other.

1.3.1 Gamma-Ray Imaging

Major uses of imaging based on gamma rays include nuclear medicine and astronomical observations. In nuclear medicine, the approach is to inject a patient with a radioactive isotope that emits gamma rays as it decays. Images are produced from the emissions collected by gamma ray detectors. Figure 1.6(a) shows an image of a complete bone scan obtained by using gamma-ray imaging. Images of this sort are used to locate sites of bone pathology, such as infections



a b
c d

FIGURE 1.6
Examples of gamma-ray imaging. (a) Bone scan. (b) PET image. (c) Cygnus Loop. (d) Gamma radiation (bright spot) from a reactor valve. (Images courtesy of (a) G.E. Medical Systems, (b) Dr. Michael E. Casey, CTI PET Systems, (c) NASA, (d) Professors Zhong He and David K. Wehe, University of Michigan.)

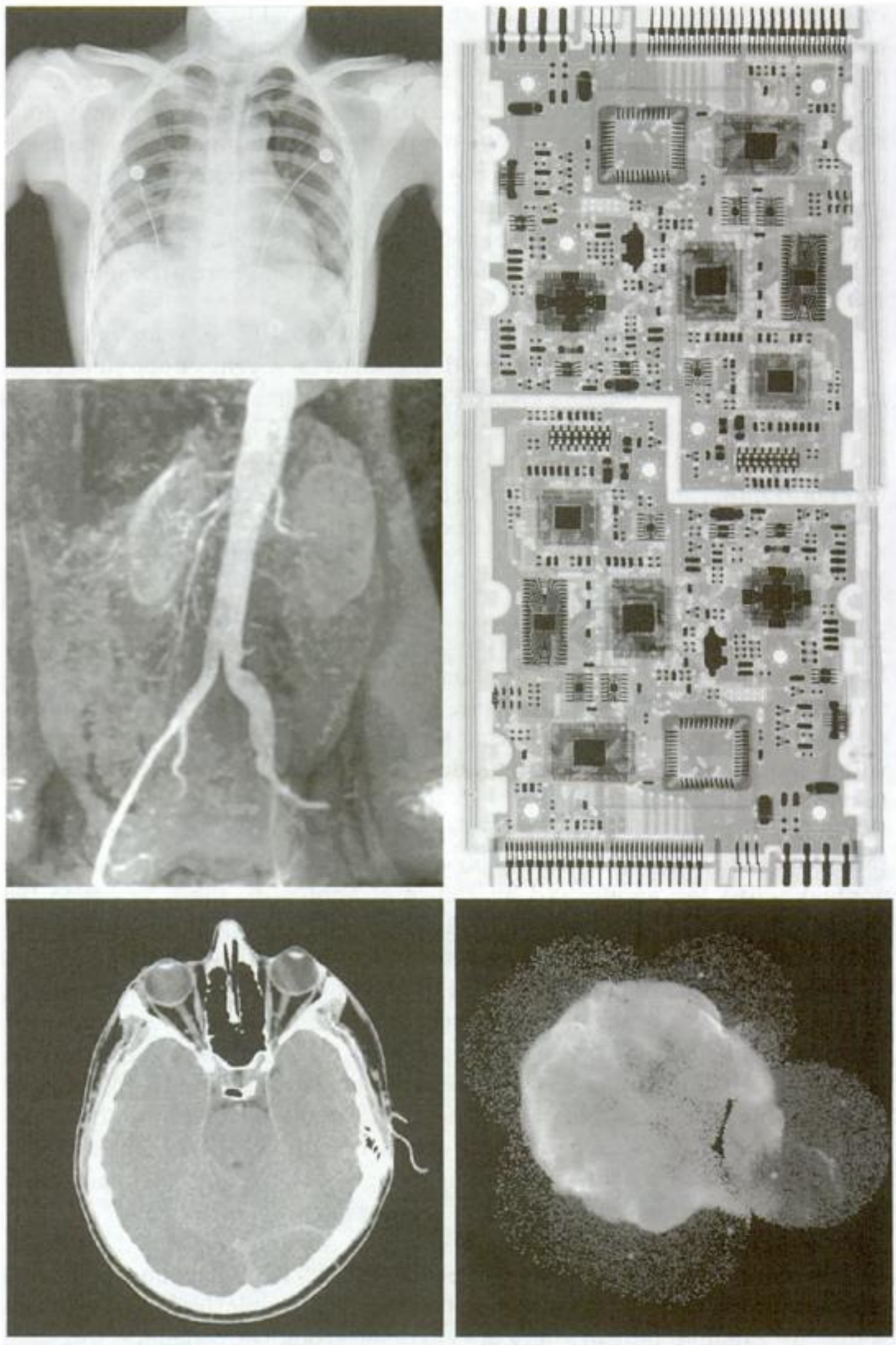
or tumors. Figure 1.6(b) shows another major modality of nuclear imaging called positron emission tomography (PET). The principle is the same as with X-ray tomography, mentioned briefly in Section 1.2. However, instead of using an external source of X-ray energy, the patient is given a radioactive isotope that emits positrons as it decays. When a positron meets an electron, both are annihilated and two gamma rays are given off. These are detected and a tomographic image is created using the basic principles of tomography. The image shown in Fig. 1.6(b) is one sample of a sequence that constitutes a 3-D rendition of the patient. This image shows a tumor in the brain and one in the lung, easily visible as small white masses.

A star in the constellation of Cygnus exploded about 15,000 years ago, generating a superheated stationary gas cloud (known as the Cygnus Loop) that glows in a spectacular array of colors. Figure 1.6(c) shows an image of the Cygnus Loop in the gamma-ray band. Unlike the two examples in Figs. 1.6(a) and (b), this image was obtained using the natural radiation of the object being imaged. Finally, Fig. 1.6(d) shows an image of gamma radiation from a valve in a nuclear reactor. An area of strong radiation is seen in the lower left side of the image.

1.3.2 X-Ray Imaging

X-rays are among the oldest sources of EM radiation used for imaging. The best known use of X-rays is medical diagnostics, but they also are used extensively in industry and other areas, like astronomy. X-rays for medical and industrial imaging are generated using an X-ray tube, which is a vacuum tube with a cathode and anode. The cathode is heated, causing free electrons to be released. These electrons flow at high speed to the positively charged anode. When the electrons strike a nucleus, energy is released in the form of X-ray radiation. The energy (penetrating power) of X-rays is controlled by a voltage applied across the anode, and by a current applied to the filament in the cathode. Figure 1.7(a) shows a familiar chest X-ray generated simply by placing the patient between an X-ray source and a film sensitive to X-ray energy. The intensity of the X-rays is modified by absorption as they pass through the patient, and the resulting energy falling on the film develops it, much in the same way that light develops photographic film. In digital radiography, digital images are obtained by one of two methods: (1) by digitizing X-ray films; or (2) by having the X-rays that pass through the patient fall directly onto devices (such as a phosphor screen) that convert X-rays to light. The light signal in turn is captured by a light-sensitive digitizing system. We discuss digitization in more detail in Chapters 2 and 4.

Angiography is another major application in an area called contrast-enhancement radiography. This procedure is used to obtain images (called *angiograms*) of blood vessels. A catheter (a small, flexible, hollow tube) is inserted, for example, into an artery or vein in the groin. The catheter is threaded into the blood vessel and guided to the area to be studied. When the catheter reaches the site under investigation, an X-ray contrast medium is injected through the tube. This enhances contrast of the blood vessels and enables the radiologist to see any irregularities or blockages. Figure 1.7(b) shows an example of an aortic angiogram. The catheter can be seen being inserted into the



a d
b e
c

FIGURE 1.7 Examples of X-ray imaging. (a) Chest X-ray. (b) Aortic angiogram. (c) Head CT. (d) Circuit boards. (e) Cygnus Loop. (Images courtesy of (a) and (c) Dr. David R. Pickens, Dept. of Radiology & Radiological Sciences, Vanderbilt University Medical Center; (b) Dr. Thomas R. Gest, Division of Anatomical Sciences, University of Michigan Medical School; (d) Mr. Joseph E. Pascente, Lixi, Inc.; and (e) NASA.)

large blood vessel on the lower left of the picture. Note the high contrast of the large vessel as the contrast medium flows up in the direction of the kidneys, which are also visible in the image. As discussed in Chapter 2, angiography is a major area of digital image processing, where image subtraction is used to enhance further the blood vessels being studied.

Another important use of X-rays in medical imaging is computerized axial tomography (CAT). Due to their resolution and 3-D capabilities, CAT scans revolutionized medicine from the moment they first became available in the early 1970s. As noted in Section 1.2, each CAT image is a "slice" taken perpendicularly through the patient. Numerous slices are generated as the patient is moved in a longitudinal direction. The ensemble of such images constitutes a 3-D rendition of the inside of the body, with the longitudinal resolution being proportional to the number of slice images taken. Figure 1.7(c) shows a typical head CAT slice image.

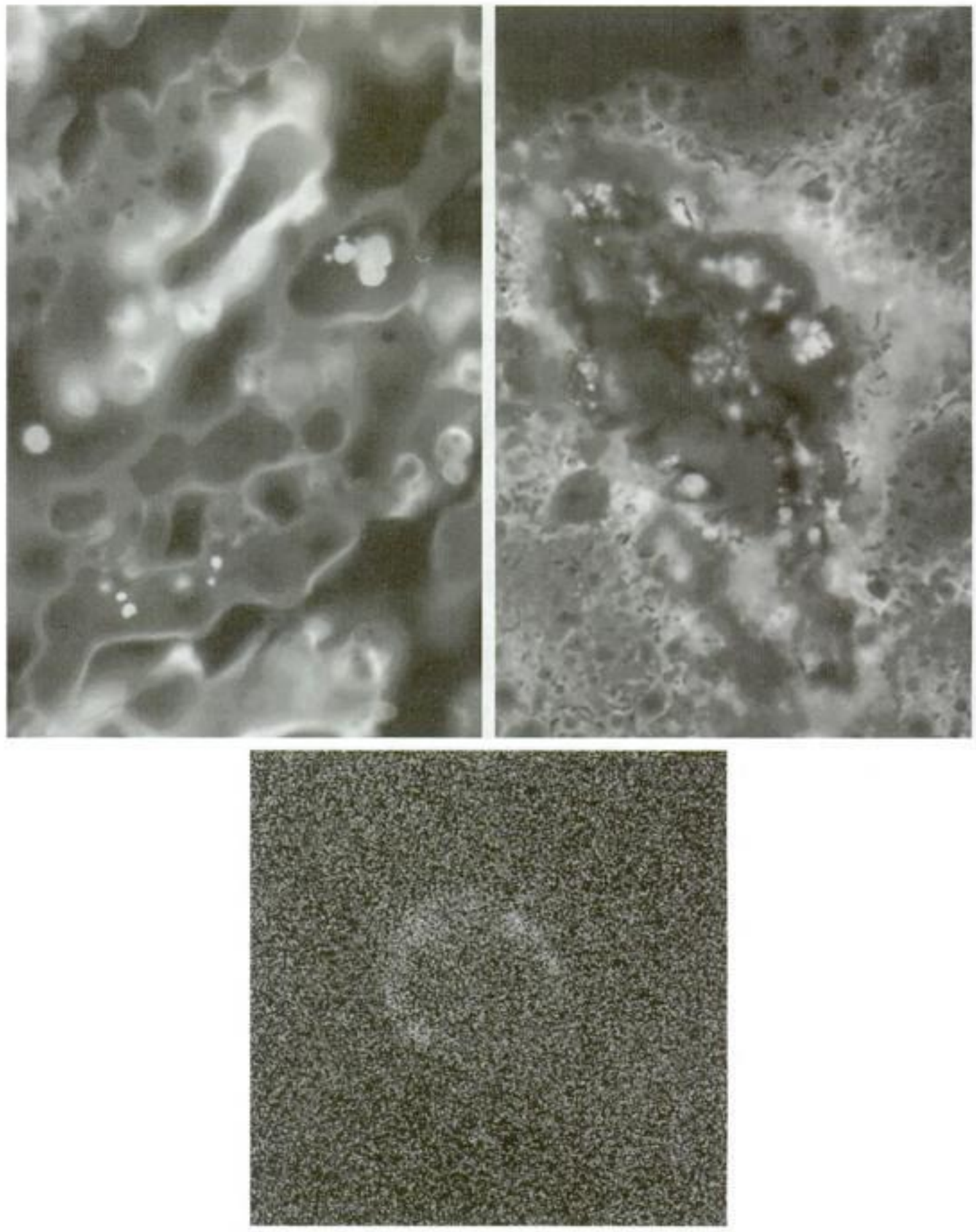
Techniques similar to the ones just discussed, but generally involving higher-energy X-rays, are applicable in industrial processes. Figure 1.7(d) shows an X-ray image of an electronic circuit board. Such images, representative of literally hundreds of industrial applications of X-rays, are used to examine circuit boards for flaws in manufacturing, such as missing components or broken traces. Industrial CAT scans are useful when the parts can be penetrated by X-rays, such as in plastic assemblies, and even large bodies, like solid-propellant rocket motors. Figure 1.7(e) shows an example of X-ray imaging in astronomy. This image is the Cygnus Loop of Fig. 1.6(c), but imaged this time in the X-ray band.

1.3.3 Imaging in the Ultraviolet Band

Applications of ultraviolet "light" are varied. They include lithography, industrial inspection, microscopy, lasers, biological imaging, and astronomical observations. We illustrate imaging in this band with examples from microscopy and astronomy.

Ultraviolet light is used in fluorescence microscopy, one of the fastest growing areas of microscopy. Fluorescence is a phenomenon discovered in the middle of the nineteenth century, when it was first observed that the mineral fluorspar fluoresces when ultraviolet light is directed upon it. The ultraviolet light itself is not visible, but when a photon of ultraviolet radiation collides with an electron in an atom of a fluorescent material, it elevates the electron to a higher energy level. Subsequently, the excited electron relaxes to a lower level and emits light in the form of a lower-energy photon in the visible (red) light region. The basic task of the fluorescence microscope is to use an excitation light to irradiate a prepared specimen and then to separate the much weaker radiating fluorescent light from the brighter excitation light. Thus, only the emission light reaches the eye or other detector. The resulting fluorescing areas shine against a dark background with sufficient contrast to permit detection. The darker the background of the nonfluorescing material, the more efficient the instrument.

Fluorescence microscopy is an excellent method for studying materials that can be made to fluoresce, either in their natural form (primary fluorescence) or when treated with chemicals capable of fluorescing (secondary fluorescence). Figures 1.8(a) and (b) show results typical of the capability of fluorescence microscopy. Figure 1.8(a) shows a fluorescence microscope image of normal corn, and Fig. 1.8(b) shows corn infected by "smut," a disease of cereals, corn,



a b
c

FIGURE 1.8
Examples of ultraviolet imaging. (a) Normal corn. (b) Smut corn. (c) Cygnus Loop. (Images courtesy of (a) and (b) Dr. Michael W. Davidson, Florida State University, (c) NASA.)

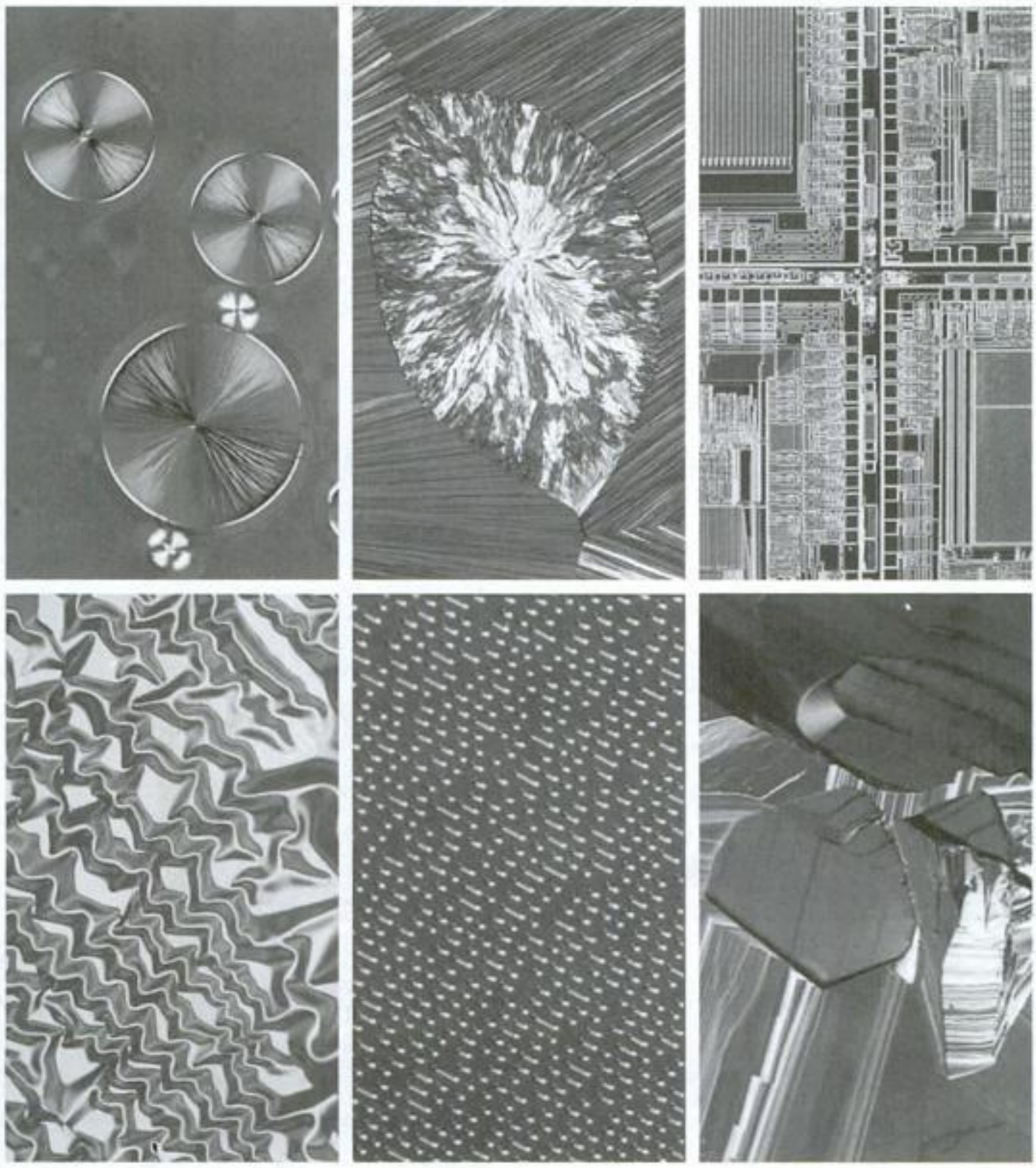
grasses, onions, and sorghum that can be caused by any of more than 700 species of parasitic fungi. Corn smut is particularly harmful because corn is one of the principal food sources in the world. As another illustration, Fig. 1.8(c) shows the Cygnus Loop imaged in the high-energy region of the ultraviolet band.

1.3.4 Imaging in the Visible and Infrared Bands

Considering that the visual band of the electromagnetic spectrum is the most familiar in all our activities, it is not surprising that imaging in this band outweighs by far all the others in terms of breadth of application. The infrared band often is used in conjunction with visual imaging, so we have grouped the

visible and infrared bands in this section for the purpose of illustration. We consider in the following discussion applications in light microscopy, astronomy, remote sensing, industry, and law enforcement.

Figure 1.9 shows several examples of images obtained with a light microscope. The examples range from pharmaceuticals and microinspection to materials characterization. Even in microscopy alone, the application areas are too numerous to detail here. It is not difficult to conceptualize the types of processes one might apply to these images, ranging from enhancement to measurements.



a b c
d e f

FIGURE 1.9 Examples of light microscopy images. (a) Taxol (anticancer agent), magnified 250×. (b) Cholesterol—40×. (c) Microprocessor—60×. (d) Nickel oxide thin film—600×. (e) Surface of audio CD—1750×. (f) Organic superconductor—450×. (Images courtesy of Dr. Michael W. Davidson, Florida State University.)

TABLE 1.1 Thematic bands in NASA's LANDSAT satellite.

Band No.	Name	Wavelength (μm)	Characteristics and Uses
1	Visible blue	0.45–0.52	Maximum water penetration
2	Visible green	0.52–0.60	Good for measuring plant vigor
3	Visible red	0.63–0.69	Vegetation discrimination
4	Near infrared	0.76–0.90	Biomass and shoreline mapping
5	Middle infrared	1.55–1.75	Moisture content of soil and vegetation
6	Thermal infrared	10.4–12.5	Soil moisture; thermal mapping
7	Middle infrared	2.08–2.35	Mineral mapping

Another major area of visual processing is remote sensing, which usually includes several bands in the visual and infrared regions of the spectrum. Table 1.1 shows the so-called *thematic bands* in NASA's LANDSAT satellite. The primary function of LANDSAT is to obtain and transmit images of the Earth from space for purposes of monitoring environmental conditions on the planet. The bands are expressed in terms of wavelength, with 1 μm being equal to 10^{-6} m (we discuss the wavelength regions of the electromagnetic spectrum in more detail in Chapter 2). Note the characteristics and uses of each band in Table 1.1.

In order to develop a basic appreciation for the power of this type of *multispectral* imaging, consider Fig. 1.10, which shows one image for each of

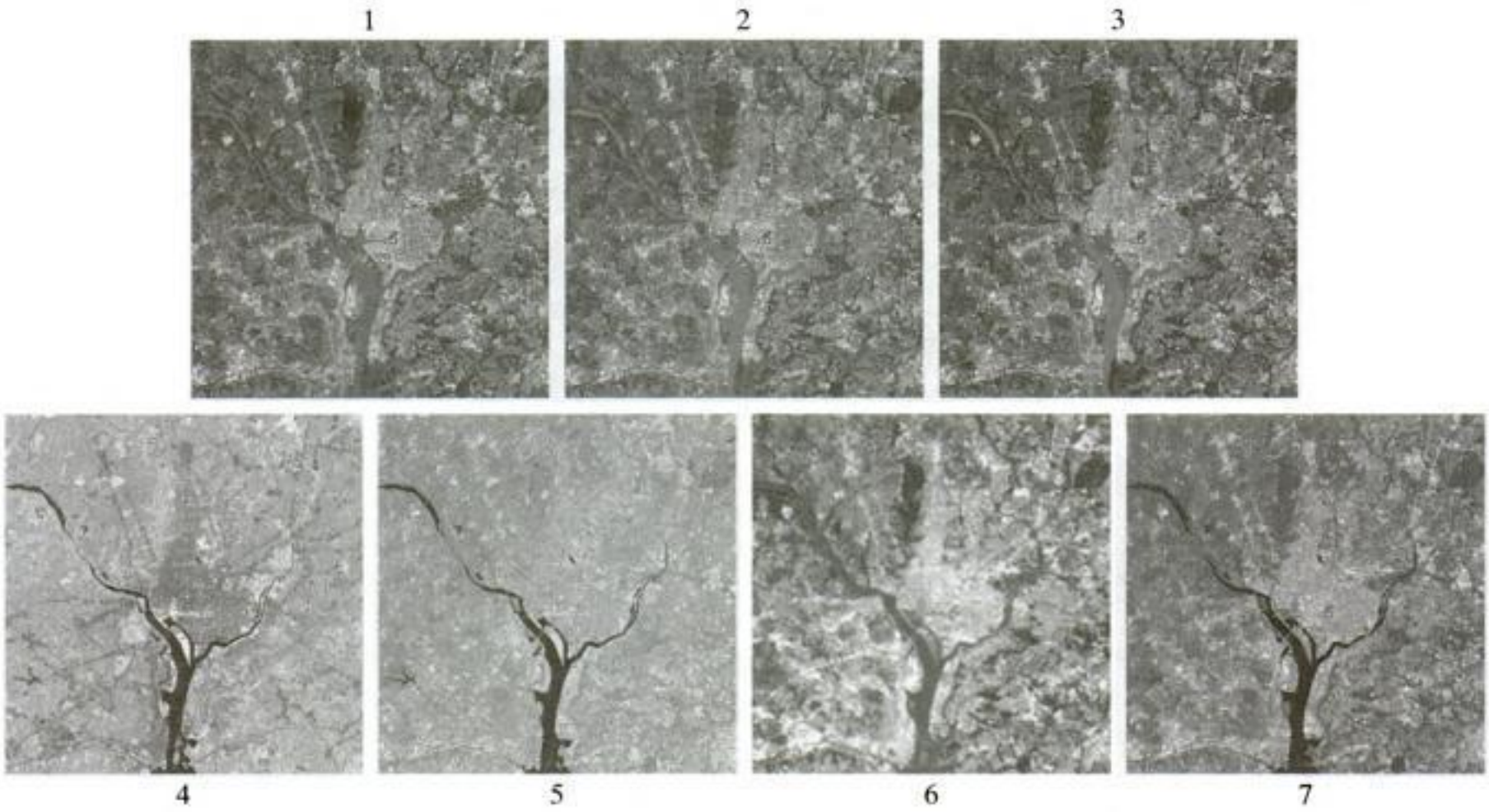


FIGURE 1.10 LANDSAT satellite images of the Washington, D.C. area. The numbers refer to the thematic bands in Table 1.1. (Images courtesy of NASA.)

FIGURE 1.11 Satellite image of Hurricane Katrina taken on August 29, 2005. (Courtesy of NOAA.)

the spectral bands in Table 1.1. The area imaged is Washington D.C., which includes features such as buildings, roads, vegetation, and a major river (the Potomac) going though the city. Images of population centers are used routinely (over time) to assess population growth and shift patterns, pollution, and other factors harmful to the environment. The differences between visual and infrared image features are quite noticeable in these images. Observe, for example, how well defined the river is from its surroundings in Bands 4 and 5.

Weather observation and prediction also are major applications of multispectral imaging from satellites. For example, Fig. 1.11 is an image of Hurricane Katrina one of the most devastating storms in recent memory in the Western Hemisphere. This image was taken by a National Oceanographic and Atmospheric Administration (NOAA) satellite using sensors in the visible and infrared bands. The eye of the hurricane is clearly visible in this image.

Figures 1.12 and 1.13 show an application of infrared imaging. These images are part of the *Nighttime Lights of the World* data set, which provides a global inventory of human settlements. The images were generated by the infrared imaging system mounted on a NOAA DMSP (Defense Meteorological Satellite Program) satellite. The infrared imaging system operates in the band 10.0 to 13.4 μm, and has the unique capability to observe faint sources of visible-near infrared emissions present on the Earth's surface, including cities, towns, villages, gas flares, and fires. Even without formal training in image processing, it is not difficult to imagine writing a computer program that would use these images to estimate the percent of total electrical energy used by various regions of the world.

A major area of imaging in the visual spectrum is in automated visual inspection of manufactured goods. Figure 1.14 shows some examples. Figure 1.14(a) is a controller board for a CD-ROM drive. A typical image processing task with products like this is to inspect them for missing parts (the black square on the top, right quadrant of the image is an example of a missing component).

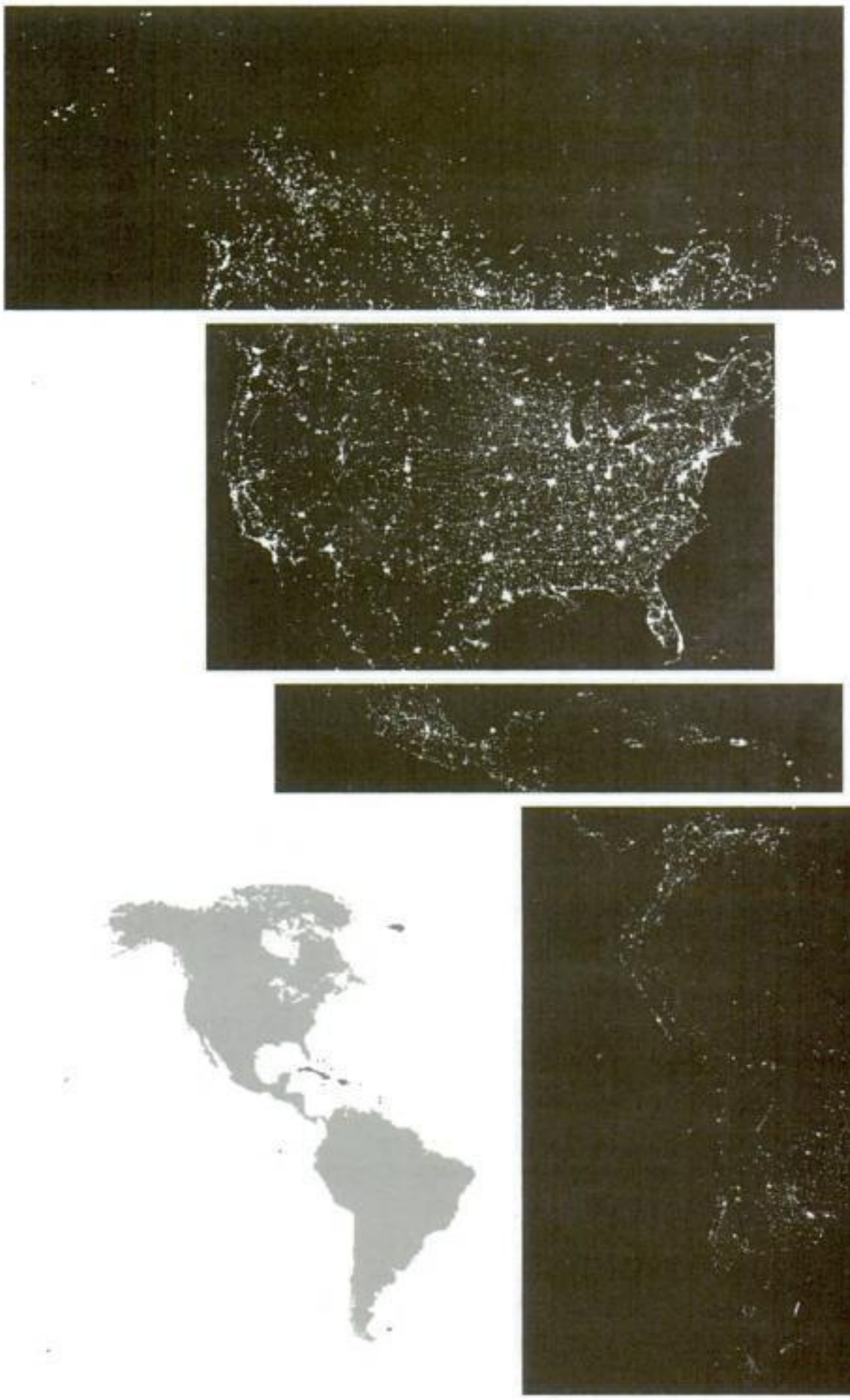


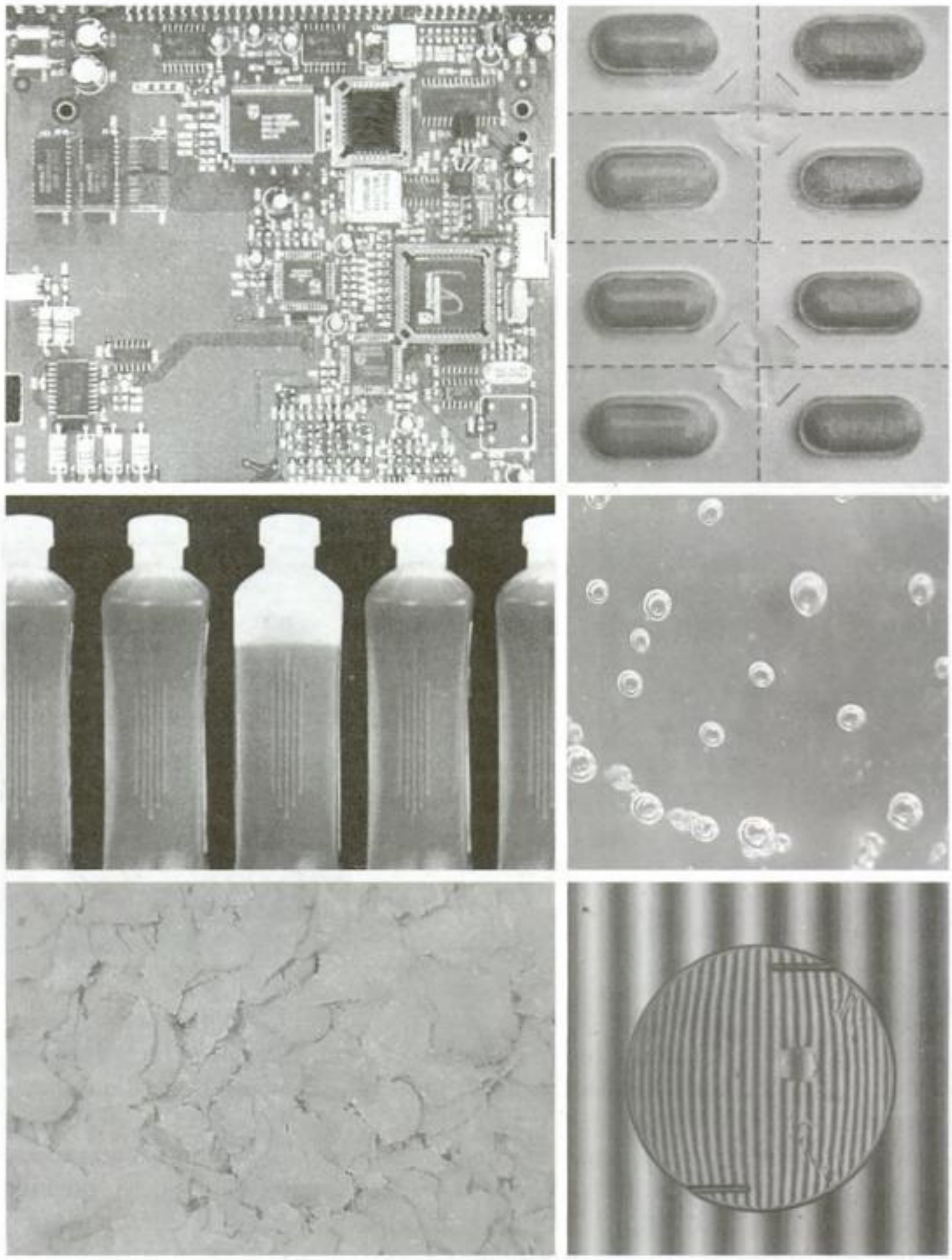
FIGURE 1.12
Infrared satellite images of the Americas. The small gray map is provided for reference. (Courtesy of NOAA.)

Figure 1.14(b) is an imaged pill container. The objective here is to have a machine look for missing pills. Figure 1.14(c) shows an application in which image processing is used to look for bottles that are not filled up to an acceptable level. Figure 1.14(d) shows a clear-plastic part with an unacceptable number of air pockets in it. Detecting anomalies like these is a major theme of industrial inspection that includes other products such as wood and cloth. Figure 1.14(e)

FIGURE 1.13
Infrared satellite images of the remaining populated part of the world. The small gray map is provided for reference. (Courtesy of NOAA.)

shows a batch of cereal during inspection for color and the presence of anomalies such as burned flakes. Finally, Fig. 1.14(f) shows an image of an intraocular implant (replacement lens for the human eye). A "structured light" illumination technique was used to highlight for easier detection flat lens deformations toward the center of the lens. The markings at 1 o'clock and 5 o'clock are tweezer damage. Most of the other small speckle detail is debris. The objective in this type of inspection is to find damaged or incorrectly manufactured implants automatically, prior to packaging.

As a final illustration of image processing in the visual spectrum, consider Fig. 1.15. Figure 1.15(a) shows a thumb print. Images of fingerprints are routinely processed by computer, either to enhance them or to find features that aid in the automated search of a database for potential matches. Figure 1.15(b) shows an image of paper currency. Applications of digital image processing in this area include automated counting and, in law enforcement, the reading of the serial number for the purpose of tracking and identifying bills. The two vehicle images shown in Figs. 1.15 (c) and (d) are examples of automated license plate reading. The light rectangles indicate the area in which the imaging system



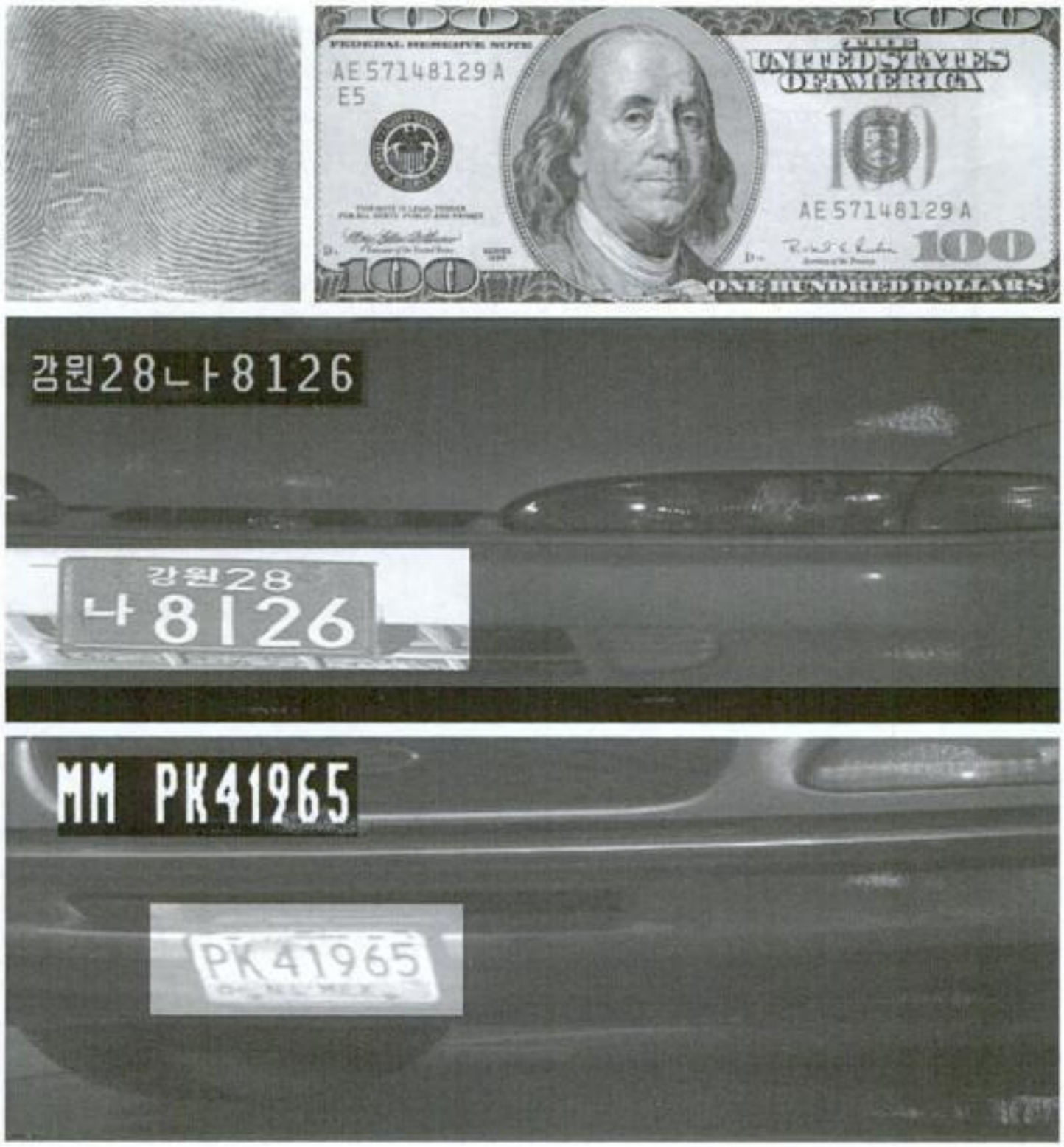
a b
c d
e f

FIGURE 1.14
Some examples of manufactured goods often checked using digital image processing. (a) A circuit board controller. (b) Packaged pills. (c) Bottles. (d) Air bubbles in a clear-plastic product. (e) Cereal. (f) Image of intraocular implant. (Fig. (f) courtesy of Mr. Pete Sites, Perceptics Corporation.)

detected the plate. The black rectangles show the results of automated reading of the plate content by the system. License plate and other applications of character recognition are used extensively for traffic monitoring and surveillance.

1.3.5 Imaging in the Microwave Band

The dominant application of imaging in the microwave band is radar. The unique feature of imaging radar is its ability to collect data over virtually any region at any time, regardless of weather or ambient lighting conditions. Some



a b
c
d

FIGURE 1.15
Some additional examples of imaging in the visual spectrum. (a) Thumb print. (b) Paper currency. (c) and (d) Automated license plate reading. (Figure (a) courtesy of the National Institute of Standards and Technology. Figures (c) and (d) courtesy of Dr. Juan Herrera, Perceptics Corporation.)

radar waves can penetrate clouds, and under certain conditions can also see through vegetation, ice, and dry sand. In many cases, radar is the only way to explore inaccessible regions of the Earth's surface. An imaging radar works like a flash camera in that it provides its own illumination (microwave pulses) to illuminate an area on the ground and take a snapshot image. Instead of a camera lens, a radar uses an antenna and digital computer processing to record its images. In a radar image, one can see only the microwave energy that was reflected back toward the radar antenna.

Figure 1.16 shows a spaceborne radar image covering a rugged mountainous area of southeast Tibet, about 90 km east of the city of Lhasa. In the lower right corner is a wide valley of the Lhasa River, which is populated by Tibetan farmers and yak herders and includes the village of Menba. Mountains in this area reach about 5800 m (19,000 ft) above sea level, while the valley floors lie about 4300 m (14,000 ft) above sea level. Note the clarity and detail of the image, unencumbered by clouds or other atmospheric conditions that normally interfere with images in the visual band.

FIGURE 1.16
Spaceborne radar image of mountains in southeast Tibet. (Courtesy of NASA.)

1.3.6 Imaging in the Radio Band

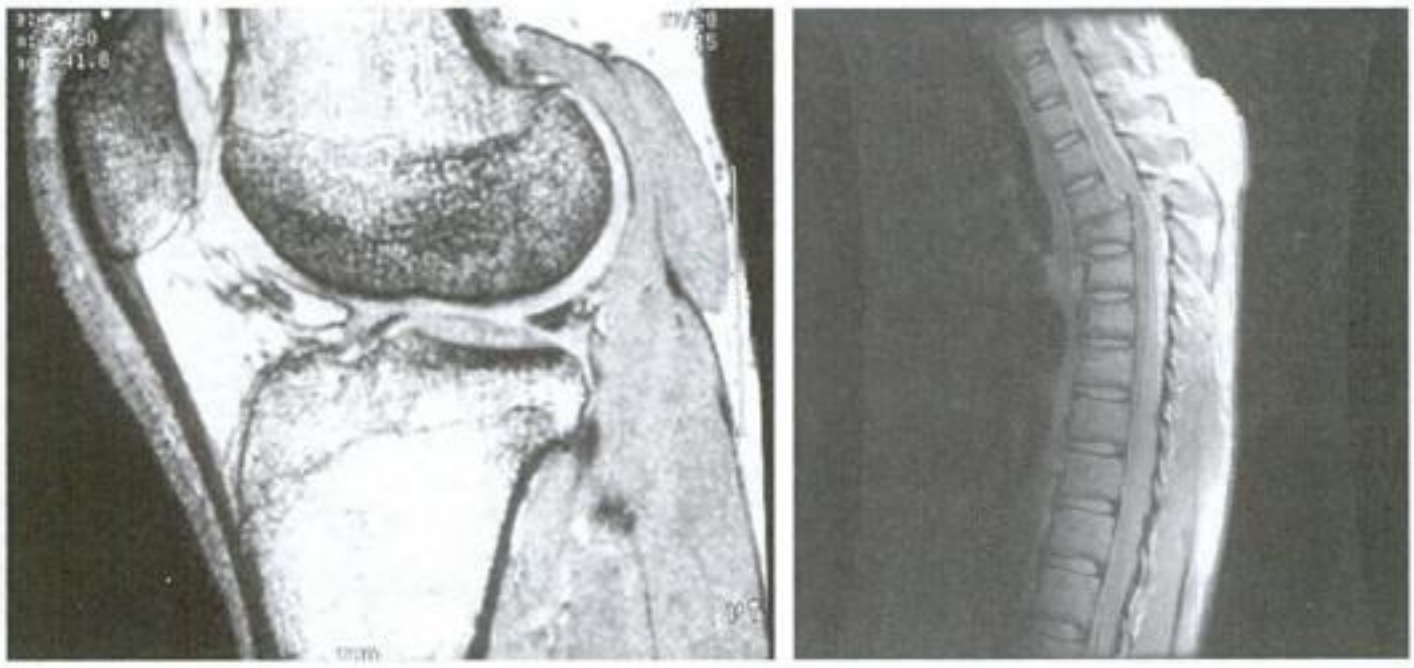
As in the case of imaging at the other end of the spectrum (gamma rays), the major applications of imaging in the radio band are in medicine and astronomy. In medicine, radio waves are used in magnetic resonance imaging (MRI). This technique places a patient in a powerful magnet and passes radio waves through his or her body in short pulses. Each pulse causes a responding pulse of radio waves to be emitted by the patient's tissues. The location from which these signals originate and their strength are determined by a computer, which produces a two-dimensional picture of a section of the patient. MRI can produce pictures in any plane. Figure 1.17 shows MRI images of a human knee and spine.

The last image to the right in Fig. 1.18 shows an image of the Crab Pulsar in the radio band. Also shown for an interesting comparison are images of the same region but taken in most of the bands discussed earlier. Note that each image gives a totally different "view" of the Pulsar.

1.3.7 Examples in which Other Imaging Modalities Are Used

Although imaging in the electromagnetic spectrum is dominant by far, there are a number of other imaging modalities that also are important. Specifically, we discuss in this section acoustic imaging, electron microscopy, and synthetic (computer-generated) imaging.

Imaging using "sound" finds application in geological exploration, industry, and medicine. Geological applications use sound in the low end of the sound spectrum (hundreds of Hz) while imaging in other areas use ultrasound (millions of Hz). The most important commercial applications of image processing in geology are in mineral and oil exploration. For image acquisition over land, one of the main approaches is to use a large truck and a large flat steel plate. The plate is pressed on the ground by the truck, and the truck is vibrated through a frequency spectrum up to 100 Hz. The strength and speed of the



a b

FIGURE 1.17 MRI images of a human (a) knee, and (b) spine. (Image (a) courtesy of Dr. Thomas R. Gest, Division of Anatomical Sciences, University of Michigan Medical School, and (b) courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

returning sound waves are determined by the composition of the Earth below the surface. These are analyzed by computer, and images are generated from the resulting analysis.

For marine acquisition, the energy source consists usually of two air guns towed behind a ship. Returning sound waves are detected by hydrophones placed in cables that are either towed behind the ship, laid on the bottom of the ocean, or hung from buoys (vertical cables). The two air guns are alternately pressurized to ~2000 psi and then set off. The constant motion of the ship provides a transversal direction of motion that, together with the returning sound waves, is used to generate a 3-D map of the composition of the Earth below the bottom of the ocean.

Figure 1.19 shows a cross-sectional image of a well-known 3-D model against which the performance of seismic imaging algorithms is tested. The arrow points to a hydrocarbon (oil and/or gas) trap. This target is brighter than the surrounding layers because the change in density in the target region is

FIGURE 1.18 Images of the Crab Pulsar (in the center of each image) covering the electromagnetic spectrum. (Courtesy of NASA.)

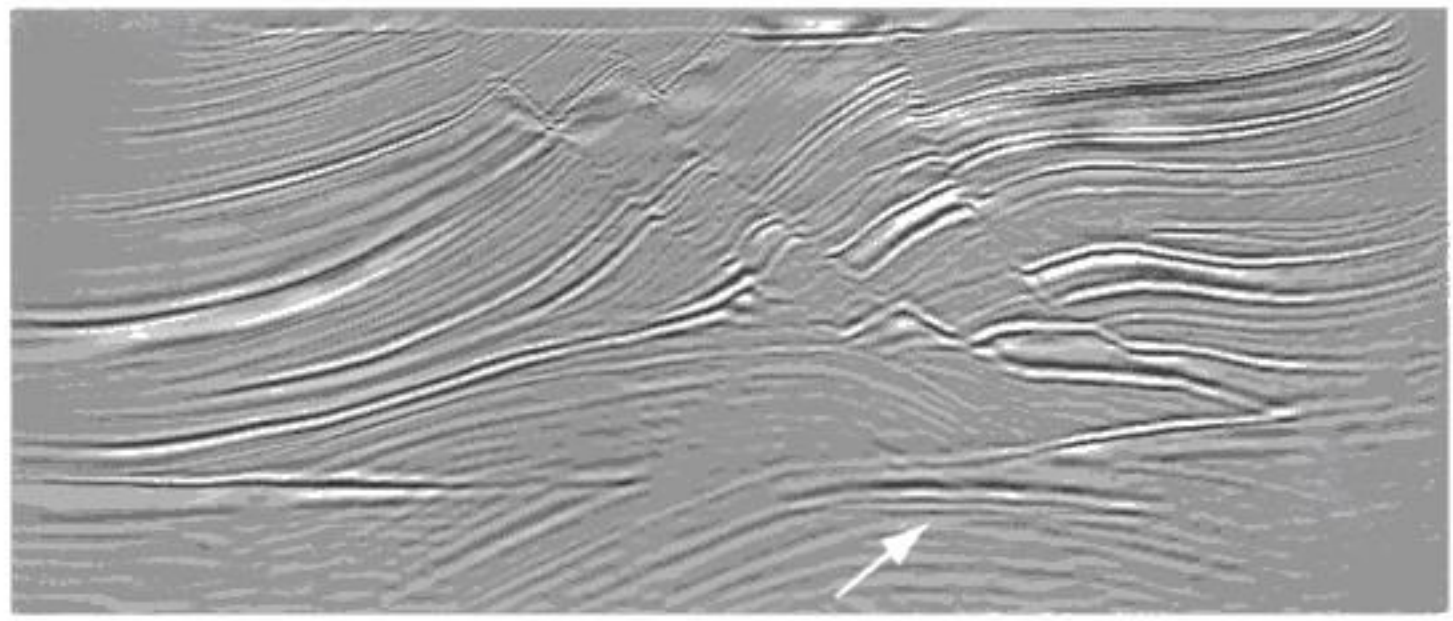


FIGURE 1.19
Cross-sectional image of a seismic model. The arrow points to a hydrocarbon (oil and/or gas) trap. (Courtesy of Dr. Curtis Ober, Sandia National Laboratories.)

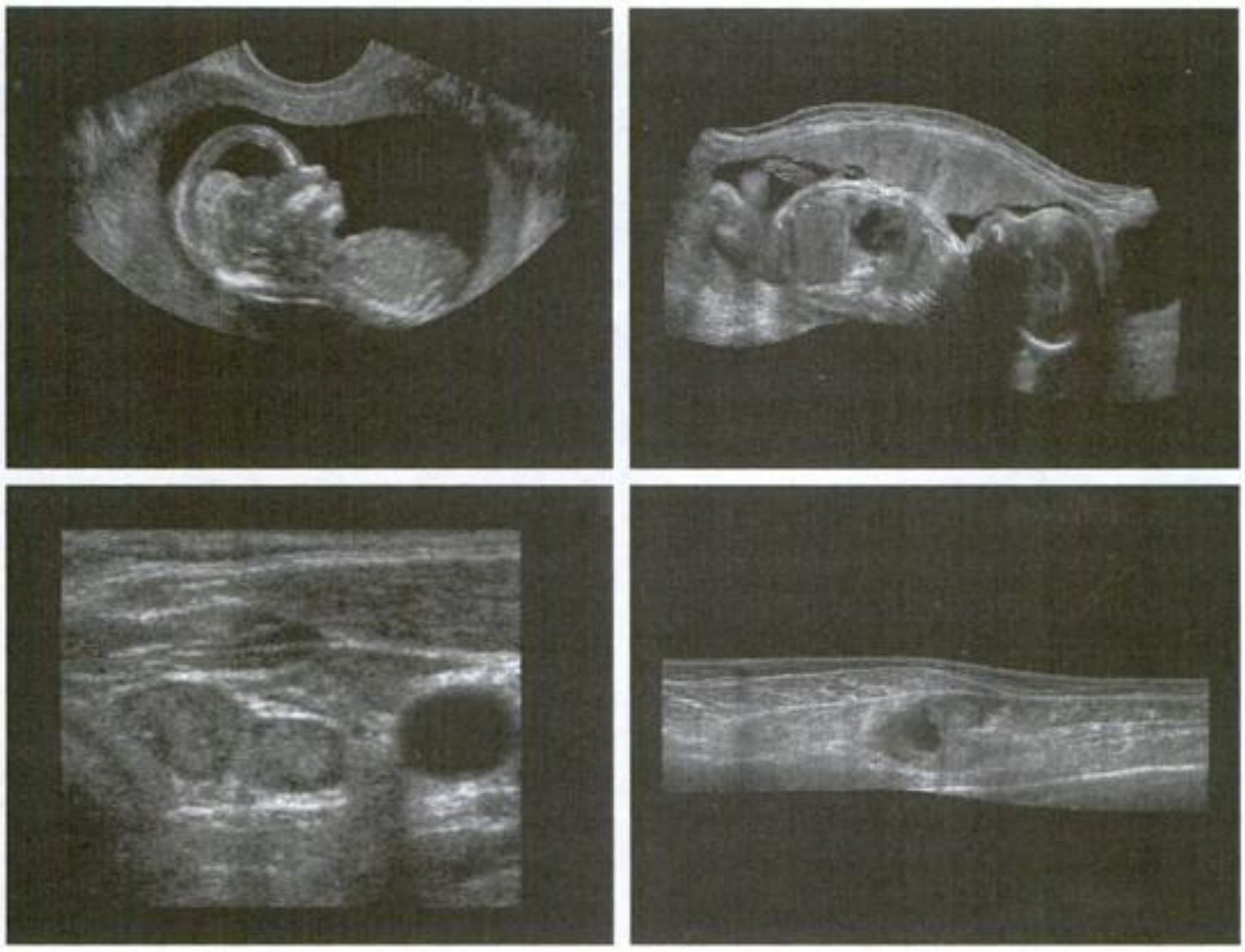
larger. Seismic interpreters look for these "bright spots" to find oil and gas. The layers above also are bright, but their brightness does not vary as strongly across the layers. Many seismic reconstruction algorithms have difficulty imaging this target because of the faults above it.

Although ultrasound imaging is used routinely in manufacturing, the best known applications of this technique are in medicine, especially in obstetrics, where unborn babies are imaged to determine the health of their development. A byproduct of this examination is determining the sex of the baby. Ultrasound images are generated using the following basic procedure:

1. The ultrasound system (a computer, ultrasound probe consisting of a source and receiver, and a display) transmits high-frequency (1 to 5 MHz) sound pulses into the body.
2. The sound waves travel into the body and hit a boundary between tissues (e.g., between fluid and soft tissue, soft tissue and bone). Some of the sound waves are reflected back to the probe, while some travel on further until they reach another boundary and get reflected.
3. The reflected waves are picked up by the probe and relayed to the computer.
4. The machine calculates the distance from the probe to the tissue or organ boundaries using the speed of sound in tissue (1540 m/s) and the time of each echo's return.
5. The system displays the distances and intensities of the echoes on the screen, forming a two-dimensional image.

In a typical ultrasound image, millions of pulses and echoes are sent and received each second. The probe can be moved along the surface of the body and angled to obtain various views. Figure 1.20 shows several examples.

We continue the discussion on imaging modalities with some examples of electron microscopy. Electron microscopes function as their optical counterparts, except that they use a focused beam of electrons instead of light to image a specimen. The operation of electron microscopes involves the following basic steps: A stream of electrons is produced by an electron source and accelerated toward the specimen using a positive electrical potential. This stream



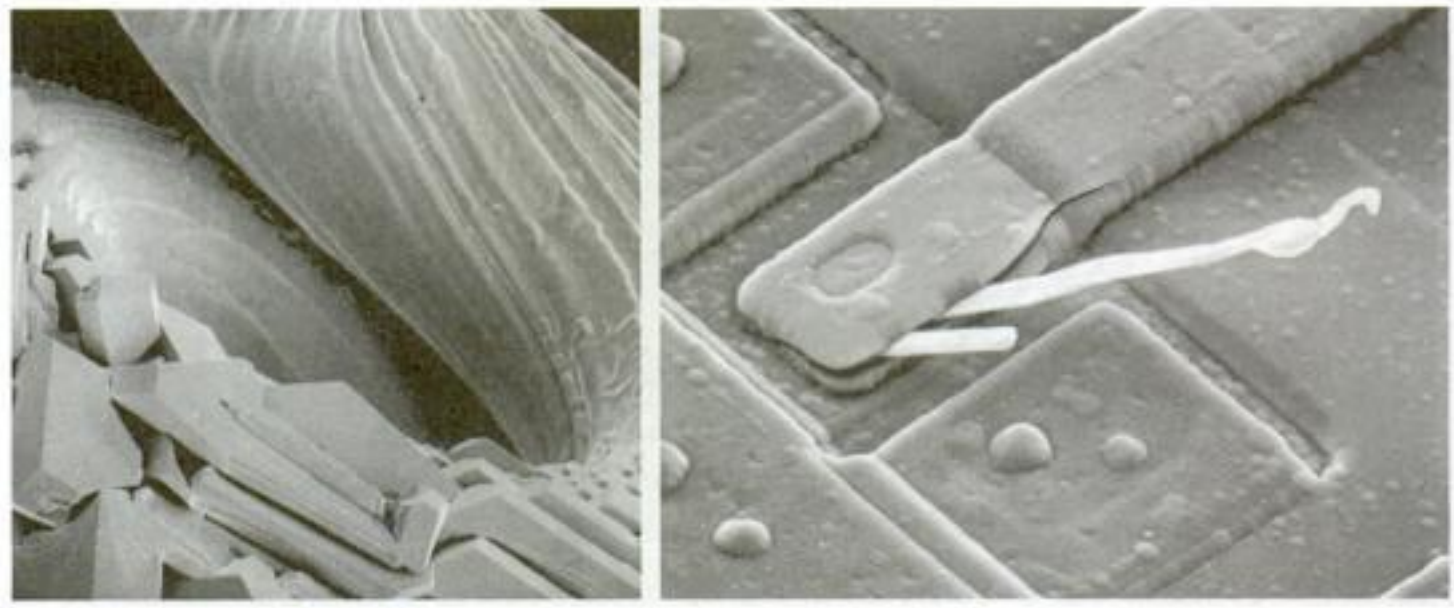
a b
c d

FIGURE 1.20 Examples of ultrasound imaging. (a) Baby. (b) Another view of baby. (c) Thyroids. (d) Muscle layers showing lesion. (Courtesy of Siemens Medical Systems, Inc., Ultrasound Group.)

is confined and focused using metal apertures and magnetic lenses into a thin, monochromatic beam. This beam is focused onto the sample using a magnetic lens. Interactions occur inside the irradiated sample, affecting the electron beam. These interactions and effects are detected and transformed into an image, much in the same way that light is reflected from, or absorbed by, objects in a scene. These basic steps are carried out in all electron microscopes.

A *transmission electron microscope* (TEM) works much like a slide projector. A projector shines (transmits) a beam of light through a slide; as the light passes through the slide, it is modulated by the contents of the slide. This transmitted beam is then projected onto the viewing screen, forming an enlarged image of the slide. TEMs work the same way, except that they shine a beam of electrons through a specimen (analogous to the slide). The fraction of the beam transmitted through the specimen is projected onto a phosphor screen. The interaction of the electrons with the phosphor produces light and, therefore, a viewable image. A *scanning electron microscope* (SEM), on the other hand, actually scans the electron beam and records the interaction of beam and sample at each location. This produces one dot on a phosphor screen. A complete image is formed by a raster scan of the beam through the sample, much like a TV camera. The electrons interact with a phosphor screen and produce light. SEMs are suitable for "bulky" samples, while TEMs require very thin samples.

Electron microscopes are capable of very high magnification. While light microscopy is limited to magnifications on the order 1000×, electron microscopes



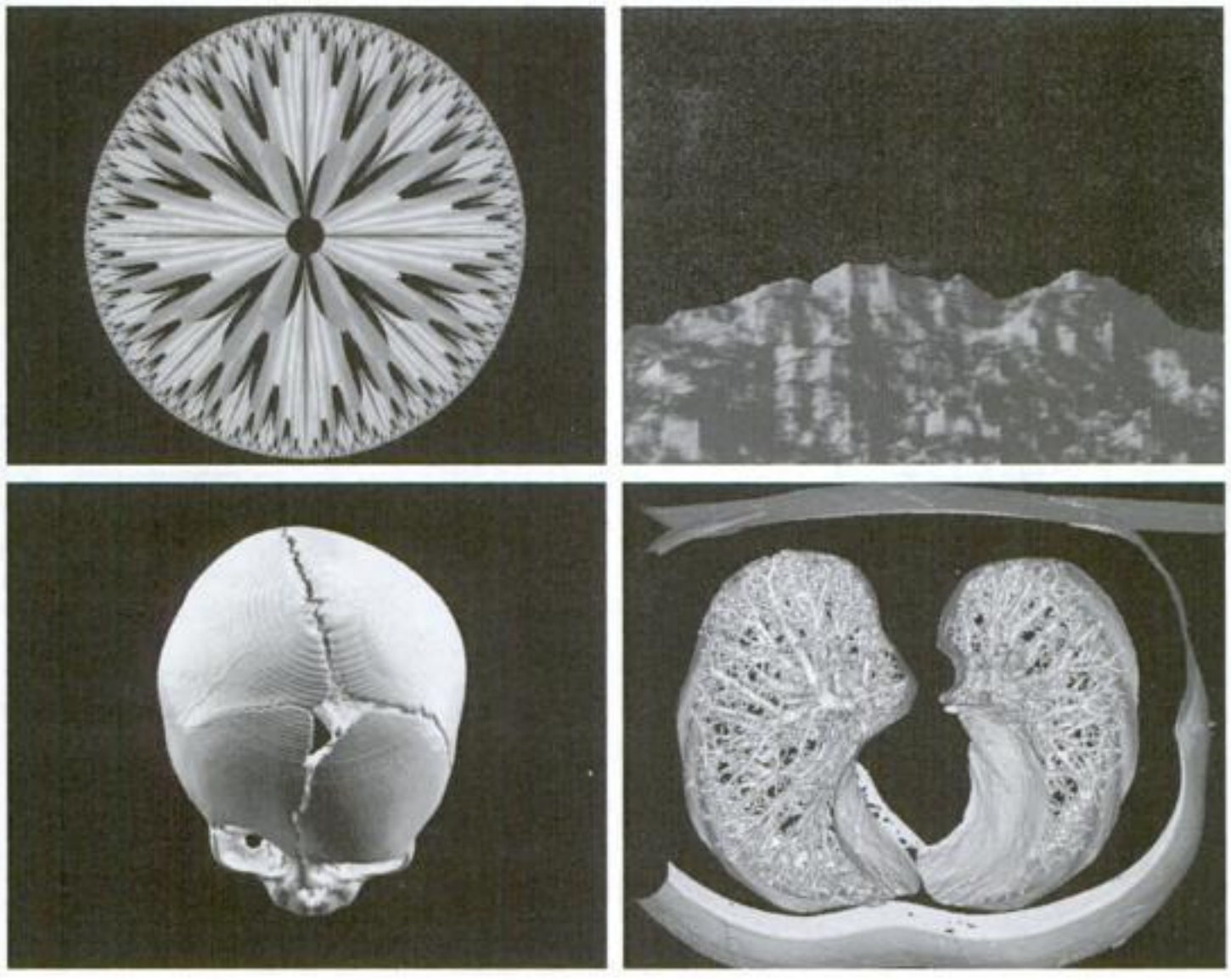
a b

FIGURE 1.21 (a) 250× SEM image of a tungsten filament following thermal failure (note the shattered pieces on the lower left). (b) 2500× SEM image of damaged integrated circuit. The white fibers are oxides resulting from thermal destruction. (Figure (a) courtesy of Mr. Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene; (b) courtesy of Dr. J. M. Hudak, McMaster University, Hamilton, Ontario, Canada.)

can achieve magnification of 10,000× or more. Figure 1.21 shows two SEM images of specimen failures due to thermal overload.

We conclude the discussion of imaging modalities by looking briefly at images that are not obtained from physical objects. Instead, they are generated by computer. *Fractals* are striking examples of computer-generated images (Lu [1997]). Basically, a fractal is nothing more than an iterative reproduction of a basic pattern according to some mathematical rules. For instance, *tiling* is one of the simplest ways to generate a fractal image. A square can be subdivided into four square subregions, each of which can be further subdivided into four smaller square regions, and so on. Depending on the complexity of the rules for filling each subsquare, some beautiful tile images can be generated using this method. Of course, the geometry can be arbitrary. For instance, the fractal image could be grown radially out of a center point. Figure 1.22(a) shows a fractal grown in this way. Figure 1.22(b) shows another fractal (a "moonscape") that provides an interesting analogy to the images of space used as illustrations in some of the preceding sections.

Fractal images tend toward artistic, mathematical formulations of "growth" of subimage elements according to a set of rules. They are useful sometimes as random textures. A more structured approach to image generation by computer lies in 3-D modeling. This is an area that provides an important intersection between image processing and computer graphics and is the basis for many 3-D visualization systems (e.g., flight simulators). Figures 1.22(c) and (d) show examples of computer-generated images. Since the original object is created in 3-D, images can be generated in any perspective from plane projections of the 3-D volume. Images of this type can be used for medical training and for a host of other applications, such as criminal forensics and special effects.



a b
c d

FIGURE 1.22
(a) and (b) Fractal images. (c) and (d) Images generated from 3-D computer models of the objects shown. (Figures (a) and (b) courtesy of Ms. Melissa D. Binde, Swarthmore College; (c) and (d) courtesy of NASA.)

1.4 Fundamental Steps in Digital Image Processing

It is helpful to divide the material covered in the following chapters into the two broad categories defined in Section 1.1: methods whose input and output are images, and methods whose inputs may be images but whose outputs are attributes extracted from those images. This organization is summarized in Fig. 1.23. The diagram does not imply that every process is applied to an image. Rather, the intention is to convey an idea of all the methodologies that can be applied to images for different purposes and possibly with different objectives. The discussion in this section may be viewed as a brief overview of the material in the remainder of the book.

Image acquisition is the first process in Fig. 1.23. The discussion in Section 1.3 gave some hints regarding the origin of digital images. This topic is considered in much more detail in Chapter 2, where we also introduce a number of basic digital image concepts that are used throughout the book. Note that acquisition could be as simple as being given an image that is already in digital form. Generally, the image acquisition stage involves preprocessing, such as scaling.

Image enhancement is the process of manipulating an image so that the result is more suitable than the original for a specific application. The word *specific* is important here, because it establishes at the outset that enhancement techniques are problem oriented. Thus, for example, a method that is quite useful for enhancing X-ray images may not be the best approach for enhancing satellite images taken in the infrared band of the electromagnetic spectrum.

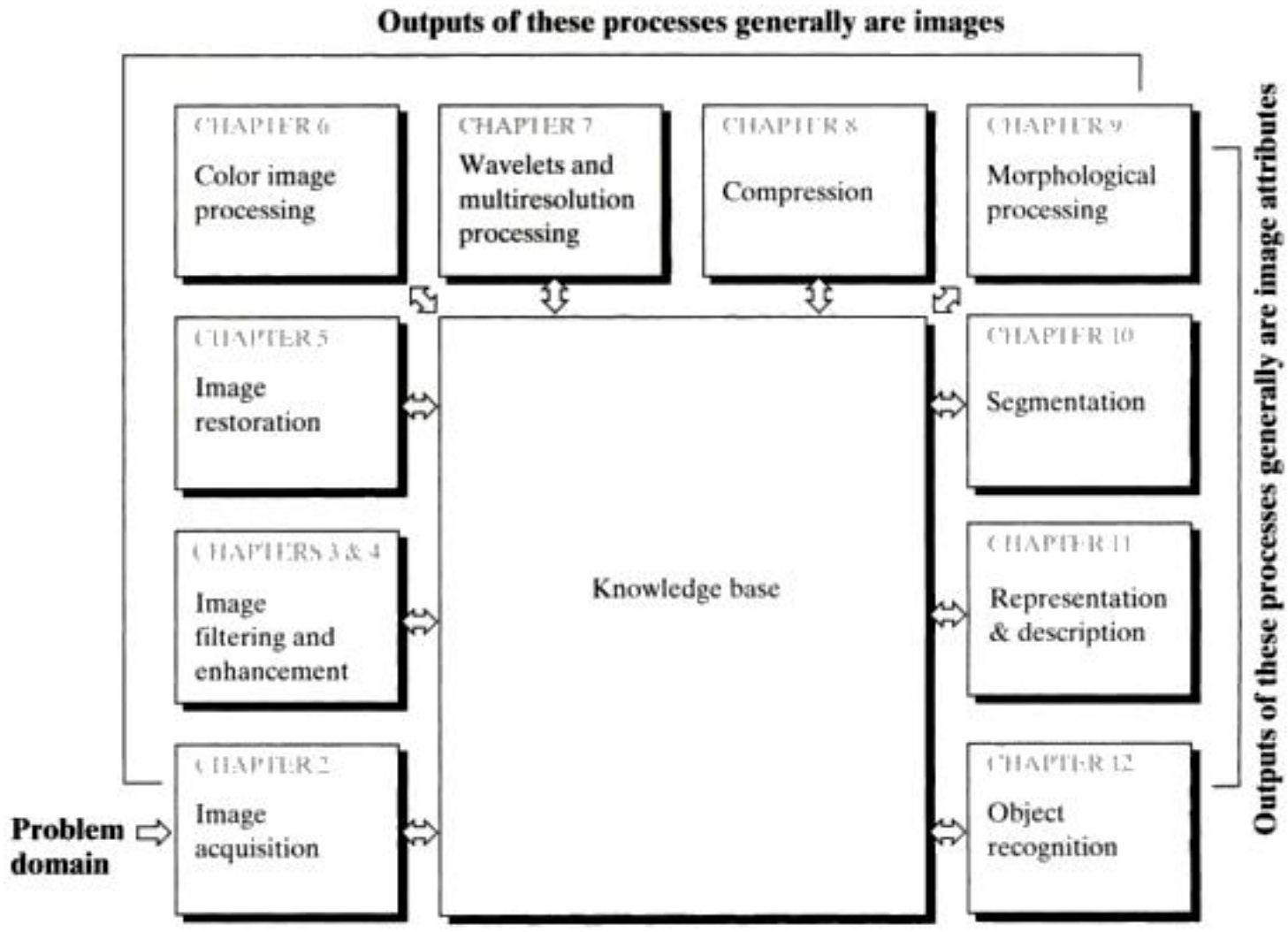


FIGURE 1.23 Fundamental steps in digital image processing. The chapter(s) indicated in the boxes is where the material described in the box is discussed.

There is no general "theory" of image enhancement. When an image is processed for visual interpretation, the viewer is the ultimate judge of how well a particular method works. Enhancement techniques are so varied, and use so many different image processing approaches, that it is difficult to assemble a meaningful body of techniques suitable for enhancement in one chapter without extensive background development. For this reason, and also because beginners in the field of image processing generally find enhancement applications visually appealing, interesting, and relatively simple to understand, we use image enhancement as examples when introducing new concepts in parts of Chapter 2 and in Chapters 3 and 4. The material in the latter two chapters span many of the methods used traditionally for image enhancement. Therefore, using examples from image enhancement to introduce new image processing methods developed in these early chapters not only saves having an extra chapter in the book dealing with image enhancement but, more importantly, is an effective approach for introducing newcomers to the details of processing techniques early in the book. However, as you will see in progressing through the rest of the book, the material developed in these chapters is applicable to a much broader class of problems than just image enhancement.

Image restoration is an area that also deals with improving the appearance of an image. However, unlike enhancement, which is subjective, image restoration is objective, in the sense that restoration techniques tend to be based on mathematical or probabilistic models of image degradation. Enhancement, on the other hand, is based on human subjective preferences regarding what constitutes a "good" enhancement result.

Color image processing is an area that has been gaining in importance because of the significant increase in the use of digital images over the Internet. Chapter 6 covers a number of fundamental concepts in color models and basic color processing in a digital domain. Color is used also in later chapters as the basis for extracting features of interest in an image.

Wavelets are the foundation for representing images in various degrees of resolution. In particular, this material is used in this book for image data compression and for pyramidal representation, in which images are subdivided successively into smaller regions.

Compression, as the name implies, deals with techniques for reducing the storage required to save an image, or the bandwidth required to transmit it. Although storage technology has improved significantly over the past decade, the same cannot be said for transmission capacity. This is true particularly in uses of the Internet, which are characterized by significant pictorial content. Image compression is familiar (perhaps inadvertently) to most users of computers in the form of image file extensions, such as the jpg file extension used in the JPEG (Joint Photographic Experts Group) image compression standard.

Morphological processing deals with tools for extracting image components that are useful in the representation and description of shape. The material in this chapter begins a transition from processes that output images to processes that output image attributes, as indicated in Section 1.1.

Segmentation procedures partition an image into its constituent parts or objects. In general, autonomous segmentation is one of the most difficult tasks in digital image processing. A rugged segmentation procedure brings the process a long way toward successful solution of imaging problems that require objects to be identified individually. On the other hand, weak or erratic segmentation algorithms almost always guarantee eventual failure. In general, the more accurate the segmentation, the more likely recognition is to succeed.

Representation and description almost always follow the output of a segmentation stage, which usually is raw pixel data, constituting either the boundary of a region (i.e., the set of pixels separating one image region from another) or all the points in the region itself. In either case, converting the data to a form suitable for computer processing is necessary. The first decision that must be made is whether the data should be represented as a boundary or as a complete region. Boundary representation is appropriate when the focus is on external shape characteristics, such as corners and inflections. Regional representation is appropriate when the focus is on internal properties, such as texture or skeletal shape. In some applications, these representations complement each other. Choosing a representation is only part of the solution for transforming raw data into a form suitable for subsequent computer processing. A method must also be specified for describing the data so that features of interest are highlighted. *Description*, also called *feature selection*, deals with extracting attributes that result in some quantitative information of interest or are basic for differentiating one class of objects from another.

Recognition is the process that assigns a label (e.g., "vehicle") to an object based on its descriptors. As detailed in Section 1.1, we conclude our coverage of

digital image processing with the development of methods for recognition of individual objects.

So far we have said nothing about the need for prior knowledge or about the interaction between the *knowledge base* and the processing modules in Fig. 1.23. Knowledge about a problem domain is coded into an image processing system in the form of a knowledge database. This knowledge may be as simple as detailing regions of an image where the information of interest is known to be located, thus limiting the search that has to be conducted in seeking that information. The knowledge base also can be quite complex, such as an interrelated list of all major possible defects in a materials inspection problem or an image database containing high-resolution satellite images of a region in connection with change-detection applications. In addition to guiding the operation of each processing module, the knowledge base also controls the interaction between modules. This distinction is made in Fig. 1.23 by the use of double-headed arrows between the processing modules and the knowledge base, as opposed to single-headed arrows linking the processing modules.

Although we do not discuss image display explicitly at this point, it is important to keep in mind that viewing the results of image processing can take place at the output of any stage in Fig. 1.23. We also note that not all image processing applications require the complexity of interactions implied by Fig. 1.23. In fact, not even all those modules are needed in many cases. For example, image enhancement for human visual interpretation seldom requires use of any of the other stages in Fig. 1.23. In general, however, as the complexity of an image processing task increases, so does the number of processes required to solve the problem.

1.5 Components of an Image Processing System

As recently as the mid-1980s, numerous models of image processing systems being sold throughout the world were rather substantial peripheral devices that attached to equally substantial host computers. Late in the 1980s and early in the 1990s, the market shifted to image processing hardware in the form of single boards designed to be compatible with industry standard buses and to fit into engineering workstation cabinets and personal computers. In addition to lowering costs, this market shift also served as a catalyst for a significant number of new companies specializing in the development of software written specifically for image processing.

Although large-scale image processing systems still are being sold for massive imaging applications, such as processing of satellite images, the trend continues toward miniaturizing and blending of general-purpose small computers with specialized image processing hardware. Figure 1.24 shows the basic components comprising a typical *general-purpose* system used for digital image processing. The function of each component is discussed in the following paragraphs, starting with image sensing.

With reference to *sensing*, two elements are required to acquire digital images. The first is a physical device that is sensitive to the energy radiated by the object we wish to image. The second, called a *digitizer*, is a device for converting

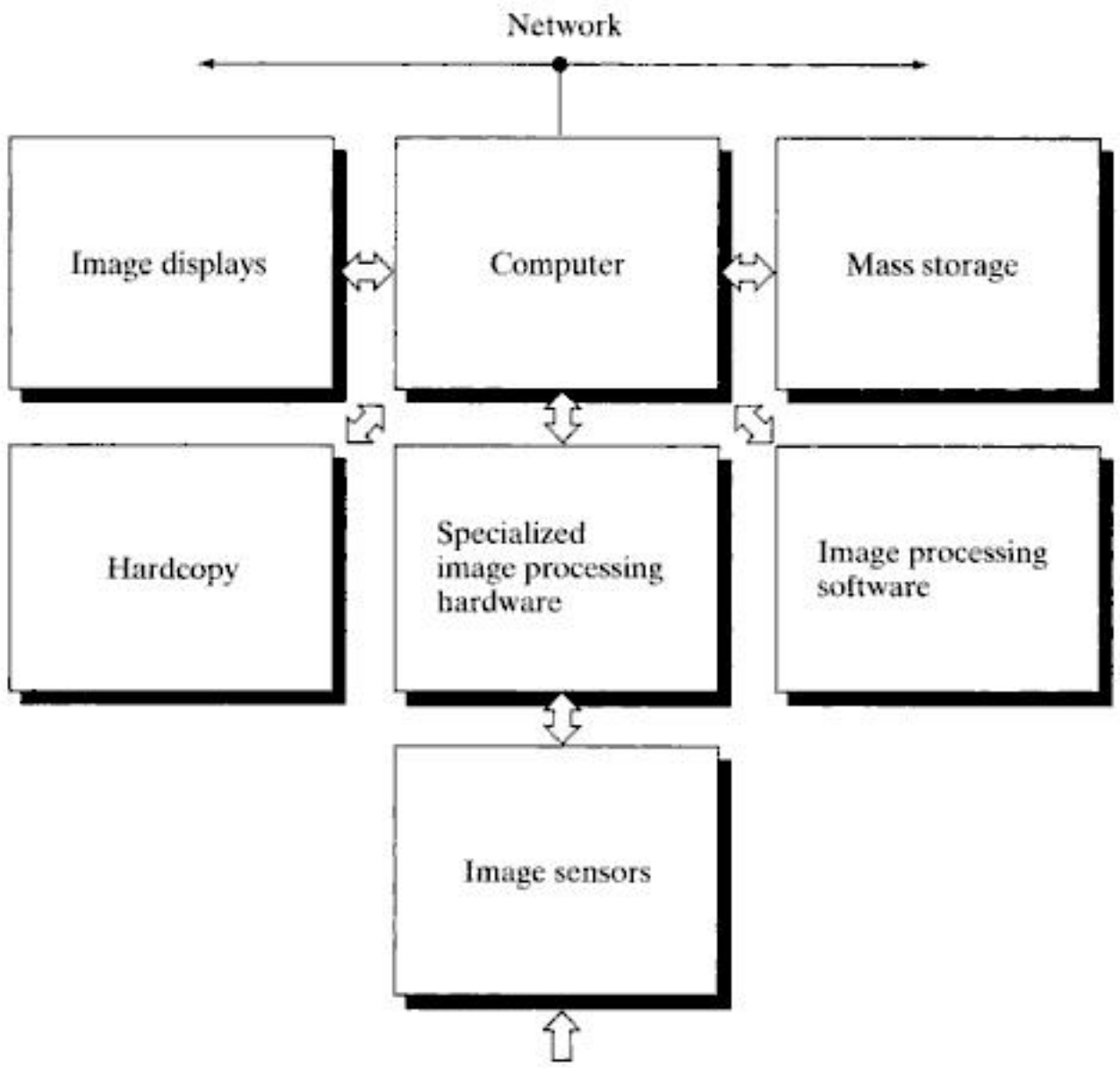


FIGURE 1.24 Components of a general-purpose image processing system.

the output of the physical sensing device into digital form. For instance, in a digital video camera, the sensors produce an electrical output proportional to light intensity. The digitizer converts these outputs to digital data. These topics are covered in Chapter 2.

Specialized image processing hardware usually consists of the digitizer just mentioned, plus hardware that performs other primitive operations, such as an arithmetic logic unit (ALU), that performs arithmetic and logical operations in parallel on entire images. One example of how an ALU is used is in averaging images as quickly as they are digitized, for the purpose of noise reduction. This type of hardware sometimes is called a *front-end subsystem*, and its most distinguishing characteristic is speed. In other words, this unit performs functions that require fast data throughputs (e.g., digitizing and averaging video images at 30 frames/s) that the typical main computer cannot handle.

The *computer* in an image processing system is a general-purpose computer and can range from a PC to a supercomputer. In dedicated applications, sometimes custom computers are used to achieve a required level of performance, but our interest here is on general-purpose image processing systems. In these systems, almost any well-equipped PC-type machine is suitable for off-line image processing tasks.

Software for image processing consists of specialized modules that perform specific tasks. A well-designed package also includes the capability for the user

to write code that, as a minimum, utilizes the specialized modules. More sophisticated software packages allow the integration of those modules and general-purpose software commands from at least one computer language.

Mass storage capability is a must in image processing applications. An image of size 1024 × 1024 pixels, in which the intensity of each pixel is an 8-bit quantity, requires one megabyte of storage space if the image is not compressed. When dealing with thousands, or even millions, of images, providing adequate storage in an image processing system can be a challenge. Digital storage for image processing applications falls into three principal categories: (1) short-term storage for use during processing, (2) on-line storage for relatively fast recall, and (3) archival storage, characterized by infrequent access. Storage is measured in bytes (eight bits), Kbytes (one thousand bytes), Mbytes (one million bytes), Gbytes (meaning giga, or one billion, bytes), and Tbytes (meaning tera, or one trillion, bytes).

One method of providing short-term storage is computer memory. Another is by specialized boards, called *frame buffers*, that store one or more images and can be accessed rapidly, usually at video rates (e.g., at 30 complete images per second). The latter method allows virtually instantaneous image *zoom*, as well as *scroll* (vertical shifts) and *pan* (horizontal shifts). Frame buffers usually are housed in the specialized image processing hardware unit in Fig. 1.24. On-line storage generally takes the form of magnetic disks or optical-media storage. The key factor characterizing on-line storage is frequent access to the stored data. Finally, archival storage is characterized by massive storage requirements but infrequent need for access. Magnetic tapes and optical disks housed in "jukeboxes" are the usual media for archival applications.

Image displays in use today are mainly color (preferably flat screen) TV monitors. Monitors are driven by the outputs of image and graphics display cards that are an integral part of the computer system. Seldom are there requirements for image display applications that cannot be met by display cards available commercially as part of the computer system. In some cases, it is necessary to have stereo displays, and these are implemented in the form of headgear containing two small displays embedded in goggles worn by the user.

Hardcopy devices for recording images include laser printers, film cameras, heat-sensitive devices, inkjet units, and digital units, such as optical and CD-ROM disks. Film provides the highest possible resolution, but paper is the obvious medium of choice for written material. For presentations, images are displayed on film transparencies or in a digital medium if image projection equipment is used. The latter approach is gaining acceptance as the standard for image presentations.

Networking is almost a default function in any computer system in use today. Because of the large amount of data inherent in image processing applications, the key consideration in image transmission is bandwidth. In dedicated networks, this typically is not a problem, but communications with remote sites via the Internet are not always as efficient. Fortunately, this situation is improving quickly as a result of optical fiber and other broadband technologies.

Summary

The main purpose of the material presented in this chapter is to provide a sense of perspective about the origins of digital image processing and, more important, about current and future areas of application of this technology. Although the coverage of these topics in this chapter was necessarily incomplete due to space limitations, it should have left you with a clear impression of the breadth and practical scope of digital image processing. As we proceed in the following chapters with the development of image processing theory and applications, numerous examples are provided to keep a clear focus on the utility and promise of these techniques. Upon concluding the study of the final chapter, a reader of this book will have arrived at a level of understanding that is the foundation for most of the work currently underway in this field.

References and Further Reading

References at the end of later chapters address specific topics discussed in those chapters, and are keyed to the Bibliography at the end of the book. However, in this chapter we follow a different format in order to summarize in one place a body of journals that publish material on image processing and related topics. We also provide a list of books from which the reader can readily develop a historical and current perspective of activities in this field. Thus, the reference material cited in this chapter is intended as a general-purpose, easily accessible guide to the published literature on image processing.

Major refereed journals that publish articles on image processing and related topics include: *IEEE Transactions on Image Processing; IEEE Transactions on Pattern Analysis and Machine Intelligence; Computer Vision, Graphics, and Image Processing* (prior to 1991); *Computer Vision and Image Understanding; IEEE Transactions on Systems, Man and Cybernetics; Artificial Intelligence; Pattern Recognition; Pattern Recognition Letters; Journal of the Optical Society of America* (prior to 1984); *Journal of the Optical Society of America—A: Optics, Image Science and Vision; Optical Engineering; Applied Optics—Information Processing; IEEE Transactions on Medical Imaging; Journal of Electronic Imaging; IEEE Transactions on Information Theory; IEEE Transactions on Communications; IEEE Transactions on Acoustics, Speech and Signal Processing; Proceedings of the IEEE;* and issues of the *IEEE Transactions on Computers* prior to 1980. Publications of the International Society for Optical Engineering (SPIE) also are of interest.

The following books, listed in reverse chronological order (with the number of books being biased toward more recent publications), contain material that complements our treatment of digital image processing. These books represent an easily accessible overview of the area for the past 30-plus years and were selected to provide a variety of treatments. They range from textbooks, which cover foundation material; to handbooks, which give an overview of techniques; and finally to edited books, which contain material representative of current research in the field.

Prince, J. L. and Links, J. M. [2006]. *Medical Imaging, Signals, and Systems*, Prentice Hall, Upper Saddle River, NJ.

Bezdek, J. C. et al. [2005]. *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Springer, New York.

Davies, E. R. [2005]. *Machine Vision: Theory, Algorithms, Practicalities*, Morgan Kaufmann, San Francisco, CA.

Rangayyan, R. M. [2005]. *Biomedical Image Analysis*, CRC Press, Boca Raton, FL.

Umbaugh, S. E. [2005]. *Computer Imaging: Digital Image Analysis and Processing*, CRC Press, Boca Raton, FL.

Gonzalez, R. C., Woods, R. E., and Eddins, S. L. [2004]. *Digital Image Processing Using MATLAB*, Prentice Hall, Upper Saddle River, NJ.

Snyder, W. E. and Qi, Hairong [2004]. *Machine Vision*, Cambridge University Press, New York.

Klette, R. and Rosenfeld, A. [2004]. *Digital Geometry—Geometric Methods for Digital Picture Analysis*, Morgan Kaufmann, San Francisco, CA.

Won, C. S. and Gray, R. M. [2004]. *Stochastic Image Processing*, Kluwer Academic/Plenum Publishers, New York.

Soille, P. [2003]. *Morphological Image Analysis: Principles and Applications*, 2nd ed., Springer-Verlag, New York.

Dougherty, E. R. and Lotufo, R. A. [2003]. *Hands-on Morphological Image Processing*, SPIE—The International Society for Optical Engineering, Bellingham, WA.

Gonzalez, R. C. and Woods, R. E. [2002]. *Digital Image Processing*, 2nd ed., Prentice Hall, Upper Saddle River, NJ.

Forsyth, D. F. and Ponce, J. [2002]. *Computer Vision—A Modern Approach*, Prentice Hall, Upper Saddle River, NJ.

Duda, R. O., Hart, P. E., and Stork, D. G. [2001]. *Pattern Classification*, 2nd ed., John Wiley & Sons, New York.

Pratt, W. K. [2001]. *Digital Image Processing*, 3rd ed., John Wiley & Sons, New York.

Ritter, G. X. and Wilson, J. N. [2001]. *Handbook of Computer Vision Algorithms in Image Algebra*, CRC Press, Boca Raton, FL.

Shapiro, L. G. and Stockman, G. C. [2001]. *Computer Vision*, Prentice Hall, Upper Saddle River, NJ.

Dougherty, E. R. (ed.) [2000]. *Random Processes for Image and Signal Processing*, IEEE Press, New York.

Etienne, E. K. and Nachtegael, M. (eds.). [2000]. *Fuzzy Techniques in Image Processing*, Springer-Verlag, New York.

Goutsias, J., Vincent, L., and Bloomberg, D. S. (eds.). [2000]. *Mathematical Morphology and Its Applications to Image and Signal Processing*, Kluwer Academic Publishers, Boston, MA.

Mallot, A. H. [2000]. *Computational Vision*, The MIT Press, Cambridge, MA.

Marchand-Maillet, S. and Sharaiha, Y. M. [2000]. *Binary Digital Image Processing: A Discrete Approach*, Academic Press, New York.

Mitra, S. K. and Sicuranza, G. L. (eds.) [2000]. *Nonlinear Image Processing*, Academic Press, New York.

Edelman, S. [1999]. *Representation and Recognition in Vision*, The MIT Press, Cambridge, MA.

Lillesand, T. M. and Kiefer, R. W. [1999]. *Remote Sensing and Image Interpretation*, John Wiley & Sons, New York.

Mather, P. M. [1999]. *Computer Processing of Remotely Sensed Images: An Introduction*, John Wiley & Sons, New York.

Petrou, M. and Bosdogianni, P. [1999]. *Image Processing: The Fundamentals*, John Wiley & Sons, UK.

Russ, J. C. [1999]. *The Image Processing Handbook*, 3rd ed., CRC Press, Boca Raton, FL.

Smirnov, A. [1999]. *Processing of Multidimensional Signals*, Springer-Verlag, New York.

Sonka, M., Hlavac, V., and Boyle, R. [1999]. *Image Processing, Analysis, and Computer Vision*, PWS Publishing, New York.

Haskell, B. G. and Netravali, A. N. [1997]. *Digital Pictures: Representation, Compression, and Standards*, Perseus Publishing, New York.

Jahne, B. [1997]. *Digital Image Processing: Concepts, Algorithms, and Scientific Applications*, Springer-Verlag, New York.

Castleman, K. R. [1996]. *Digital Image Processing*, 2nd ed., Prentice Hall, Upper Saddle River, NJ.

Geladi, P. and Grahn, H. [1996]. *Multivariate Image Analysis*, John Wiley & Sons, New York.

Bracewell, R. N. [1995]. *Two-Dimensional Imaging*, Prentice Hall, Upper Saddle River, NJ.

Sid-Ahmed, M. A. [1995]. *Image Processing: Theory, Algorithms, and Architectures*, McGraw-Hill, New York.

Jain, R., Rangachar, K., and Schunk, B. [1995]. *Computer Vision*, McGraw-Hill, New York.

Mitiche, A. [1994]. *Computational Analysis of Visual Motion*, Perseus Publishing, New York.

Baxes, G. A. [1994]. *Digital Image Processing: Principles and Applications*, John Wiley & Sons, New York.

Gonzalez, R. C. and Woods, R. E. [1992]. *Digital Image Processing*, Addison-Wesley, Reading, MA.

Haralick, R. M. and Shapiro, L. G. [1992]. *Computer and Robot Vision*, vols. 1 & 2, Addison-Wesley, Reading, MA.

Pratt, W. K. [1991] *Digital Image Processing*, 2nd ed., Wiley-Interscience, New York.

Lim, J. S. [1990]. *Two-Dimensional Signal and Image Processing*, Prentice Hall, Upper Saddle River, NJ.

Jain, A. K. [1989]. *Fundamentals of Digital Image Processing*, Prentice Hall, Upper Saddle River, NJ.

Schalkoff, R. J. [1989]. *Digital Image Processing and Computer Vision*, John Wiley & Sons, New York.

Giardina, C. R. and Dougherty, E. R. [1988]. *Morphological Methods in Image and Signal Processing*, Prentice Hall, Upper Saddle River, NJ.

Levine, M. D. [1985]. *Vision in Man and Machine*, McGraw-Hill, New York.

Serra, J. [1982]. *Image Analysis and Mathematical Morphology*, Academic Press, New York.

Ballard, D. H. and Brown, C. M. [1982]. *Computer Vision*, Prentice Hall, Upper Saddle River, NJ.

Fu, K. S. [1982]. *Syntactic Pattern Recognition and Applications*, Prentice Hall, Upper Saddle River, NJ.

Nevatia, R. [1982]. *Machine Perception*, Prentice Hall, Upper Saddle River, NJ.

Pavlidis, T. [1982]. *Algorithms for Graphics and Image Processing*, Computer Science Press, Rockville, MD.

Rosenfeld, A. and Kak, A. C. [1982]. *Digital Picture Processing*, 2nd ed., vols. 1 & 2, Academic Press, New York.

Hall, E. L. [1979]. *Computer Image Processing and Recognition*, Academic Press, New York.

Gonzalez, R. C. and Thomason, M. G. [1978]. *Syntactic Pattern Recognition: An Introduction*, Addison-Wesley, Reading, MA.

Andrews, H. C. and Hunt, B. R. [1977]. *Digital Image Restoration*, Prentice Hall, Upper Saddle River, NJ.

Pavlidis, T. [1977]. *Structural Pattern Recognition*, Springer-Verlag, New York.

Tou, J. T. and Gonzalez, R. C. [1974]. *Pattern Recognition Principles*, Addison-Wesley, Reading, MA.

Andrews, H. C. [1970]. *Computer Techniques in Image Processing*, Academic Press, New York.

2 *Digital Image Fundamentals*

Those who wish to succeed must **ask the right** preliminary questions.

Aristotle

Preview

The purpose of this chapter is to introduce you to a number of basic concepts in digital image processing that are used throughout the book. Section 2.1 summarizes the mechanics of the human visual system, including image formation in the eye and its capabilities for brightness adaptation and discrimination. Section 2.2 discusses light, other components of the electromagnetic spectrum, and their imaging characteristics. Section 2.3 discusses imaging sensors and how they are used to generate digital images. Section 2.4 introduces the concepts of uniform image sampling and intensity quantization. Additional topics discussed in that section include digital image representation, the effects of varying the number of samples and intensity levels in an image, the concepts of spatial and intensity resolution, and the principles of image interpolation. Section 2.5 deals with a variety of basic relationships between pixels. Finally, Section 2.6 is an introduction to the principal mathematical tools we use throughout the book. A second objective of that section is to help you begin developing a "feel" for how these tools are used in a variety of basic image processing tasks. The scope of these tools and their application are expanded as needed in the remainder of the book.

2.1 Elements of Visual Perception

Although the field of digital image processing is built on a foundation of mathematical and probabilistic formulations, human intuition and analysis play a central role in the choice of one technique versus another, and this choice often is made based on subjective, visual judgments. Hence, developing a basic understanding of human visual perception as a first step in our journey through this book is appropriate. Given the complexity and breadth of this topic, we can only aspire to cover the most rudimentary aspects of human vision. In particular, our interest is in the mechanics and parameters related to how images are formed and perceived by humans. We are interested in learning the physical limitations of human vision in terms of factors that also are used in our work with digital images. Thus, factors such as how human and electronic imaging devices compare in terms of resolution and ability to adapt to changes in illumination are not only interesting, they also are important from a practical point of view.

2.1.1 Structure of the Human Eye

Figure 2.1 shows a simplified horizontal cross section of the human eye. The eye is nearly a sphere, with an average diameter of approximately 20 mm. Three membranes enclose the eye: the *cornea* and *sclera* outer cover; the *choroid*; and the *retina*. The cornea is a tough, transparent tissue that covers

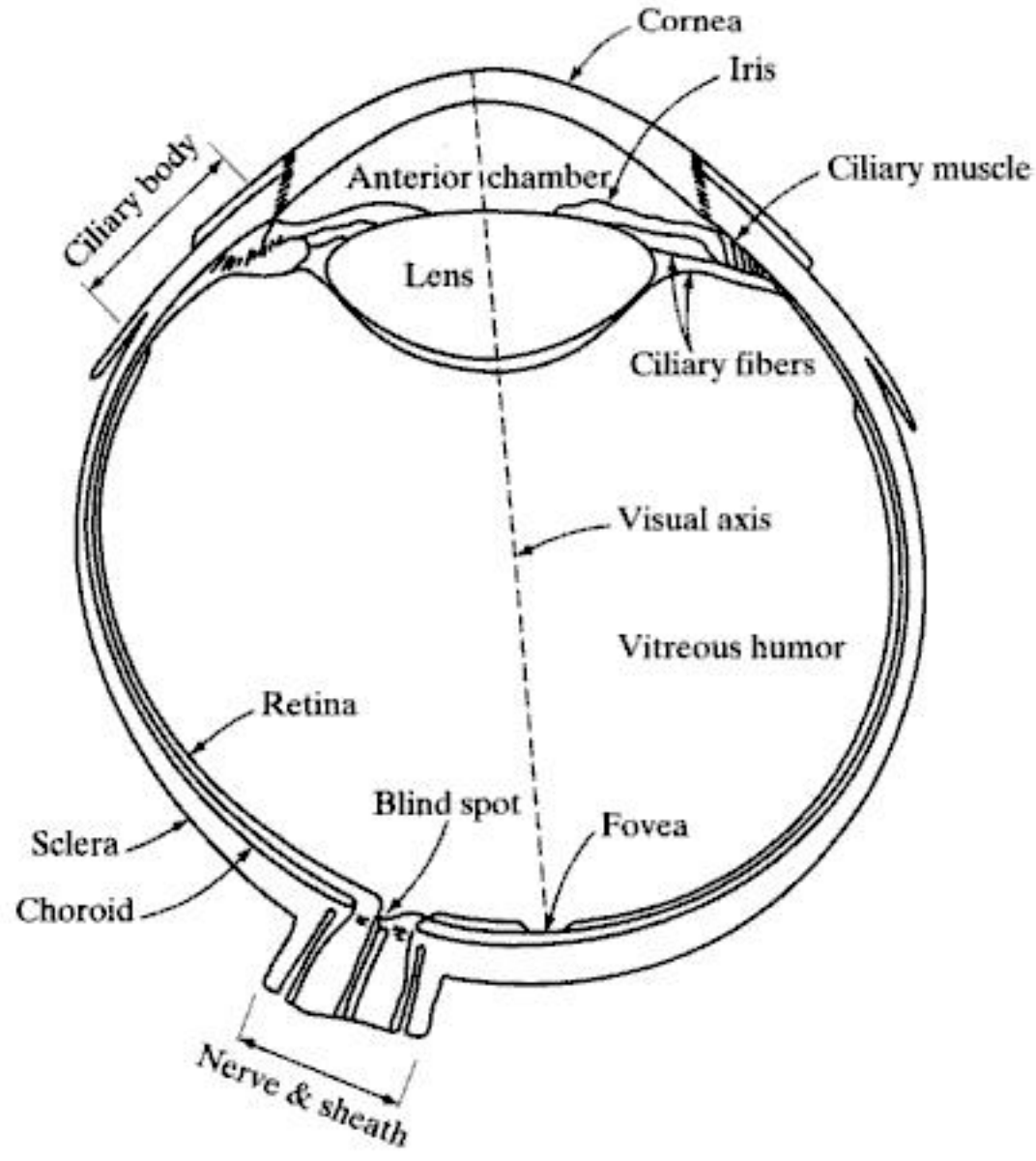


FIGURE 2.1 Simplified diagram of a cross section of the human eye.

the anterior surface of the eye. Continuous with the cornea, the sclera is an opaque membrane that encloses the remainder of the optic globe.

The choroid lies directly below the sclera. This membrane contains a network of blood vessels that serve as the major source of nutrition to the eye. Even superficial injury to the choroid, often not deemed serious, can lead to severe eye damage as a result of inflammation that restricts blood flow. The choroid coat is heavily pigmented and hence helps to reduce the amount of extraneous light entering the eye and the backscatter within the optic globe. At its anterior extreme, the choroid is divided into the *ciliary body* and the *iris*. The latter contracts or expands to control the amount of light that enters the eye. The central opening of the iris (the pupil) varies in diameter from approximately 2 to 8 mm. The front of the iris contains the visible pigment of the eye, whereas the back contains a black pigment.

The *lens* is made up of concentric layers of fibrous cells and is suspended by fibers that attach to the ciliary body. It contains 60 to 70% water, about 6% fat, and more protein than any other tissue in the eye. The lens is colored by a slightly yellow pigmentation that increases with age. In extreme cases, excessive clouding of the lens, caused by the affliction commonly referred to as *cataracts*, can lead to poor color discrimination and loss of clear vision. The lens absorbs approximately 8% of the visible light spectrum, with relatively higher absorption at shorter wavelengths. Both infrared and ultraviolet light are absorbed appreciably by proteins within the lens structure and, in excessive amounts, can damage the eye.

The innermost membrane of the eye is the *retina*, which lines the inside of the wall's entire posterior portion. When the eye is properly focused, light from an object outside the eye is imaged on the retina. Pattern vision is afforded by the distribution of discrete light receptors over the surface of the retina. There are two classes of receptors: *cones* and *rods*. The cones in each eye number between 6 and 7 million. They are located primarily in the central portion of the retina, called the *fovea*, and are highly sensitive to color. Humans can resolve fine details with these cones largely because each one is connected to its own nerve end. Muscles controlling the eye rotate the eyeball until the image of an object of interest falls on the fovea. Cone vision is called *photopic* or bright-light vision.

The number of rods is much larger: Some 75 to 150 million are distributed over the retinal surface. The larger area of distribution and the fact that several rods are connected to a single nerve end reduce the amount of detail discernible by these receptors. Rods serve to give a general, overall picture of the field of view. They are not involved in color vision and are sensitive to low levels of illumination. For example, objects that appear brightly colored in daylight when seen by moonlight appear as colorless forms because only the rods are stimulated. This phenomenon is known as *scotopic* or dim-light vision.

Figure 2.2 shows the density of rods and cones for a cross section of the right eye passing through the region of emergence of the optic nerve from the eye. The absence of receptors in this area results in the so-called *blind spot* (see Fig. 2.1). Except for this region, the distribution of receptors is radially symmetric about the fovea. Receptor density is measured in degrees from the

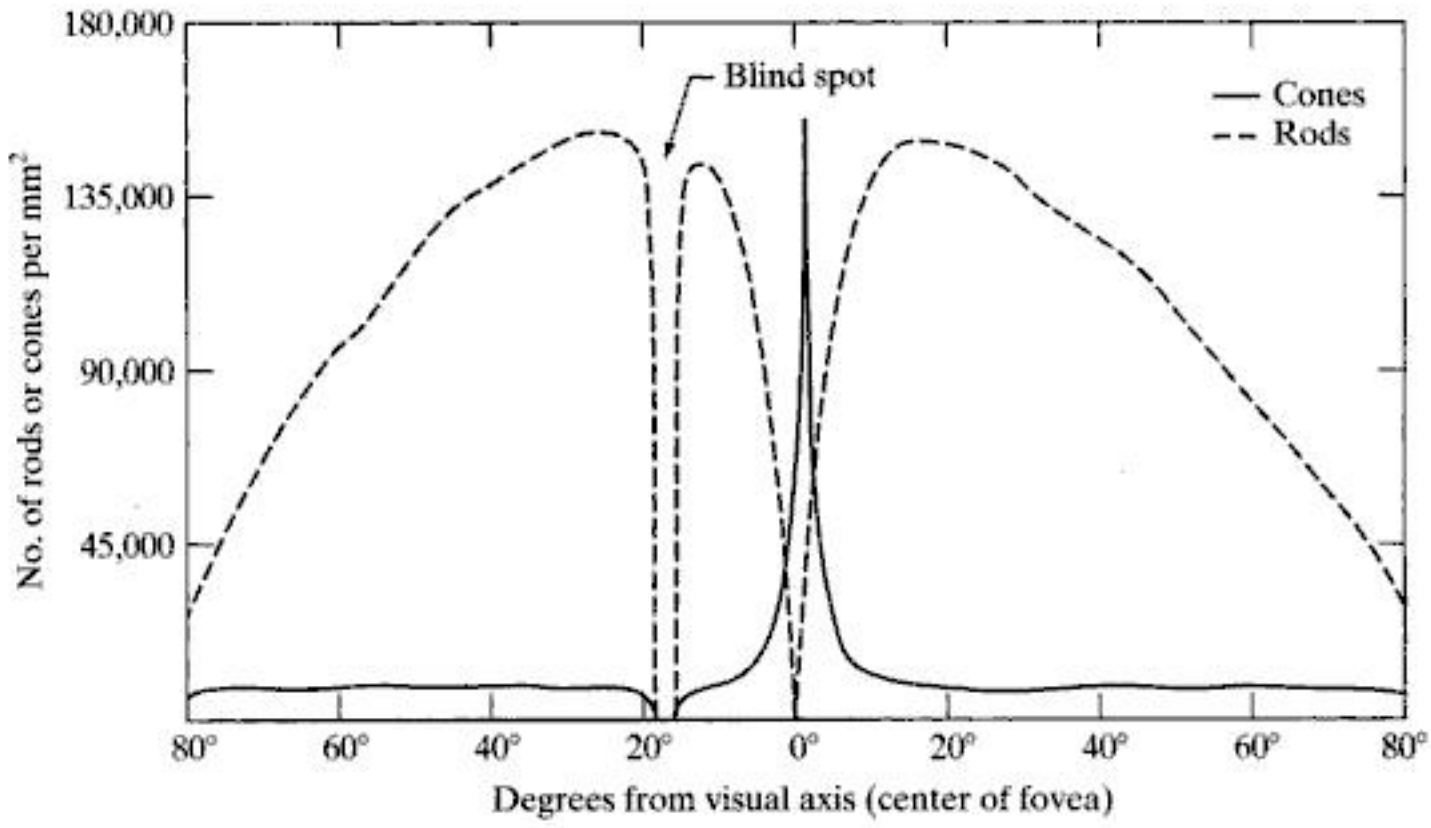


FIGURE 2.2 Distribution of rods and cones in the retina.

fovea (that is, in degrees off axis, as measured by the angle formed by the visual axis and a line passing through the center of the lens and intersecting the retina). Note in Fig. 2.2 that cones are most dense in the center of the retina (in the center area of the fovea). Note also that rods increase in density from the center out to approximately 20° off axis and then decrease in density out to the extreme periphery of the retina.

The fovea itself is a circular indentation in the retina of about 1.5 mm in diameter. However, in terms of future discussions, talking about square or rectangular arrays of sensing elements is more useful. Thus, by taking some liberty in interpretation, we can view the fovea as a square sensor array of size 1.5 mm × 1.5 mm. The density of cones in that area of the retina is approximately 150,000 elements per mm^2. Based on these approximations, the number of cones in the region of highest acuity in the eye is about 337,000 elements. Just in terms of raw resolving power, a charge-coupled device (CCD) imaging chip of medium resolution can have this number of elements in a receptor array no larger than 5 mm × 5 mm. While the ability of humans to integrate intelligence and experience with vision makes these types of number comparisons somewhat superficial, keep in mind for future discussions that the basic ability of the eye to resolve detail certainly is comparable to current electronic imaging sensors.

2.1.2 Image Formation in the Eye

In an ordinary photographic camera, the lens has a fixed focal length, and focusing at various distances is achieved by varying the distance between the lens and the imaging plane, where the film (or imaging chip in the case of a digital camera) is located. In the human eye, the converse is true; the distance between the lens and the imaging region (the retina) is fixed, and the focal length needed to achieve proper focus is obtained by varying the shape of the lens. The fibers in the ciliary body accomplish this, flattening or thickening the

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

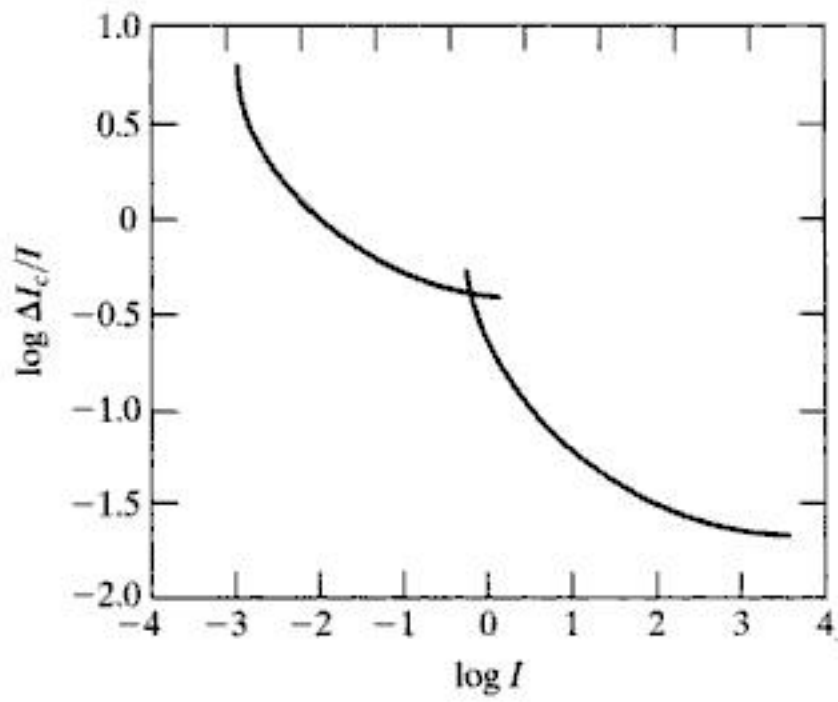


FIGURE 2.6
Typical Weber ratio as a function of intensity.

A plot of $\log \Delta I_c/I$ as a function of $\log I$ has the general shape shown in Fig. 2.6. This curve shows that brightness discrimination is poor (the Weber ratio is large) at low levels of illumination, and it improves significantly (the Weber ratio decreases) as background illumination increases. The two branches in the curve reflect the fact that at low levels of illumination vision is carried out by the rods, whereas at high levels (showing better discrimination) vision is the function of cones.

If the background illumination is held constant and the intensity of the other source, instead of flashing, is now allowed to vary incrementally from never being perceived to always being perceived, the typical observer can discern a total of one to two dozen different intensity changes. Roughly, this result is related to the number of different intensities a person can see at any one point in a monochrome image. This result does not mean that an image can be represented by such a small number of intensity values because, as the eye roams about the image, the average background changes, thus allowing a *different* set of incremental changes to be detected at each new adaptation level. The net consequence is that the eye is capable of a much broader range of *overall* intensity discrimination. In fact, we show in Section 2.4.3 that the eye is capable of detecting objectionable contouring effects in monochrome images whose overall intensity is represented by fewer than approximately two dozen levels.

Two phenomena clearly demonstrate that perceived brightness is not a simple function of intensity. The first is based on the fact that the visual system tends to undershoot or overshoot around the boundary of regions of different intensities. Figure 2.7(a) shows a striking example of this phenomenon. Although the intensity of the stripes is constant, we actually perceive a brightness pattern that is strongly scalloped near the boundaries [Fig. 2.7(c)]. These seemingly scalloped bands are called *Mach bands* after Ernst Mach, who first described the phenomenon in 1865.

The second phenomenon, called *simultaneous contrast*, is related to the fact that a region's perceived brightness does not depend simply on its intensity, as Fig. 2.8 demonstrates. All the center squares have exactly the same intensity.

a
b
c

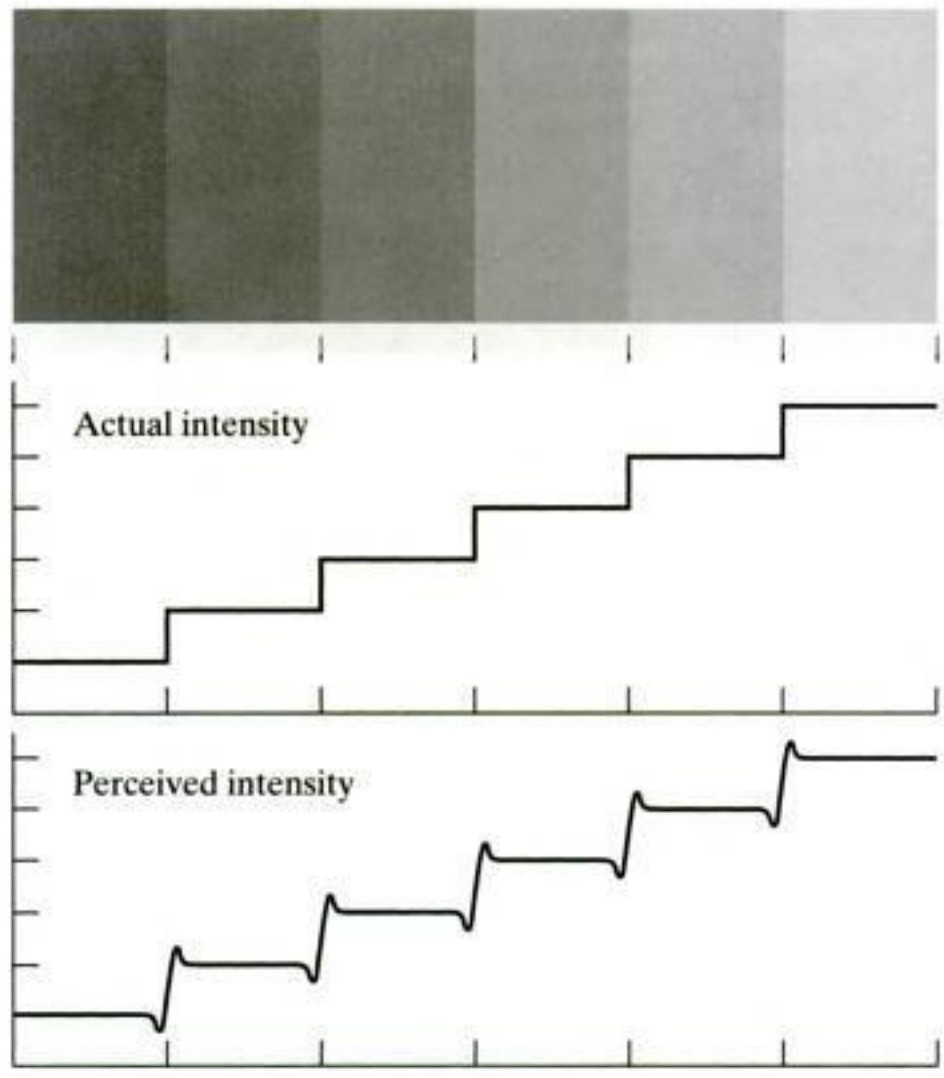
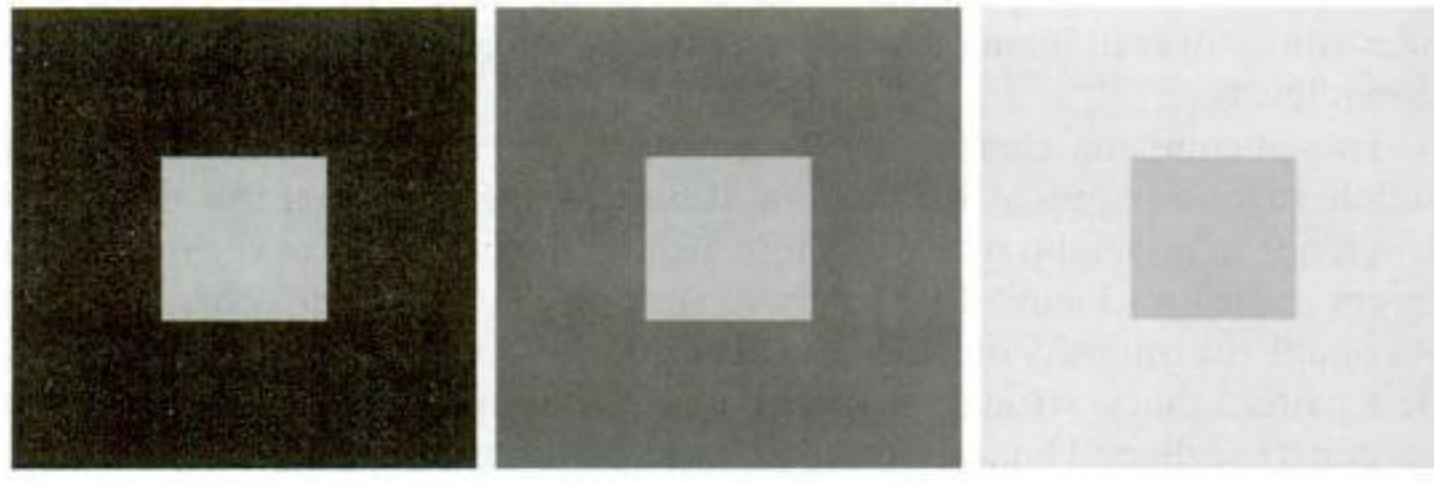


FIGURE 2.7
Illustration of the Mach band effect. Perceived intensity is not a simple function of actual intensity.

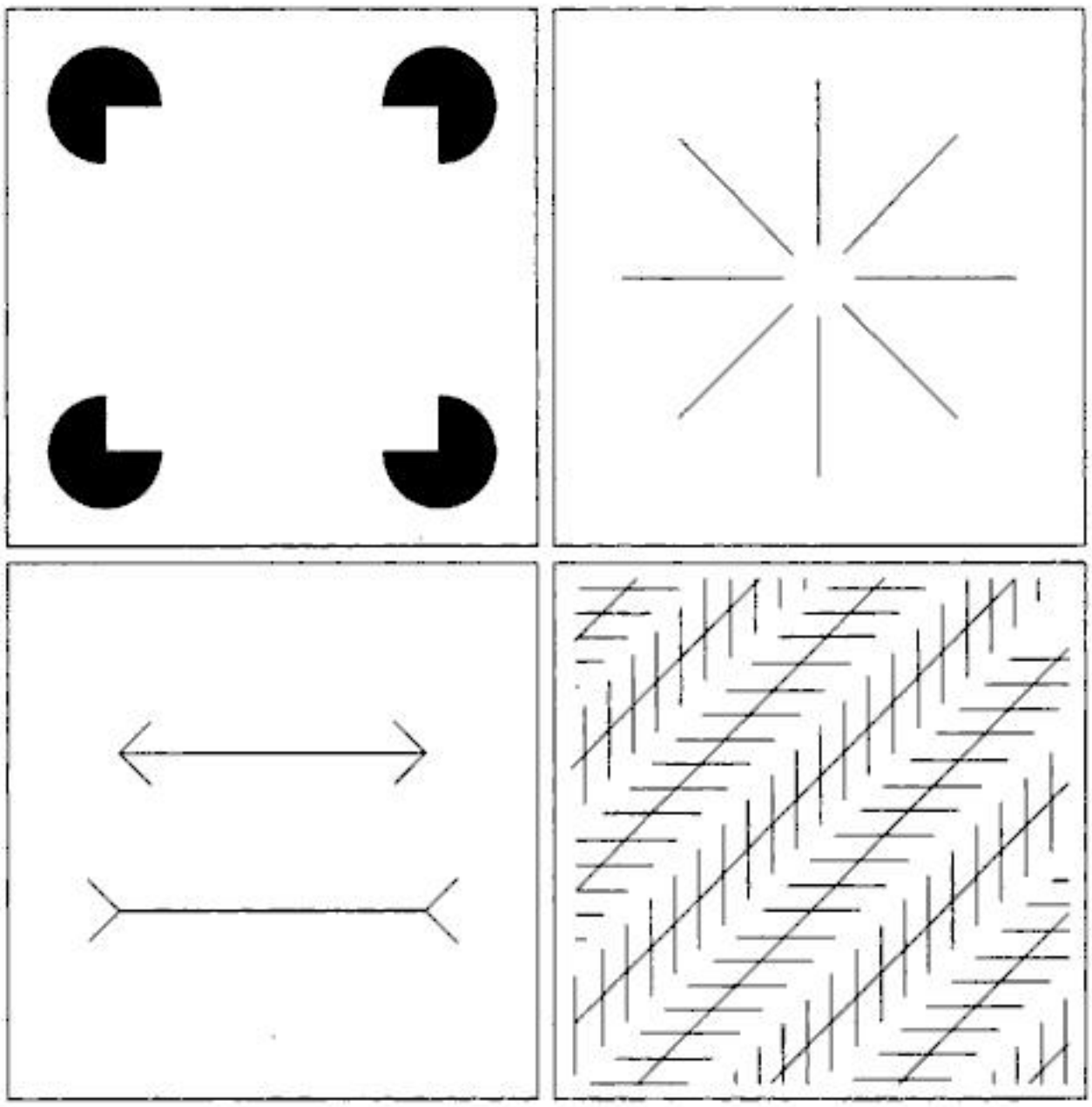
However, they appear to the eye to become darker as the background gets lighter. A more familiar example is a piece of paper that seems white when lying on a desk, but can appear totally black when used to shield the eyes while looking directly at a bright sky.

Other examples of human perception phenomena are optical illusions, in which the eye fills in nonexisting information or wrongly perceives geometrical properties of objects. Figure 2.9 shows some examples. In Fig. 2.9(a), the outline of a square is seen clearly, despite the fact that no lines defining such a figure are part of the image. The same effect, this time with a circle, can be seen in Fig. 2.9(b); note how just a few lines are sufficient to give the illusion of a



a b c

FIGURE 2.8 Examples of simultaneous contrast. All the inner squares have the same intensity, but they appear progressively darker as the background becomes lighter.



a b
c d

FIGURE 2.9 Some well-known optical illusions.

complete circle. The two horizontal line segments in Fig. 2.9(c) are of the same length, but one appears shorter than the other. Finally, all lines in Fig. 2.9(d) that are oriented at 45° are equidistant and parallel. Yet the crosshatching creates the illusion that those lines are far from being parallel. Optical illusions are a characteristic of the human visual system that is not fully understood.

2.2 Light and the Electromagnetic Spectrum

The electromagnetic spectrum was introduced in Section 1.3. We now consider this topic in more detail. In 1666, Sir Isaac Newton discovered that when a beam of sunlight is passed through a glass prism, the emerging beam of light is not white but consists instead of a continuous spectrum of colors ranging from violet at one end to red at the other. As Fig. 2.10 shows, the range of colors we perceive in visible light represents a very small portion of the electromagnetic spectrum. On one end of the spectrum are radio waves with wavelengths billions of times longer than those of visible light. On the other end of the spectrum are gamma rays with wavelengths millions of times smaller than those of visible light. The electromagnetic spectrum can be expressed in terms of wavelength, frequency, or energy. Wavelength (λ) and frequency (ν) are related by the expression

$$\lambda = \frac{c}{\nu} \tag{2.2-1}$$

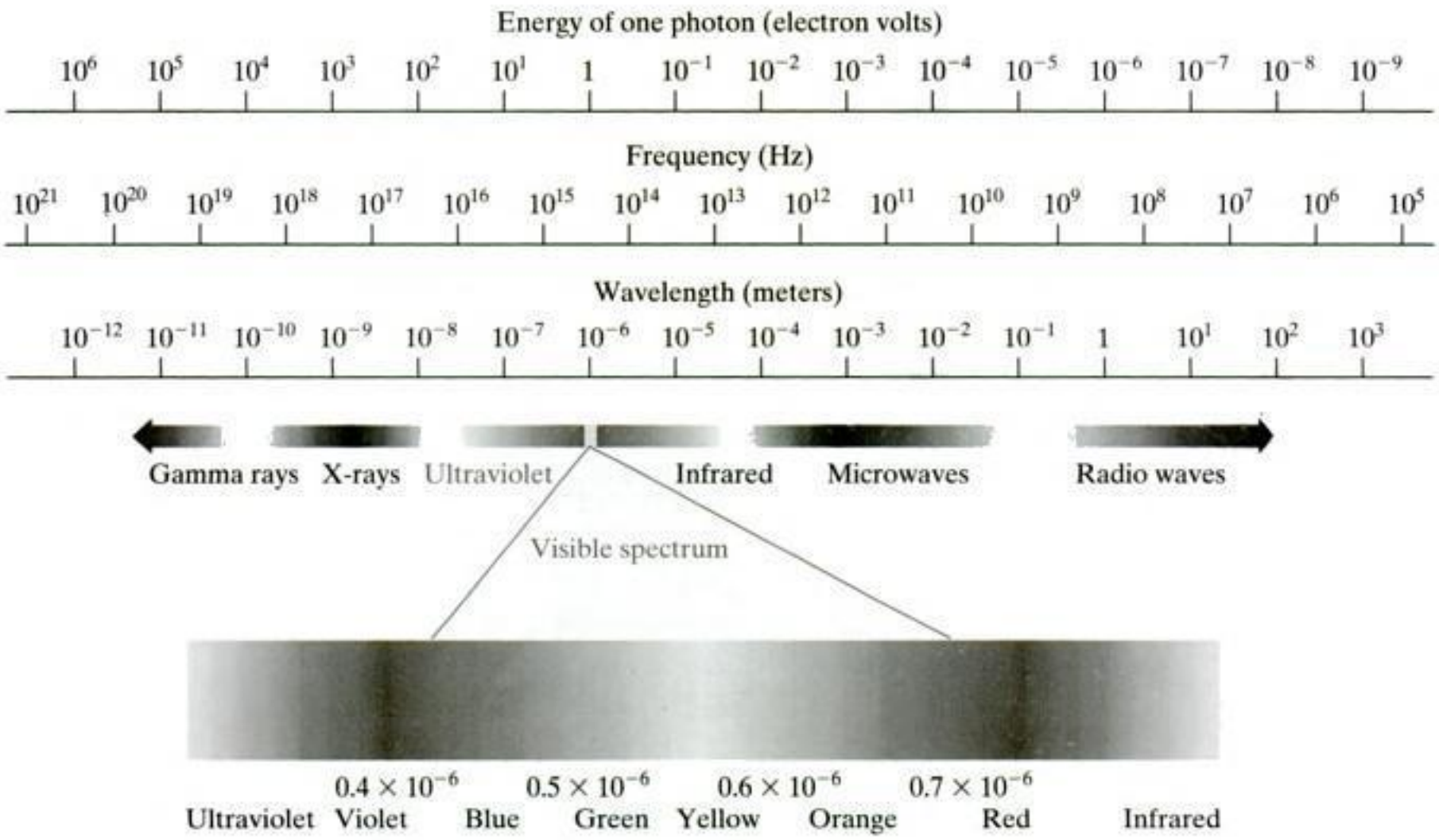


FIGURE 2.10 The electromagnetic spectrum. The visible spectrum is shown zoomed to facilitate explanation, but note that the visible spectrum is a rather narrow portion of the EM spectrum.

where c is the speed of light (2.998×10^8 m/s). The energy of the various components of the electromagnetic spectrum is given by the expression

$$E = h\nu \tag{2.2-2}$$

where h is Planck's constant. The units of wavelength are meters, with the terms *microns* (denoted μm and equal to 10^{-6} m) and *nanometers* (denoted nm and equal to 10^{-9} m) being used just as frequently. Frequency is measured in Hertz (Hz), with one Hertz being equal to one cycle of a sinusoidal wave per second. A commonly used unit of energy is the electron-volt.

Electromagnetic waves can be visualized as propagating sinusoidal waves with wavelength λ (Fig. 2.11), or they can be thought of as a stream of massless particles, each traveling in a wavelike pattern and moving at the speed of light. Each massless particle contains a certain amount (or bundle) of energy. Each

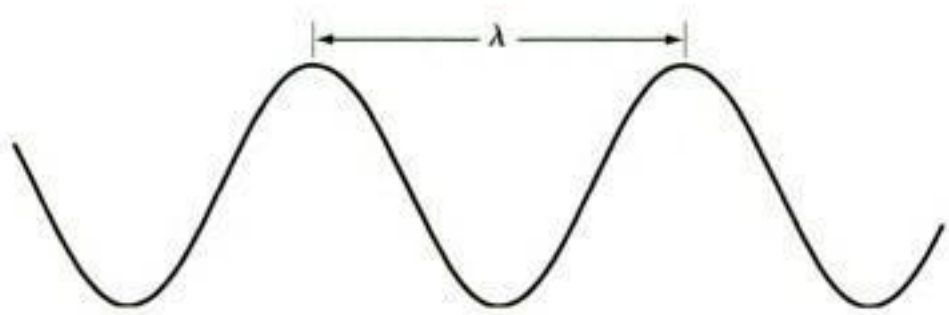


FIGURE 2.11 Graphical representation of one wavelength.

bundle of energy is called a *photon*. We see from Eq. (2.2-2) that energy is proportional to frequency, so the higher-frequency (shorter wavelength) electromagnetic phenomena carry more energy per photon. Thus, radio waves have photons with low energies, microwaves have more energy than radio waves, infrared still more, then visible, ultraviolet, X-rays, and finally gamma rays, the most energetic of all. This is the reason why gamma rays are so dangerous to living organisms.

Light is a particular type of electromagnetic radiation that can be sensed by the human eye. The visible (color) spectrum is shown expanded in Fig. 2.10 for the purpose of discussion (we consider color in much more detail in Chapter 6). The visible band of the electromagnetic spectrum spans the range from approximately 0.43 μm (violet) to about 0.79 μm (red). For convenience, the color spectrum is divided into six broad regions: violet, blue, green, yellow, orange, and red. No color (or other component of the electromagnetic spectrum) ends abruptly, but rather each range blends smoothly into the next, as shown in Fig. 2.10.

The colors that humans perceive in an object are determined by the nature of the light *reflected* from the object. A body that reflects light relatively balanced in all visible wavelengths appears white to the observer. However, a body that favors reflectance in a limited range of the visible spectrum exhibits some shades of color. For example, green objects reflect light with wavelengths primarily in the 500 to 570 nm range while absorbing most of the energy at other wavelengths.

Light that is void of color is called *monochromatic* (or *achromatic*) *light*. The only attribute of monochromatic light is its *intensity* or amount. Because the intensity of monochromatic light is perceived to vary from black to grays and finally to white, the term *gray level* is used commonly to denote monochromatic intensity. We use the terms *intensity* and *gray level* interchangeably in subsequent discussions. The range of measured values of monochromatic light from black to white is usually called the *gray scale*, and monochromatic images are frequently referred to as *gray-scale images*.

Chromatic (*color*) *light* spans the electromagnetic energy spectrum from approximately 0.43 to 0.79 μm, as noted previously. In addition to frequency, three basic quantities are used to describe the quality of a chromatic light source: radiance, luminance, and brightness. *Radiance* is the total amount of energy that flows from the light source, and it is usually measured in watts (W). *Luminance*, measured in lumens (lm), gives a measure of the amount of energy an observer *perceives* from a light source. For example, light emitted from a source operating in the far infrared region of the spectrum could have significant energy (radiance), but an observer would hardly perceive it; its luminance would be almost zero. Finally, as discussed in Section 2.1, *brightness* is a subjective descriptor of light perception that is practically impossible to measure. It embodies the achromatic notion of intensity and is one of the key factors in describing color sensation.

Continuing with the discussion of Fig. 2.10, we note that at the short-wavelength end of the electromagnetic spectrum, we have gamma rays and X-rays. As discussed in Section 1.3.1, gamma radiation is important for medical and astronomical imaging, and for imaging radiation in nuclear environments.

Hard (high-energy) X-rays are used in industrial applications. Chest and dental X-rays are in the lower energy (soft) end of the X-ray band. The soft X-ray band transitions into the far ultraviolet light region, which in turn blends with the visible spectrum at longer wavelengths. Moving still higher in wavelength, we encounter the infrared band, which radiates heat, a fact that makes it useful in imaging applications that rely on "heat signatures." The part of the infrared band close to the visible spectrum is called the *near-infrared* region. The opposite end of this band is called the *far-infrared* region. This latter region blends with the microwave band. This band is well known as the source of energy in microwave ovens, but it has many other uses, including communication and radar. Finally, the radio wave band encompasses television as well as AM and FM radio. In the higher energies, radio signals emanating from certain stellar bodies are useful in astronomical observations. Examples of images in most of the bands just discussed are given in Section 1.3.

In principle, if a sensor can be developed that is capable of detecting energy radiated by a band of the electromagnetic spectrum, we can image events of interest in that band. It is important to note, however, that the wavelength of an electromagnetic wave required to "see" an object must be of the same size as or smaller than the object. For example, a water molecule has a diameter on the order of 10^{-10} m. Thus, to study molecules, we would need a source capable of emitting in the far ultraviolet or soft X-ray region. This limitation, along with the physical properties of the sensor material, establishes the fundamental limits on the capability of imaging sensors, such as visible, infrared, and other sensors in use today.

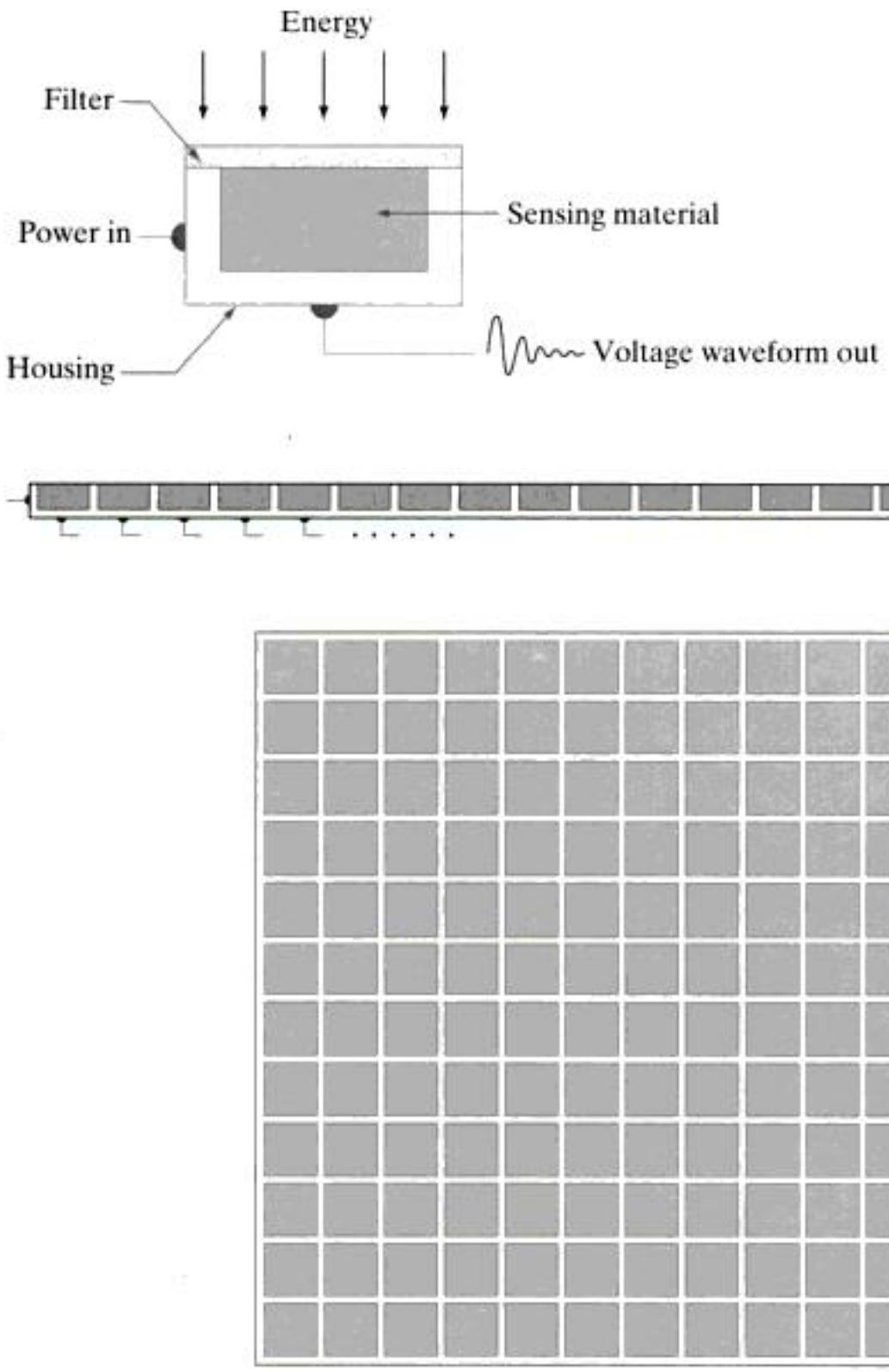
Although imaging is based predominantly on energy radiated by electromagnetic waves, this is not the only method for image generation. For example, as discussed in Section 1.3.7, sound reflected from objects can be used to form ultrasonic images. Other major sources of digital images are electron beams for electron microscopy and synthetic images used in graphics and visualization.

2.3 Image Sensing and Acquisition

Most of the images in which we are interested are generated by the combination of an "illumination" source and the reflection or absorption of energy from that source by the elements of the "scene" being imaged. We enclose *illumination* and *scene* in quotes to emphasize the fact that they are considerably more general than the familiar situation in which a visible light source illuminates a common everyday 3-D (three-dimensional) scene. For example, the illumination may originate from a source of electromagnetic energy such as radar, infrared, or X-ray system. But, as noted earlier, it could originate from less traditional sources, such as ultrasound or even a computer-generated illumination pattern. Similarly, the scene elements could be familiar objects, but they can just as easily be molecules, buried rock formations, or a human brain. Depending on the nature of the source, illumination energy is reflected from, or transmitted through, objects. An example in the first category is light

reflected from a planar surface. An example in the second category is when X-rays pass through a patient's body for the purpose of generating a diagnostic X-ray film. In some applications, the reflected or transmitted energy is focused onto a photoconverter (e.g., a phosphor screen), which converts the energy into visible light. Electron microscopy and some applications of gamma imaging use this approach.

Figure 2.12 shows the three principal sensor arrangements used to transform illumination energy into digital images. The idea is simple: Incoming energy is transformed into a voltage by the combination of input electrical power and sensor material that is responsive to the particular type of energy being detected. The output voltage waveform is the response of the sensor(s), and a digital quantity is obtained from each sensor by digitizing its response. In this section, we look at the principal modalities for image sensing and generation. Image digitizing is discussed in Section 2.4.



a
b
c

FIGURE 2.12
(a) Single imaging sensor.
(b) Line sensor.
(c) Array sensor.

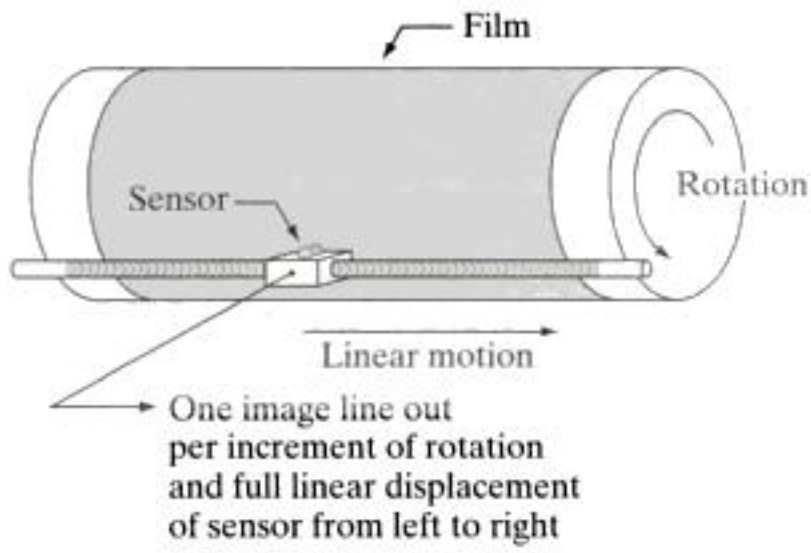


FIGURE 2.13
Combining a single sensor with motion to generate a 2-D image.

2.3.1 Image Acquisition Using a Single Sensor

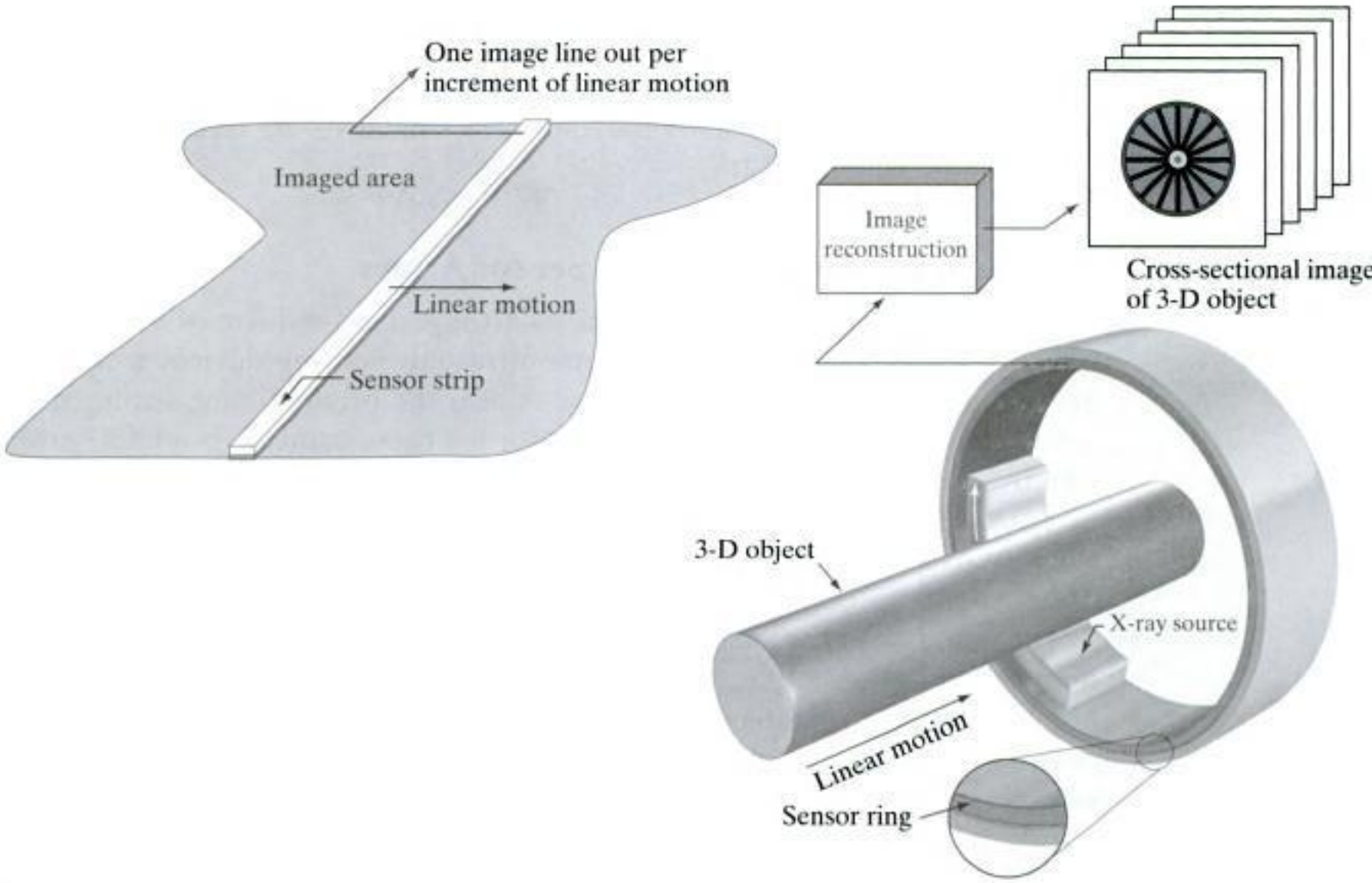
Figure 2.12(a) shows the components of a single sensor. Perhaps the most familiar sensor of this type is the photodiode, which is constructed of silicon materials and whose output voltage waveform is proportional to light. The use of a filter in front of a sensor improves selectivity. For example, a green (pass) filter in front of a light sensor favors light in the green band of the color spectrum. As a consequence, the sensor output will be stronger for green light than for other components in the visible spectrum.

In order to generate a 2-D image using a single sensor, there has to be relative displacements in both the x- and y-directions between the sensor and the area to be imaged. Figure 2.13 shows an arrangement used in high-precision scanning, where a film negative is mounted onto a drum whose mechanical rotation provides displacement in one dimension. The single sensor is mounted on a lead screw that provides motion in the perpendicular direction. Because mechanical motion can be controlled with high precision, this method is an inexpensive (but slow) way to obtain high-resolution images. Other similar mechanical arrangements use a flat bed, with the sensor moving in two linear directions. These types of mechanical digitizers sometimes are referred to as *microdensitometers*.

Another example of imaging with a single sensor places a laser source coincident with the sensor. Moving mirrors are used to control the outgoing beam in a scanning pattern and to direct the reflected laser signal onto the sensor. This arrangement can be used also to acquire images using strip and array sensors, which are discussed in the following two sections.

2.3.2 Image Acquisition Using Sensor Strips

A geometry that is used much more frequently than single sensors consists of an in-line arrangement of sensors in the form of a sensor strip, as Fig. 2.12(b) shows. The strip provides imaging elements in one direction. Motion perpendicular to the strip provides imaging in the other direction, as shown in Fig. 2.14(a). This is the type of arrangement used in most flat bed scanners. Sensing devices with 4000 or more in-line sensors are possible. In-line sensors are used routinely in airborne imaging applications, in which the imaging system is mounted on an aircraft that



a b

FIGURE 2.14 (a) Image acquisition using a linear sensor strip. (b) Image acquisition using a circular sensor strip.

flies at a constant altitude and speed over the geographical area to be imaged. One-dimensional imaging sensor strips that respond to various bands of the electromagnetic spectrum are mounted perpendicular to the direction of flight. The imaging strip gives one line of an image at a time, and the motion of the strip completes the other dimension of a two-dimensional image. Lenses or other focusing schemes are used to project the area to be scanned onto the sensors.

Sensor strips mounted in a ring configuration are used in medical and industrial imaging to obtain cross-sectional ("slice") images of 3-D objects, as Fig. 2.14(b) shows. A rotating X-ray source provides illumination and the sensors opposite the source collect the X-ray energy that passes through the object (the sensors obviously have to be sensitive to X-ray energy). This is the basis for medical and industrial computerized axial tomography (CAT) imaging as indicated in Sections 1.2 and 1.3.2. It is important to note that the output of the sensors must be processed by reconstruction algorithms whose objective is to transform the sensed data into meaningful cross-sectional images (see Section 5.11). In other words, images are not obtained directly from the sensors by motion alone; they require extensive processing. A 3-D digital volume consisting of stacked images is generated as the object is moved in a direction

perpendicular to the sensor ring. Other modalities of imaging based on the CAT principle include magnetic resonance imaging (MRI) and positron emission tomography (PET). The illumination sources, sensors, and types of images are different, but conceptually they are very similar to the basic imaging approach shown in Fig. 2.14(b).

2.3.3 Image Acquisition Using Sensor Arrays

Figure 2.12(c) shows individual sensors arranged in the form of a 2-D array. Numerous electromagnetic and some ultrasonic sensing devices frequently are arranged in an array format. This is also the predominant arrangement found in digital cameras. A typical sensor for these cameras is a CCD array, which can be manufactured with a broad range of sensing properties and can be packaged in rugged arrays of 4000×4000 elements or more. CCD sensors are used widely in digital cameras and other light sensing instruments. The response of each sensor is proportional to the integral of the light energy projected onto the surface of the sensor, a property that is used in astronomical and other applications requiring low noise images. Noise reduction is achieved by letting the sensor integrate the input light signal over minutes or even hours. Because the sensor array in Fig. 2.12(c) is two-dimensional, its key advantage is that a complete image can be obtained by focusing the energy pattern onto the surface of the array. Motion obviously is not necessary, as is the case with the sensor arrangements discussed in the preceding two sections.

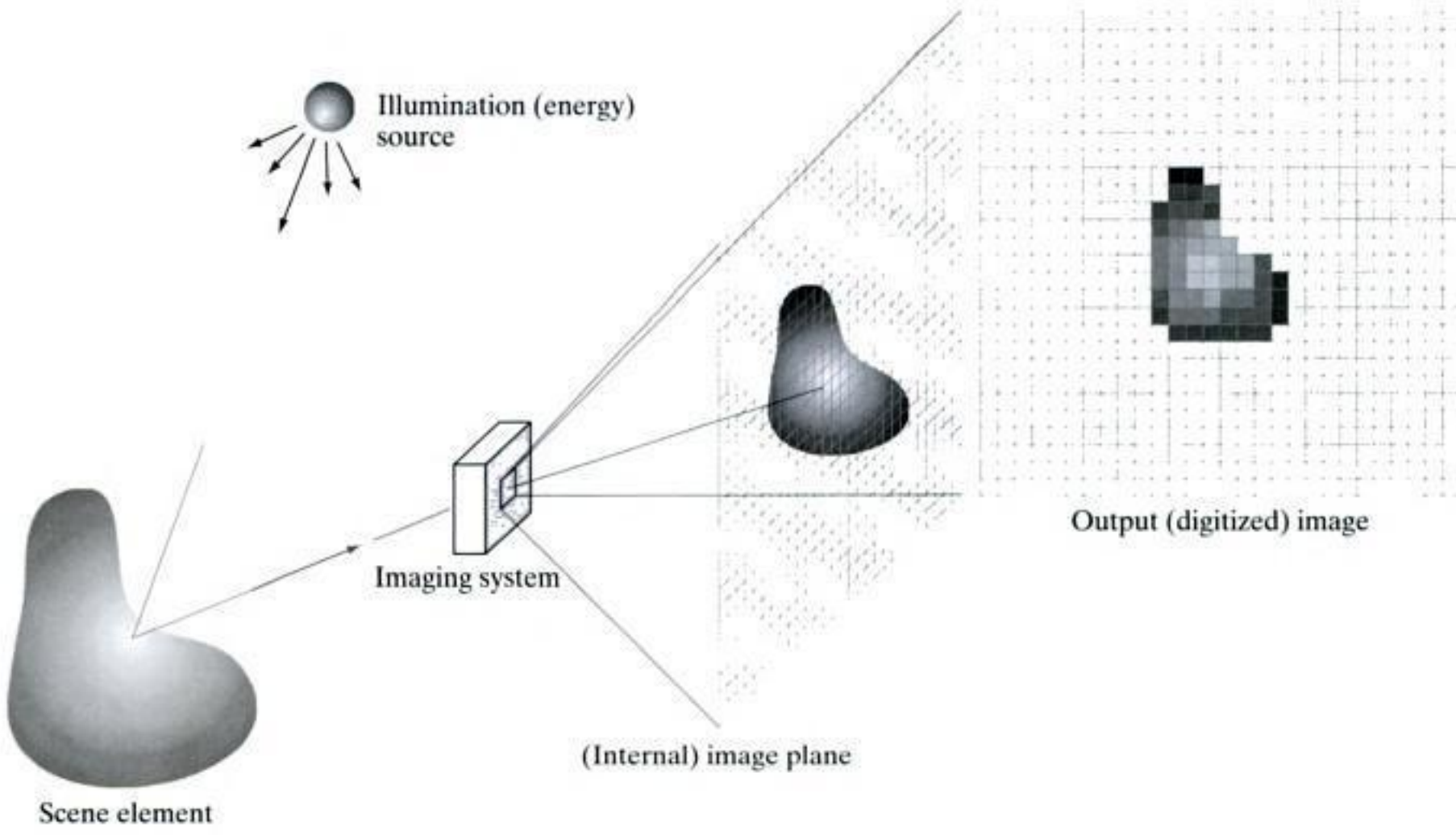
In some cases, we image the source directly, as in obtaining images of the sun.

The principal manner in which array sensors are used is shown in Fig. 2.15. This figure shows the energy from an illumination source being reflected from a scene element (as mentioned at the beginning of this section, the energy also could be transmitted through the scene elements). The first function performed by the imaging system in Fig. 2.15(c) is to collect the incoming energy and focus it onto an image plane. If the illumination is light, the front end of the imaging system is an optical lens that projects the viewed scene onto the lens focal plane, as Fig. 2.15(d) shows. The sensor array, which is coincident with the focal plane, produces outputs proportional to the integral of the light received at each sensor. Digital and analog circuitry sweep these outputs and convert them to an analog signal, which is then digitized by another section of the imaging system. The output is a digital image, as shown diagrammatically in Fig. 2.15(e). Conversion of an image into digital form is the topic of Section 2.4.

Image intensities can become negative during processing or as a result of interpretation. For example, in radar images objects moving toward a radar system often are interpreted as having negative velocities while objects moving away are interpreted as having positive velocities. Thus, a velocity image might be coded as having both positive and negative values. When storing and displaying images, we normally scale the intensities so that the smallest negative value becomes 0 (see Section 2.6.3 regarding intensity scaling).

2.3.4 A Simple Image Formation Model

As introduced in Section 1.1, we denote images by two-dimensional functions of the form $f(x, y)$. The value or amplitude of f at spatial coordinates (x, y) is a positive scalar quantity whose physical meaning is determined by the source of the image. When an image is generated from a physical process, its intensity values are proportional to energy radiated by a physical source (e.g., electromagnetic waves). As a consequence, $f(x, y)$ must be nonzero



a b c d e

FIGURE 2.15 An example of the digital image acquisition process. (a) Energy ("illumination") source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

and finite; that is,

$$0 < f(x, y) < \infty \tag{2.3-1}$$

The function $f(x, y)$ may be characterized by two components: (1) the amount of source illumination incident on the scene being viewed, and (2) the amount of illumination reflected by the objects in the scene. Appropriately, these are called the *illumination* and *reflectance* components and are denoted by $i(x, y)$ and $r(x, y)$, respectively. The two functions combine as a product to form $f(x, y)$:

$$f(x, y) = i(x, y)r(x, y) \tag{2.3-2}$$

where

$$0 < i(x, y) < \infty \tag{2.3-3}$$

and

$$0 < r(x, y) < 1 \tag{2.3-4}$$

Equation (2.3-4) indicates that reflectance is bounded by 0 (total absorption) and 1 (total reflectance). The nature of $i(x, y)$ is determined by the illumination source, and $r(x, y)$ is determined by the characteristics of the imaged objects. It is noted that these expressions also are applicable to images formed via transmission of the illumination through a medium, such as a chest X-ray.

In this case, we would deal with a *transmissivity* instead of a *reflectivity* function, but the limits would be the same as in Eq. (2.3-4), and the image function formed would be modeled as the product in Eq. (2.3-2).

EXAMPLE 2.1: Some typical values of illumination and reflectance.

■ The values given in Eqs. (2.3-3) and (2.3-4) are theoretical bounds. The following *average* numerical figures illustrate some typical ranges of $i(x, y)$ for visible light. On a clear day, the sun may produce in excess of 90,000 lm/m^2 of illumination on the surface of the Earth. This figure decreases to less than 10,000 lm/m^2 on a cloudy day. On a clear evening, a full moon yields about 0.1 lm/m^2 of illumination. The typical illumination level in a commercial office is about 1000 lm/m^2. Similarly, the following are typical values of $r(x, y)$: 0.01 for black velvet, 0.65 for stainless steel, 0.80 for flat-white wall paint, 0.90 for silver-plated metal, and 0.93 for snow. ■

Let the intensity (gray level) of a monochrome image at any coordinates (x_0, y_0) be denoted by

$$\ell = f(x_0, y_0) \tag{2.3-5}$$

From Eqs. (2.3-2) through (2.3-4), it is evident that ℓ lies in the range

$$L_{\min} \leq \ell \leq L_{\max} \tag{2.3-6}$$

In theory, the only requirement on $L_{\min}$ is that it be positive, and on $L_{\max}$ that it be finite. In practice, $L_{\min} = i_{\min} r_{\min}$ and $L_{\max} = i_{\max} r_{\max}$. Using the preceding average office illumination and range of reflectance values as guidelines, we may expect $L_{\min} \approx 10$ and $L_{\max} \approx 1000$ to be typical limits for indoor values in the absence of additional illumination.

The interval $[L_{\min}, L_{\max}]$ is called the *gray* (or *intensity*) *scale*. Common practice is to shift this interval numerically to the interval $[0, L - 1]$, where $\ell = 0$ is considered black and $\ell = L - 1$ is considered white on the gray scale. All intermediate values are shades of gray varying from black to white.

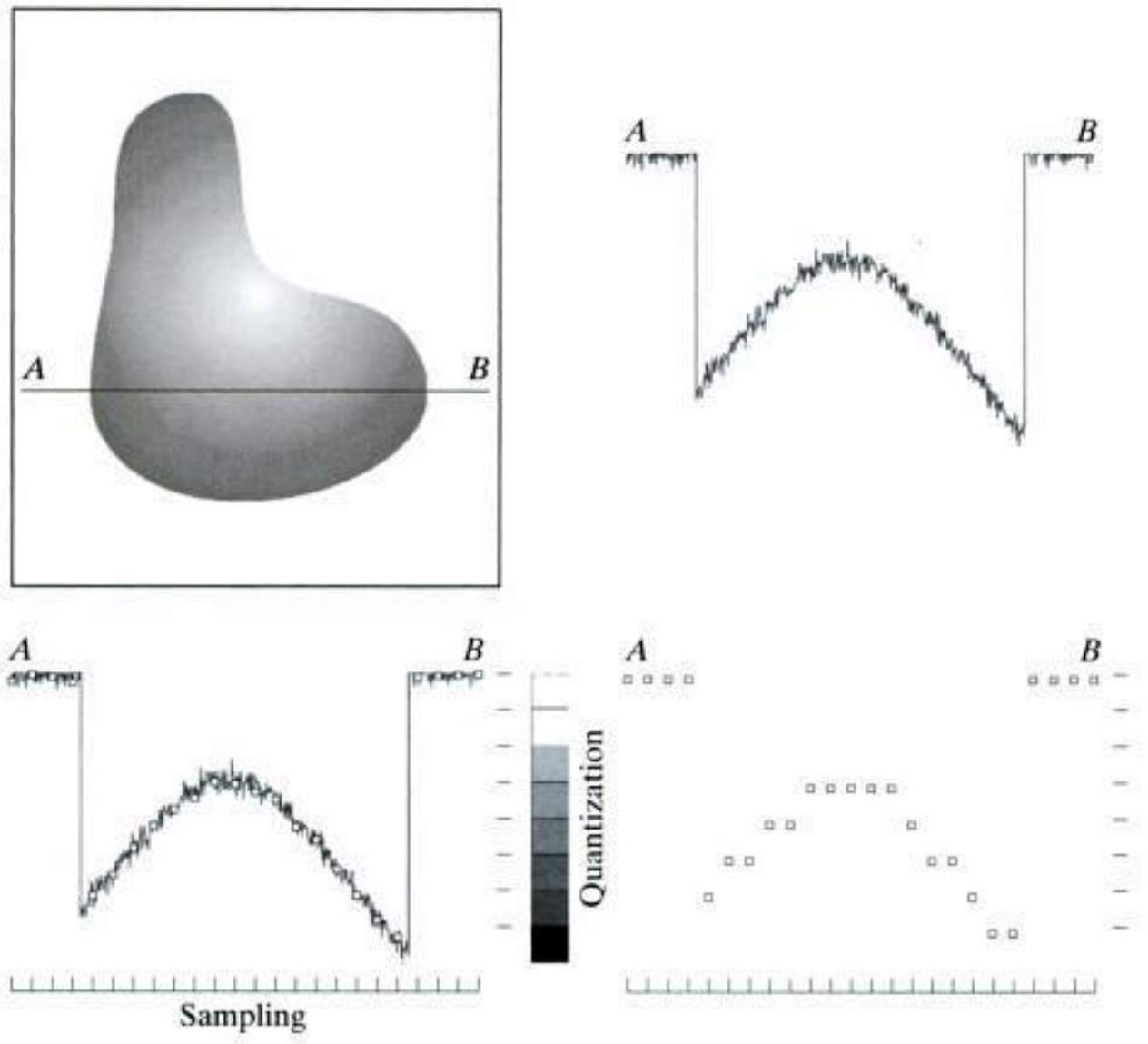
2.4 Image Sampling and Quantization

The discussion of sampling in this section is of an intuitive nature. We consider this topic in depth in Chapter 4.

From the discussion in the preceding section, we see that there are numerous ways to acquire images, but our objective in all is the same: to generate digital images from sensed data. The output of most sensors is a continuous voltage waveform whose amplitude and spatial behavior are related to the physical phenomenon being sensed. To create a digital image, we need to convert the continuous sensed data into digital form. This involves two processes: *sampling* and *quantization*.

2.4.1 Basic Concepts in Sampling and Quantization

The basic idea behind sampling and quantization is illustrated in Fig. 2.16. Figure 2.16(a) shows a continuous image f that we want to convert to digital form. An image may be continuous with respect to the x- and y-coordinates, and also in amplitude. To convert it to digital form, we have to sample the



a b
c d

FIGURE 2.16 Generating a digital image. (a) Continuous image. (b) A scan line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) Sampling and quantization. (d) Digital scan line.

function in both coordinates and in amplitude. Digitizing the coordinate values is called *sampling*. Digitizing the amplitude values is called *quantization*.

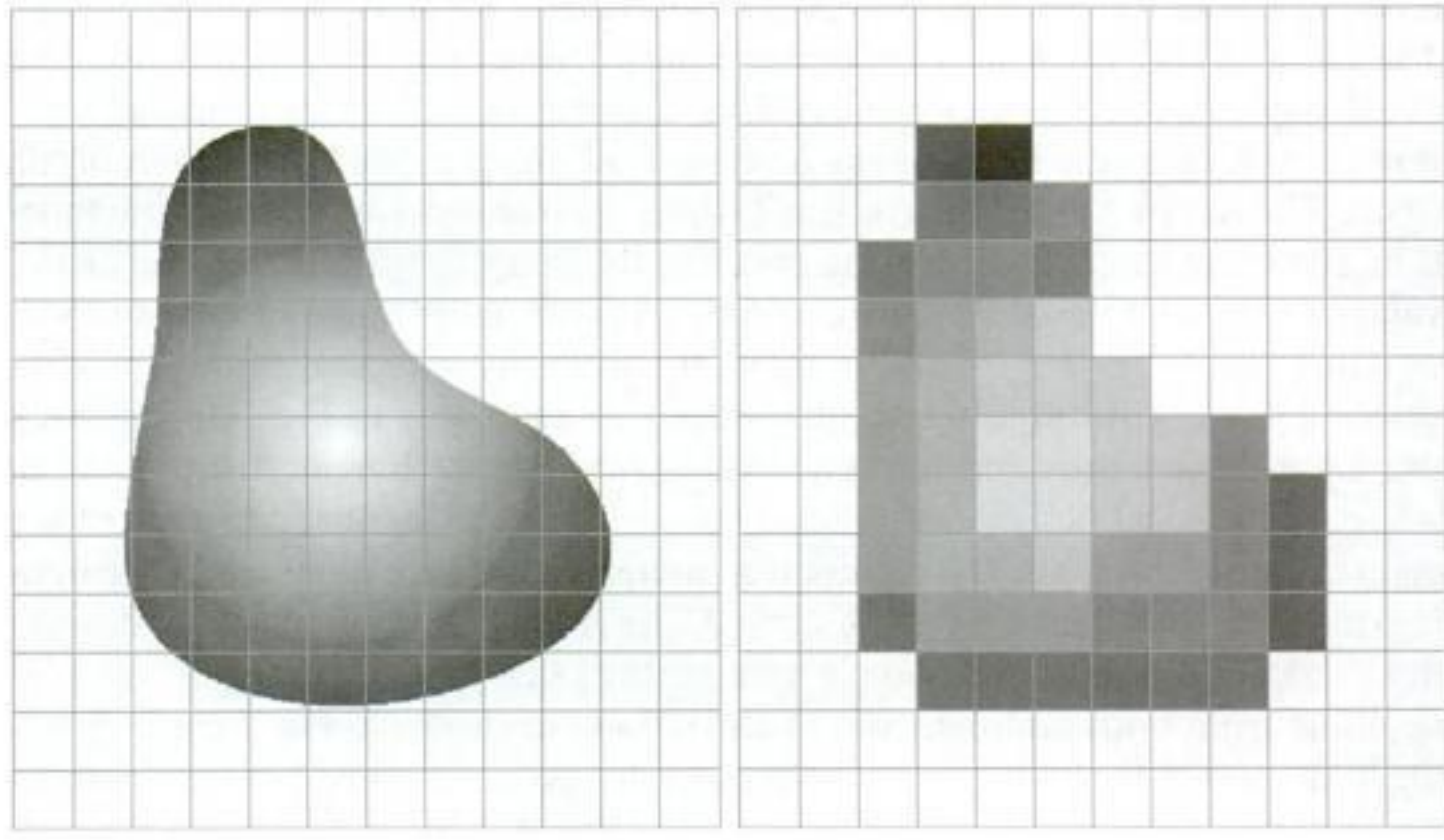
The one-dimensional function in Fig. 2.16(b) is a plot of amplitude (intensity level) values of the continuous image along the line segment AB in Fig. 2.16(a). The random variations are due to image noise. To sample this function, we take equally spaced samples along line AB, as shown in Fig. 2.16(c). The spatial location of each sample is indicated by a vertical tick mark in the bottom part of the figure. The samples are shown as small white squares superimposed on the function. The set of these discrete locations gives the sampled function. However, the values of the samples still span (vertically) a continuous range of intensity values. In order to form a digital function, the intensity values also must be converted (*quantized*) into discrete quantities. The right side of Fig. 2.16(c) shows the intensity scale divided into eight discrete intervals, ranging from black to white. The vertical tick marks indicate the specific value assigned to each of the eight intensity intervals. The continuous intensity levels are quantized by assigning one of the eight values to each sample. The assignment is made depending on the vertical proximity of a sample to a vertical tick mark. The digital samples resulting from both sampling and quantization are shown in Fig. 2.16(d). Starting at the top of the image and carrying out this procedure line by line produces a two-dimensional digital image. It is implied in Fig. 2.16 that, in addition to the number of discrete levels used, the accuracy achieved in quantization is highly dependent on the noise content of the sampled signal.

Sampling in the manner just described assumes that we have a continuous image in both coordinate directions as well as in amplitude. In practice, the

method of sampling is determined by the sensor arrangement used to generate the image. When an image is generated by a single sensing element combined with mechanical motion, as in Fig. 2.13, the output of the sensor is quantized in the manner described above. However, spatial sampling is accomplished by selecting the number of individual mechanical increments at which we activate the sensor to collect data. Mechanical motion can be made very exact so, in principle, there is almost no limit as to how fine we can sample an image using this approach. In practice, limits on sampling accuracy are determined by other factors, such as the quality of the optical components of the system.

When a sensing strip is used for image acquisition, the number of sensors in the strip establishes the sampling limitations in one image direction. Mechanical motion in the other direction can be controlled more accurately, but it makes little sense to try to achieve sampling density in one direction that exceeds the sampling limits established by the number of sensors in the other. Quantization of the sensor outputs completes the process of generating a digital image.

When a sensing array is used for image acquisition, there is no motion and the number of sensors in the array establishes the limits of sampling in both directions. Quantization of the sensor outputs is as before. Figure 2.17 illustrates this concept. Figure 2.17(a) shows a continuous image projected onto the plane of an array sensor. Figure 2.17(b) shows the image after sampling and quantization. Clearly, the quality of a digital image is determined to a large degree by the number of samples and discrete intensity levels used in sampling and quantization. However, as we show in Section 2.4.3, image content is also an important consideration in choosing these parameters.



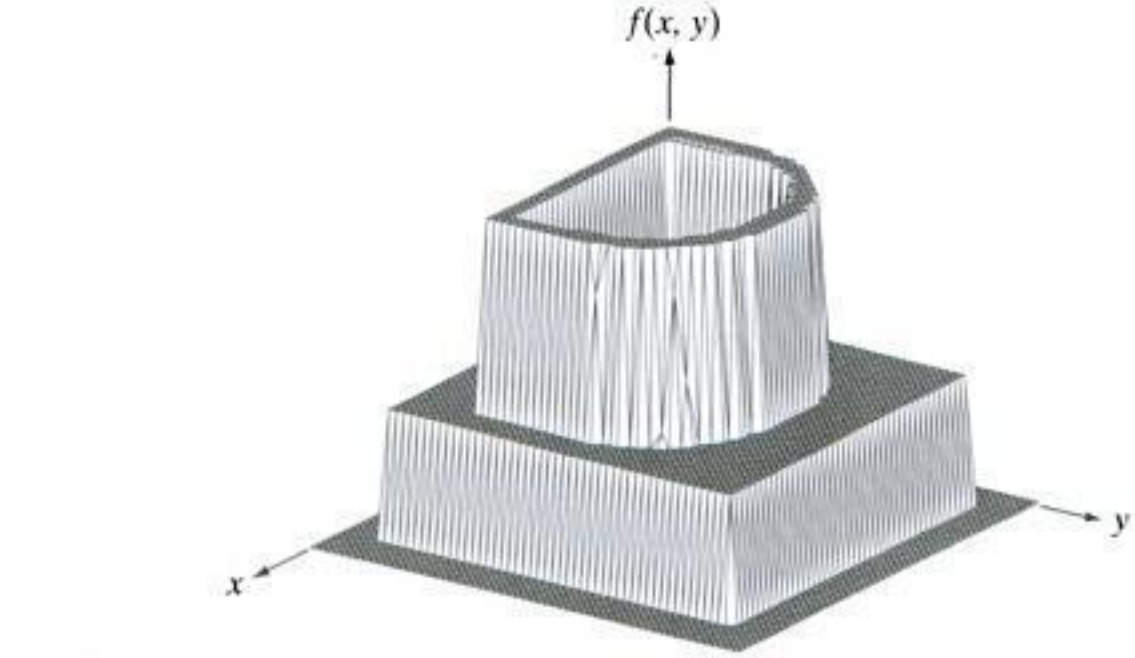
a b

FIGURE 2.17 (a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.

2.4.2 Representing Digital Images

Let $f(s, t)$ represent a continuous image function of two continuous variables, s and t. We convert this function into a *digital image* by sampling and quantization, as explained in the previous section. Suppose that we sample the continuous image into a 2-D array, $f(x, y)$, containing M rows and N columns, where (x, y) are discrete coordinates. For notational clarity and convenience, we use integer values for these discrete coordinates: $x = 0, 1, 2, \ldots, M - 1$ and $y = 0, 1, 2, \ldots, N - 1$. Thus, for example, the value of the digital image at the origin is $f(0, 0)$, and the next coordinate value along the first row is $f(0, 1)$. Here, the notation $(0, 1)$ is used to signify the second sample along the first row. It *does not* mean that these are the values of the physical coordinates when the image was sampled. In general, the value of the image at any coordinates (x, y) is denoted $f(x, y)$, where x and y are integers. The section of the real plane spanned by the coordinates of an image is called the *spatial domain*, with x and y being referred to as *spatial variables* or *spatial coordinates*.

As Fig. 2.18 shows, there are three basic ways to represent $f(x, y)$. Figure 2.18(a) is a plot of the function, with two axes determining spatial location



a
b c

FIGURE 2.18
(a) Image plotted as a surface.
(b) Image displayed as a visual intensity array.
(c) Image shown as a 2-D numerical array (0, .5, and 1 represent black, gray, and white, respectively).

and the third axis being the values of f (intensities) as a function of the two spatial variables x and y. Although we can infer the structure of the image in this example by looking at the plot, complex images generally are too detailed and difficult to interpret from such plots. This representation is useful when working with gray-scale sets whose elements are expressed as triplets of the form (x, y, z), where x and y are spatial coordinates and z is the value of f at coordinates (x, y). We work with this representation in Section 2.6.4.

The representation in Fig. 2.18(b) is much more common. It shows $f(x, y)$ as it would appear on a monitor or photograph. Here, the intensity of each point is proportional to the value of f at that point. In this figure, there are only three equally spaced intensity values. If the intensity is normalized to the interval [0, 1], then each point in the image has the value 0, 0.5, or 1. A monitor or printer simply converts these three values to black, gray, or white, respectively, as Fig. 2.18(b) shows. The third representation is simply to display the numerical values of $f(x, y)$ as an array (matrix). In this example, f is of size 600×600 elements, or 360,000 numbers. Clearly, printing the complete array would be cumbersome and convey little information. When developing algorithms, however, this representation is quite useful when only parts of the image are printed and analyzed as numerical values. Figure 2.18(c) conveys this concept graphically.

We conclude from the previous paragraph that the representations in Figs. 2.18(b) and (c) are the most useful. Image displays allow us to view results at a glance. Numerical arrays are used for processing and algorithm development. In equation form, we write the representation of an $M \times N$ numerical array as

$$f(x, y) = \begin{bmatrix} f(0, 0) & f(0, 1) & \cdots & f(0, N-1) \\ f(1, 0) & f(1, 1) & \cdots & f(1, N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1, 0) & f(M-1, 1) & \cdots & f(M-1, N-1) \end{bmatrix} \quad (2.4\text{-}1)$$

Both sides of this equation are equivalent ways of expressing a digital image quantitatively. The right side is a matrix of real numbers. Each element of this matrix is called an *image element, picture element, pixel*, or *pel*. The terms *image* and *pixel* are used throughout the book to denote a digital image and its elements.

In some discussions it is advantageous to use a more traditional matrix notation to denote a digital image and its elements:

$$\mathbf{A} = \begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,N-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,N-1} \\ \vdots & \vdots & & \vdots \\ a_{M-1,0} & a_{M-1,1} & \cdots & a_{M-1,N-1} \end{bmatrix} \quad (2.4\text{-}2)$$

Clearly, $a_{ij} = f(x = i, y = j) = f(i, j)$, so Eqs. (2.4-1) and (2.4-2) are identical matrices. We can even represent an image as a vector, **v**. For example, a column vector of size $MN \times 1$ is formed by letting the first M elements of **v** be the first column of **A**, the next M elements be the second column, and so on. Alternatively, we can use the rows instead of the columns of **A** to form such a vector. Either representation is valid, as long as we are consistent.

Returning briefly to Fig. 2.18, note that the origin of a digital image is at the top left, with the positive x-axis extending downward and the positive y-axis extending to the right. This is a conventional representation based on the fact that many image displays (e.g., TV monitors) sweep an image starting at the top left and moving to the right one row at a time. More important is the fact that the first element of a matrix is by convention at the top left of the array, so choosing the origin of $f(x, y)$ at that point makes sense mathematically. Keep in mind that this representation is the standard right-handed Cartesian coordinate system with which you are familiar.[†] We simply show the axes pointing downward and to the right, instead of to the right and up.

Expressing sampling and quantization in more formal mathematical terms can be useful at times. Let Z and R denote the set of integers and the set of real numbers, respectively. The sampling process may be viewed as partitioning the xy-plane into a grid, with the coordinates of the center of each cell in the grid being a pair of elements from the Cartesian product Z^2, which is the set of all ordered pairs of elements (z_i, z_j), with z_i and z_j being integers from Z. Hence, $f(x, y)$ is a digital image if (x, y) are integers from Z^2 and f is a function that assigns an intensity value (that is, a real number from the set of real numbers, R) to each distinct pair of coordinates (x, y). This functional assignment is the quantization process described earlier. If the intensity levels also are integers (as usually is the case in this and subsequent chapters), Z replaces R, and a digital image then becomes a 2-D function whose coordinates and amplitude values are integers.

This digitization process requires that decisions be made regarding the values for M, N, and for the number, L, of discrete intensity levels. There are no restrictions placed on M and N, other than they have to be positive integers. However, due to storage and quantizing hardware considerations, the number of intensity levels typically is an integer power of 2:

$$L = 2^k \tag{2.4-3}$$

We assume that the discrete levels are equally spaced and that they are integers in the interval $[0, L - 1]$. Sometimes, the range of values spanned by the gray scale is referred to informally as the dynamic range. This is a term used in different ways in different fields. Here, we define the *dynamic range* of an imaging system to be the ratio of the maximum measurable intensity to the minimum

Often, it is useful for computation or for algorithm development purposes to scale the L intensity values to the range $[0, 1]$, in which case they cease to be integers. However, in most cases these values are scaled back to the integer range $[0, L - 1]$ for image storage and display.

[†]Recall that a right-handed coordinate system is such that, when the index of the right hand points in the direction of the positive x-axis and the middle finger points in the (perpendicular) direction of the positive y-axis, the thumb points up. As Fig. 2.18(a) shows, this indeed is the case in our image coordinate system.

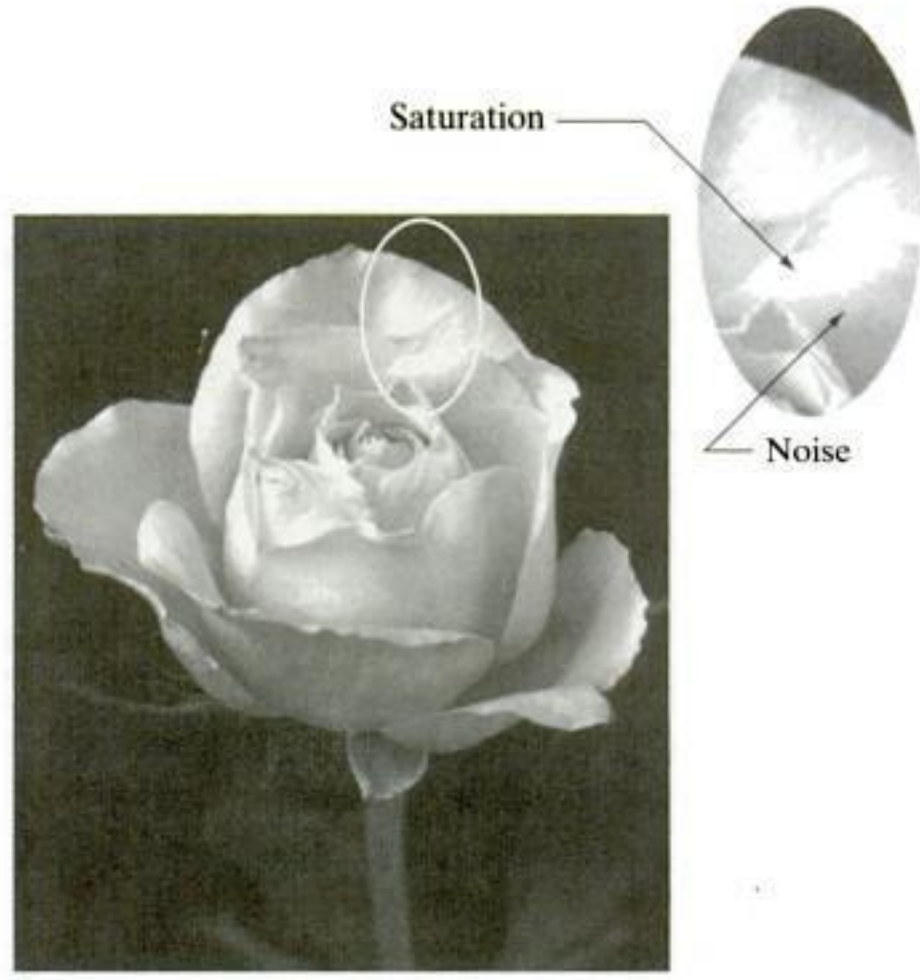


FIGURE 2.19 An image exhibiting saturation and noise. Saturation is the highest value beyond which all intensity levels are clipped (note how the entire saturated area has a high, *constant* intensity level). Noise in this case appears as a grainy texture pattern. Noise, especially in the darker regions of an image (e.g., the stem of the rose) masks the lowest detectable true intensity level.

detectable intensity level in the system. As a rule, the upper limit is determined by *saturation* and the lower limit by *noise* (see Fig. 2.19). Basically, dynamic range establishes the lowest and highest intensity levels that a system can represent and, consequently, that an image can have. Closely associated with this concept is image *contrast*, which we define as the difference in intensity between the highest and lowest intensity levels in an image. When an appreciable number of pixels in an image have a high dynamic range, we can expect the image to have high contrast. Conversely, an image with low dynamic range typically has a dull, washed-out gray look. We discuss these concepts in more detail in Chapter 3.

The number, b, of bits required to store a digitized image is

$$b = M \times N \times k \tag{2.4-4}$$

When $M = N$, this equation becomes

$$b = N^2 k \tag{2.4-5}$$

Table 2.1 shows the number of bits required to store square images with various values of N and k. The number of intensity levels corresponding to each value of k is shown in parentheses. When an image can have 2^k intensity levels, it is common practice to refer to the image as a "k-bit image." For example, an image with 256 possible discrete intensity values is called an 8-bit image. Note that storage requirements for 8-bit images of size 1024×1024 and higher are not insignificant.

TABLE 2.1
Number of storage bits for various values of N and k. L is the number of intensity levels.

N/k	1 ($L = 2$)	2 ($L = 4$)	3 ($L = 8$)	4 ($L = 16$)	5 ($L = 32$)	6 ($L = 64$)	7 ($L = 128$)	8 ($L = 256$)
32	1,024	2,048	3,072	4,096	5,120	6,144	7,168	8,192
64	4,096	8,192	12,288	16,384	20,480	24,576	28,672	32,768
128	16,384	32,768	49,152	65,536	81,920	98,304	114,688	131,072
256	65,536	131,072	196,608	262,144	327,680	393,216	458,752	524,288
512	262,144	524,288	786,432	1,048,576	1,310,720	1,572,864	1,835,008	2,097,152
1024	1,048,576	2,097,152	3,145,728	4,194,304	5,242,880	6,291,456	7,340,032	8,388,608
2048	4,194,304	8,388,608	12,582,912	16,777,216	20,971,520	25,165,824	29,369,128	33,554,432
4096	16,777,216	33,554,432	50,331,648	67,108,864	83,886,080	100,663,296	117,440,512	134,217,728
8192	67,108,864	134,217,728	201,326,592	268,435,456	335,544,320	402,653,184	469,762,048	536,870,912

2.4.3 Spatial and Intensity Resolution

Intuitively, spatial resolution is a measure of the smallest discernible detail in an image. Quantitatively, *spatial resolution* can be stated in a number of ways, with *line pairs per unit distance*, and *dots (pixels) per unit distance* being among the most common measures. Suppose that we construct a chart with alternating black and white vertical lines, each of width W units (W can be less than 1). The width of a *line pair* is thus $2W$, and there are $1/2W$ line pairs per unit distance. For example, if the width of a line is 0.1 mm, there are 5 line pairs per unit distance (mm). A widely used definition of image resolution is the largest number of *discernible* line pairs per unit distance (e.g., 100 line pairs per mm). Dots per unit distance is a measure of image resolution used commonly in the printing and publishing industry. In the U.S., this measure usually is expressed as *dots per inch* (dpi). To give you an idea of quality, newspapers are printed with a resolution of 75 dpi, magazines at 133 dpi, glossy brochures at 175 dpi, and the book page at which you are presently looking is printed at 2400 dpi.

The key point in the preceding paragraph is that, to be meaningful, measures of spatial resolution must be stated with respect to spatial units. Image size by itself does not tell the complete story. To say that an image has, say, a resolution 1024×1024 pixels is not a meaningful statement without stating the spatial dimensions encompassed by the image. Size by itself is helpful only in making comparisons between imaging capabilities. For example, a digital camera with a 20-megapixel CCD imaging chip can be expected to have a higher capability to resolve detail than an 8-megapixel camera, assuming that both cameras are equipped with comparable lenses and the comparison images are taken at the same distance.

Intensity resolution similarly refers to the smallest discernible change in intensity level. We have considerable discretion regarding the number of samples used to generate a digital image, but this is not true regarding the number

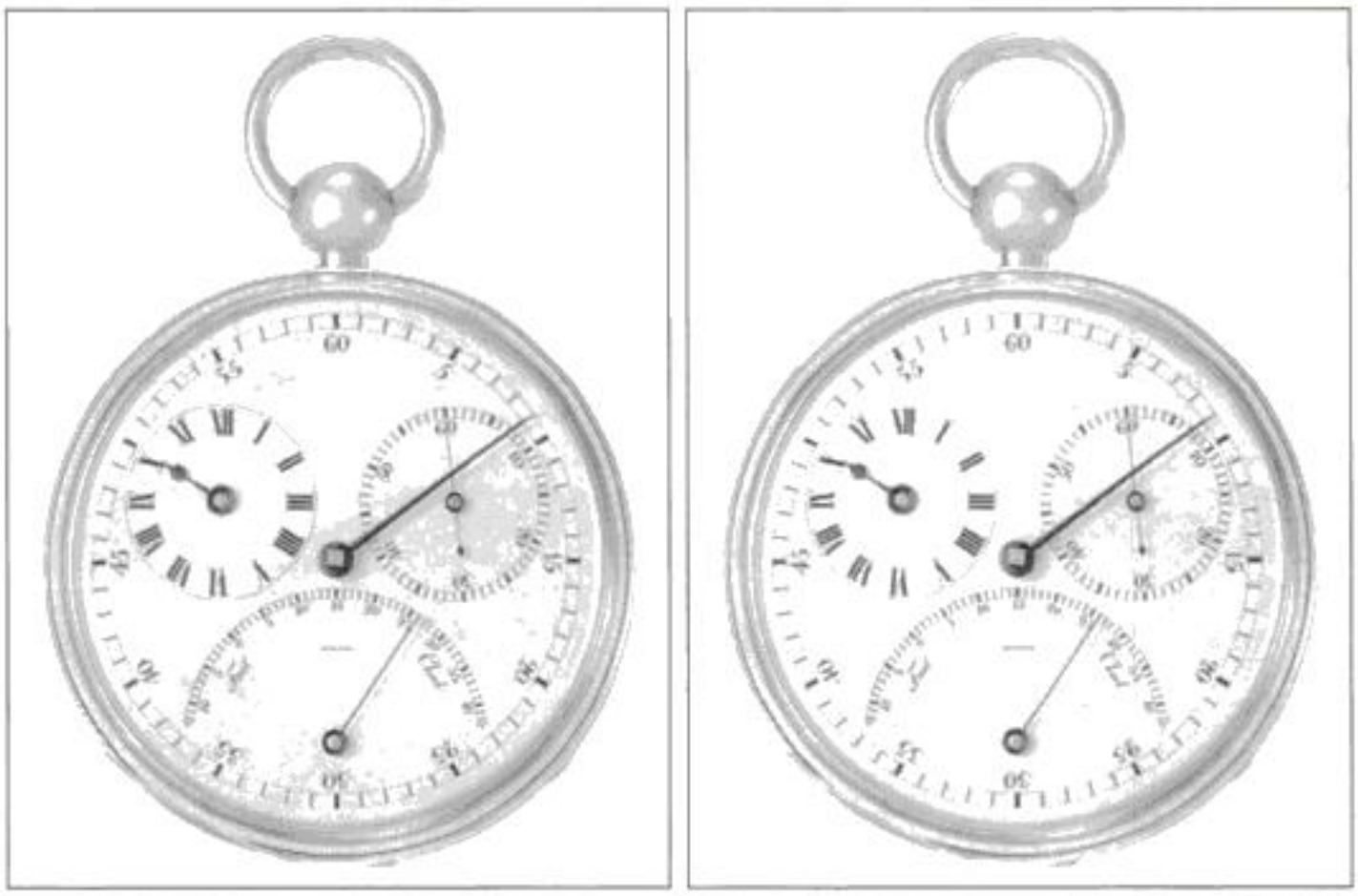
of intensity levels. Based on hardware considerations, the number of intensity levels usually is an integer power of two, as mentioned in the previous section. The most common number is 8 bits, with 16 bits being used in some applications in which enhancement of specific intensity ranges is necessary. Intensity quantization using 32 bits is rare. Sometimes one finds systems that can digitize the intensity levels of an image using 10 or 12 bits, but these are the exception, rather than the rule. Unlike spatial resolution, which must be based on a per unit of distance basis to be meaningful, it is common practice to refer to the number of bits used to quantize intensity as the *intensity resolution*. For example, it is common to say that an image whose intensity is quantized into 256 levels has 8 bits of intensity resolution. Because true discernible changes in intensity are influenced not only by noise and saturation values but also by the capabilities of human perception (see Section 2.1), saying than an image has 8 bits of intensity resolution is nothing more than a statement regarding the ability of an 8-bit system to quantize intensity in fixed increments of 1/256 units of intensity amplitude.

The following two examples illustrate individually the comparative effects of image size and intensity resolution on discernable detail. Later in this section, we discuss how these two parameters interact in determining perceived image quality.

EXAMPLE 2.2: Illustration of the effects of reducing image spatial resolution.

■ Figure 2.20 shows the effects of reducing spatial resolution in an image. The images in Figs. 2.20(a) through (d) are shown in 1250, 300, 150, and 72 dpi, respectively. Naturally, the lower resolution images are smaller than the original. For example, the original image is of size 3692×2812 pixels, but the 72 dpi image is an array of size 213×162. In order to facilitate comparisons, all the smaller images were zoomed back to the original size (the method used for zooming is discussed in Section 2.4.4). This is somewhat equivalent to "getting closer" to the smaller images so that we can make comparable statements about visible details.

There are some small visual differences between Figs. 2.20(a) and (b), the most notable being a slight distortion in the large black needle. For the most part, however, Fig. 2.20(b) is quite acceptable. In fact, 300 dpi is the typical minimum image spatial resolution used for book publishing, so one would not expect to see much difference here. Figure 2.20(c) begins to show visible degradation (see, for example, the round edges of the chronometer and the small needle pointing to 60 on the right side). Figure 2.20(d) shows degradation that is visible in most features of the image. As we discuss in Section 4.5.4, when printing at such low resolutions, the printing and publishing industry uses a number of "tricks" (such as locally varying the pixel size) to produce much better results than those in Fig. 2.20(d). Also, as we show in Section 2.4.4, it is possible to improve on the results of Fig. 2.20 by the choice of interpolation method used. ■



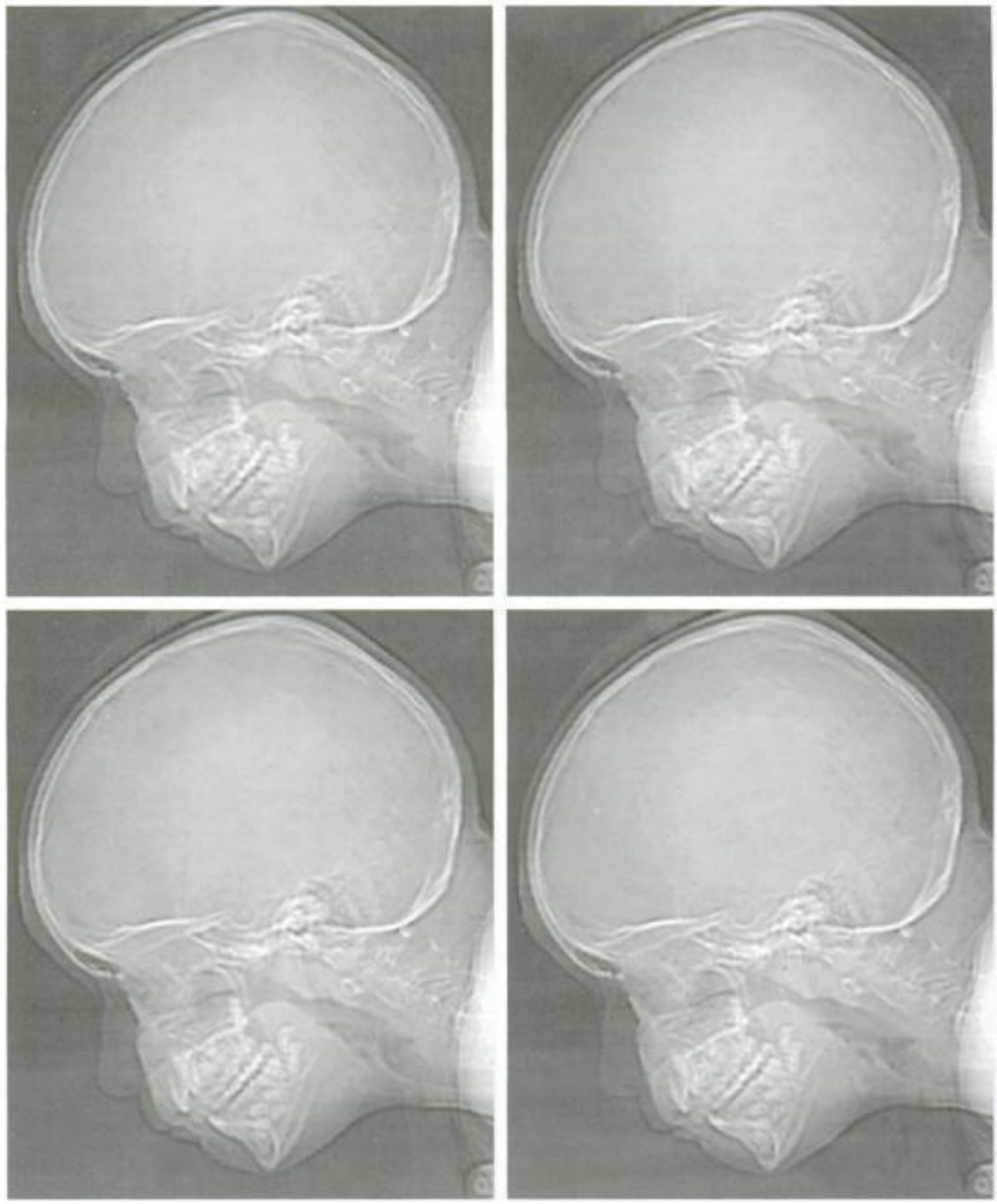
a b
c d

FIGURE 2.20 Typical effects of reducing spatial resolution. Images shown at: (a) 1250 dpi, (b) 300 dpi, (c) 150 dpi, and (d) 72 dpi. The thin black borders were added for clarity. They are not part of the data.

EXAMPLE 2.3: Typical effects of varying the number of intensity levels in a digital image.

■ In this example, we keep the number of samples constant and reduce the number of intensity levels from 256 to 2, in integer powers of 2. Figure 2.21(a) is a 452 × 374 CT projection image, displayed with $k = 8$ (256 intensity levels). Images such as this are obtained by fixing the X-ray source in one position, thus producing a 2-D image in any desired direction. Projection images are used as guides to set up the parameters for a CT scanner, including tilt, number of slices, and range.

Figures 2.21(b) through (h) were obtained by reducing the number of bits from $k = 7$ to $k = 1$ while keeping the image size constant at 452 × 374 pixels. The 256-, 128-, and 64-level images are visually identical for all practical purposes. The 32-level image in Fig. 2.21(d), however, has an imperceptible set of

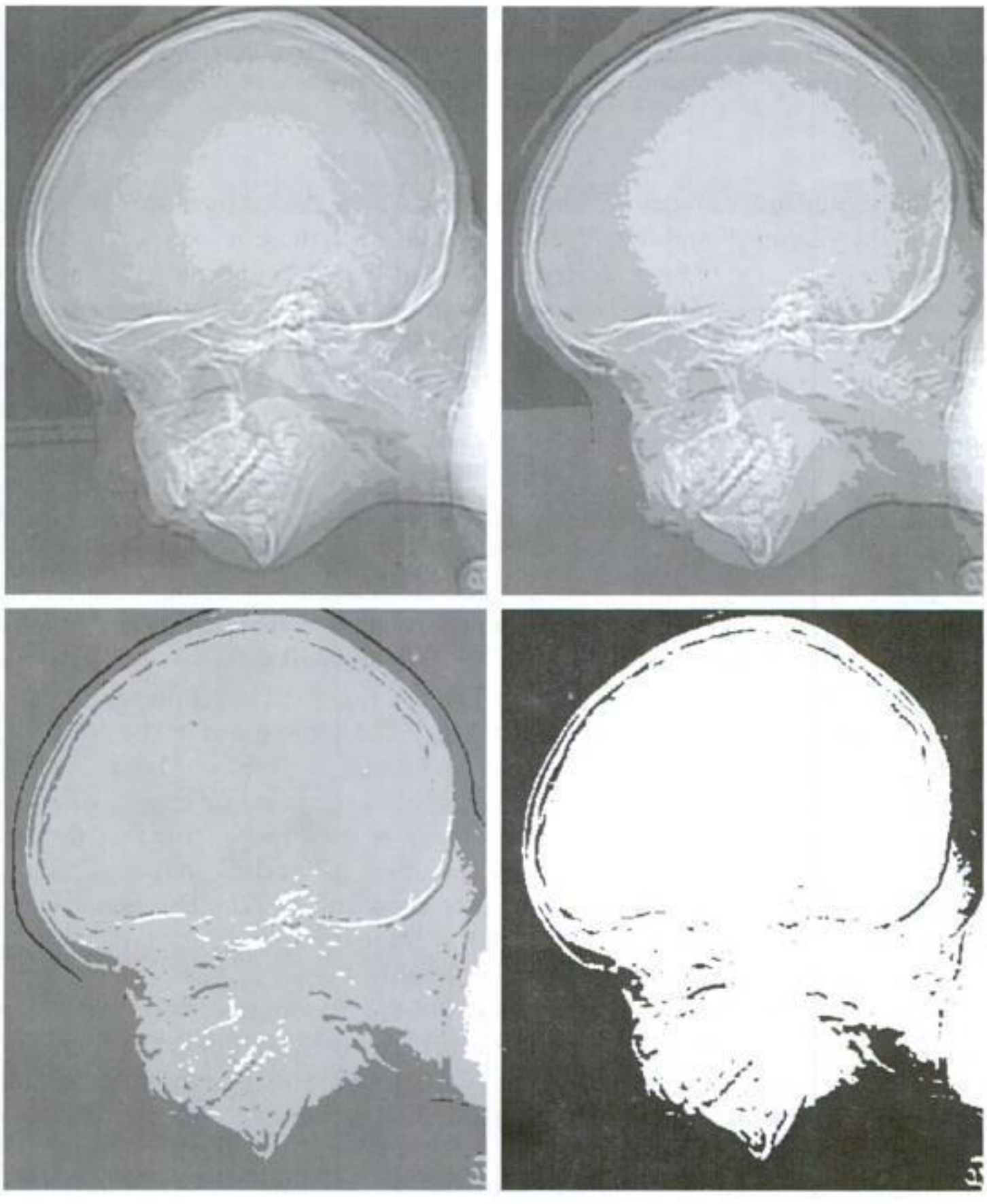


a b
c d

FIGURE 2.21 (a) 452 × 374, 256-level image. (b)–(d) Image displayed in 128, 64, and 32 intensity levels, while keeping the image size constant.

very fine ridge-like structures in areas of constant or nearly constant intensity (particularly in the skull). This effect, caused by the use of an insufficient number of intensity levels in smooth areas of a digital image, is called *false contouring*, so called because the ridges resemble topographic contours in a map. False contouring generally is quite visible in images displayed using 16 or less uniformly spaced intensity levels, as the images in Figs. 2.21(e) through (h) show.

As a very rough rule of thumb, and assuming integer powers of 2 for convenience, images of size 256 × 256 pixels with 64 intensity levels and printed on a size format on the order of 5 × 5 cm are about the lowest spatial and intensity resolution images that can be expected to be reasonably free of objectionable sampling checkerboards and false contouring. ■



e f
g h

FIGURE 2.21
(*Continued*)
(e)–(h) Image displayed in 16, 8, 4, and 2 intensity levels. (Original courtesy of Dr. David R. Pickens, Department of Radiology & Radiological Sciences, Vanderbilt University Medical Center.)

a b c

FIGURE 2.22 (a) Image with a low level of detail. (b) Image with a medium level of detail. (c) Image with a relatively large amount of detail. (Image (b) courtesy of the Massachusetts Institute of Technology.)

The results in Examples 2.2 and 2.3 illustrate the effects produced on image quality by varying N and k independently. However, these results only partially answer the question of how varying N and k affects images because we have not considered yet any relationships that might exist between these two parameters. An early study by Huang [1965] attempted to quantify experimentally the effects on image quality produced by varying N and k simultaneously. The experiment consisted of a set of subjective tests. Images similar to those shown in Fig. 2.22 were used. The woman's face is representative of an image with relatively little detail; the picture of the cameraman contains an intermediate amount of detail; and the crowd picture contains, by comparison, a large amount of detail.

Sets of these three types of images were generated by varying N and k, and observers were then asked to rank them according to their subjective quality. Results were summarized in the form of so-called *isopreference curves* in the Nk-plane. (Figure 2.23 shows average isopreference curves representative of curves corresponding to the images in Fig. 2.22.) Each point in the Nk-plane represents an image having values of N and k equal to the coordinates of that point. Points lying on an isopreference curve correspond to images of equal subjective quality. It was found in the course of the experiments that the isopreference curves tended to shift right and upward, but their shapes in each of the three image categories were similar to those in Fig. 2.23. This is not unexpected, because a shift up and right in the curves simply means larger values for N and k, which implies better picture quality.

The key point of interest in the context of the present discussion is that isopreference curves tend to become more vertical as the detail in the image increases. This result suggests that for images with a large amount of detail only a few intensity levels may be needed. For example, the isopreference curve in Fig. 2.23 corresponding to the crowd is nearly vertical. This indicates that, for a fixed value of N, the perceived quality for this type of image is

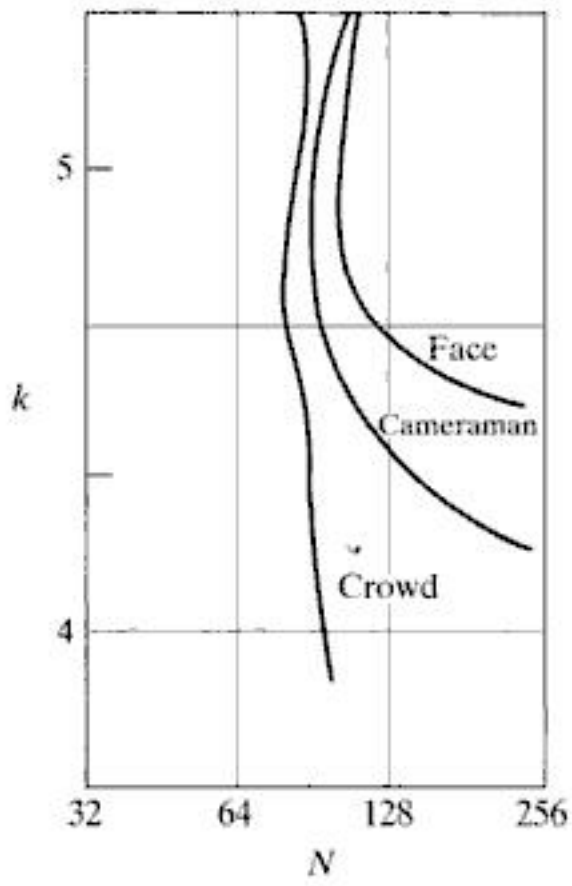


FIGURE 2.23 Typical isopreference curves for the three types of images in Fig. 2.22.

nearly independent of the number of intensity levels used (for the range of intensity levels shown in Fig. 2.23). It is of interest also to note that perceived quality in the other two image categories remained the same in some intervals in which the number of samples was increased, but the number of intensity levels actually decreased. The most likely reason for this result is that a decrease in k tends to increase the apparent contrast, a visual effect that humans often perceive as improved quality in an image.

2.4.4 Image Interpolation

Interpolation is a basic tool used extensively in tasks such as zooming, shrinking, rotating, and geometric corrections. Our principal objective in this section is to introduce interpolation and apply it to image resizing (shrinking and zooming), which are basically image *resampling* methods. Uses of interpolation in applications such as rotation and geometric corrections are discussed in Section 2.6.5. We also return to this topic in Chapter 4, where we discuss image resampling in more detail.

Fundamentally, *interpolation* is the process of using known data to estimate values at unknown locations. We begin the discussion of this topic with a simple example. Suppose that an image of size 500×500 pixels has to be enlarged 1.5 times to 750×750 pixels. A simple way to visualize zooming is to create an imaginary 750×750 grid with the same pixel spacing as the original, and then shrink it so that it fits exactly over the original image. Obviously, the pixel spacing in the shrunken 750×750 grid will be less than the pixel spacing in the original image. To perform intensity-level assignment for any point in the overlay, we look for its closest pixel in the original image and assign the intensity of that pixel to the new pixel in the 750×750 grid. When we are finished assigning intensities to all the points in the overlay grid, we expand it to the original specified size to obtain the zoomed image.

The method just discussed is called *nearest neighbor interpolation* because it assigns to each new location the intensity of its nearest neighbor in the original image (pixel neighborhoods are discussed formally in Section 2.5). This approach is simple but, as we show later in this section, it has the tendency to produce undesirable artifacts, such as severe distortion of straight edges. For this reason, it is used infrequently in practice. A more suitable approach is *bilinear interpolation*, in which we use the four nearest neighbors to estimate the intensity at a given location. Let (x, y) denote the coordinates of the location to which we want to assign an intensity value (think of it as a point of the grid described previously), and let $v(x, y)$ denote that intensity value. For bilinear interpolation, the assigned value is obtained using the equation

$$v(x, y) = ax + by + cxy + d \tag{2.4-6}$$

Contrary to what the name suggests, note that bilinear interpolation is *not* linear because of the xy term.

where the four coefficients are determined from the four equations in four unknowns that can be written using the four nearest neighbors of point (x, y). As you will see shortly, bilinear interpolation gives much better results than nearest neighbor interpolation, with a modest increase in computational burden.

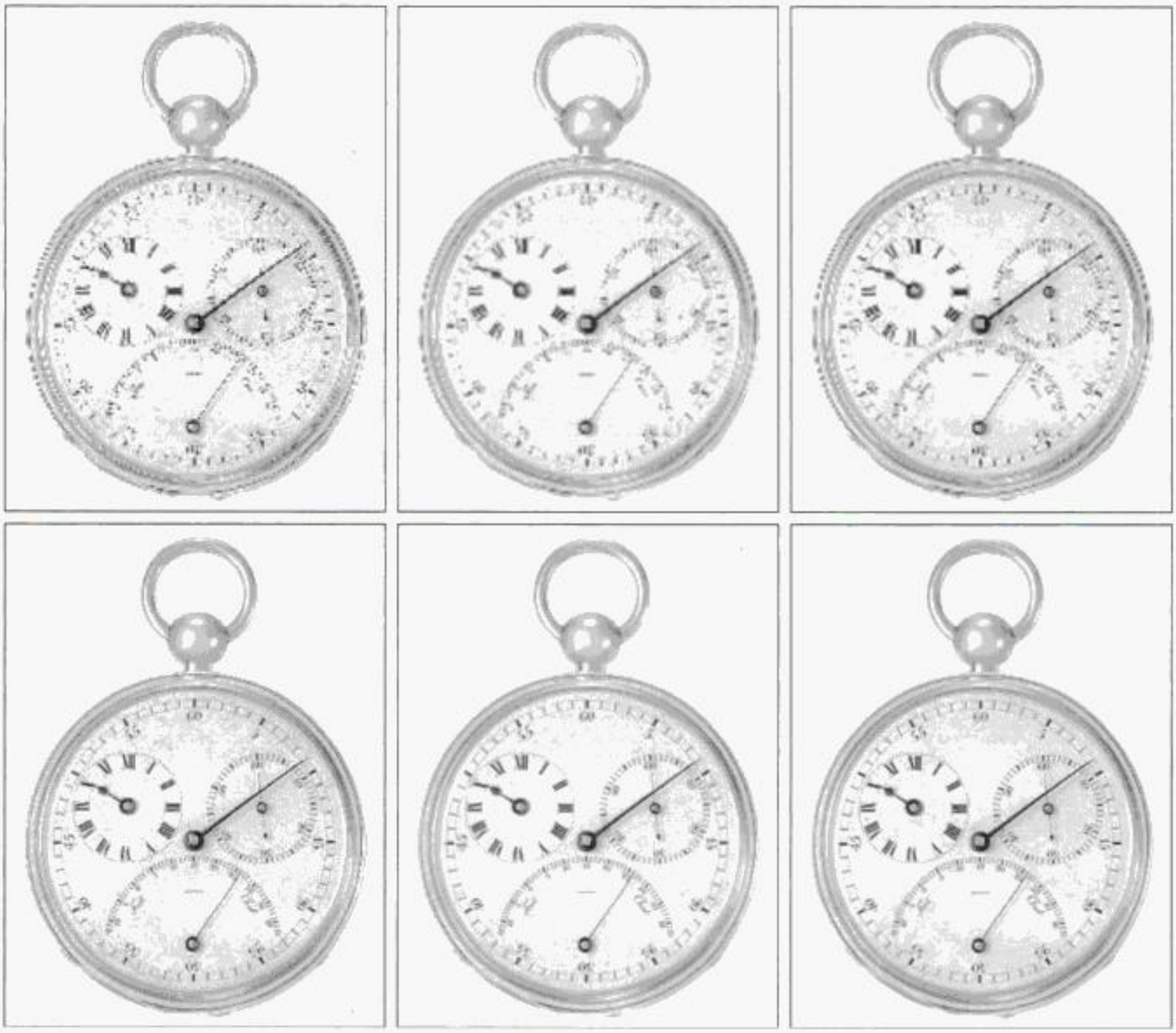
The next level of complexity is *bicubic interpolation*, which involves the sixteen nearest neighbors of a point. The intensity value assigned to point (x, y) is obtained using the equation

$$v(x, y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j \tag{2.4-7}$$

where the sixteen coefficients are determined from the sixteen equations in sixteen unknowns that can be written using the sixteen nearest neighbors of point (x, y). Observe that Eq. (2.4-7) reduces in form to Eq. (2.4-6) if the limits of both summations in the former equation are 0 to 1. Generally, bicubic interpolation does a better job of preserving fine detail than its bilinear counterpart. Bicubic interpolation is the standard used in commercial image editing programs, such as Adobe Photoshop and Corel Photopaint.

EXAMPLE 2.4: Comparison of interpolation approaches for image shrinking and zooming.

Figure 2.24(a) is the same image as Fig. 2.20(d), which was obtained by reducing the resolution of the 1250 dpi image in Fig. 2.20(a) to 72 dpi (the size shrank from the original size of 3692 × 2812 to 213 × 162 pixels) and then zooming the reduced image back to its original size. To generate Fig. 2.20(d) we used nearest neighbor interpolation both to shrink and zoom the image. As we commented before, the result in Fig. 2.24(a) is rather poor. Figures 2.24(b) and (c) are the results of repeating the same procedure but using, respectively, bilinear and bicubic interpolation for both shrinking and zooming. The result obtained by using bilinear interpolation is a significant improvement over nearest neighbor interpolation. The bicubic result is slightly sharper than the bilinear image. Figure 2.24(d) is the same as Fig. 2.20(c), which was obtained using nearest neighbor interpolation for both shrinking and zooming. We commented in discussing that figure that reducing the resolution to 150 dpi began showing degradation in the image. Figures 2.24(e) and (f) show the results of using



a b c
d e f

FIGURE 2.24 (a) Image reduced to 72 dpi and zoomed back to its original size (3692 × 2812 pixels) using nearest neighbor interpolation. This figure is the same as Fig. 2.20(d). (b) Image shrunk and zoomed using bilinear interpolation. (c) Same as (b) but using bicubic interpolation. (d)–(f) Same sequence, but shrinking down to 150 dpi instead of 72 dpi [Fig. 2.24(d) is the same as Fig. 2.20(c)]. Compare Figs. 2.24(e) and (f), especially the latter, with the original image in Fig. 2.20(a).

bilinear and bicubic interpolation, respectively, to shrink and zoom the image. In spite of a reduction in resolution from 1250 to 150, these last two images compare reasonably favorably with the original, showing once again the power of these two interpolation methods. As before, bicubic interpolation yielded slightly sharper results. ■

It is possible to use more neighbors in interpolation, and there are more complex techniques, such as using splines and wavelets, that in some instances can yield better results than the methods just discussed. While preserving fine detail is an exceptionally important consideration in image generation for 3-D graphics (Watt [1993], Shirley [2002]) and in medical image processing (Lehmann et al. [1999]), the extra computational burden seldom is justifiable for general-purpose digital image processing, where bilinear or bicubic interpolation typically are the methods of choice.

2.5 Some Basic Relationships between Pixels

In this section, we consider several important relationships between pixels in a digital image. As mentioned before, an image is denoted by $f(x, y)$. When referring in this section to a particular pixel, we use lowercase letters, such as p and q.

2.5.1 Neighbors of a Pixel

A pixel p at coordinates (x, y) has four *horizontal* and *vertical* neighbors whose coordinates are given by

$$(x + 1, y), (x - 1, y), (x, y + 1), (x, y - 1)$$

This set of pixels, called the 4-*neighbors* of p, is denoted by $N_4(p)$. Each pixel is a unit distance from (x, y), and some of the neighbor locations of p lie outside the digital image if (x, y) is on the border of the image. We deal with this issue in Chapter 3.

The four *diagonal* neighbors of p have coordinates

$$(x + 1, y + 1), (x + 1, y - 1), (x - 1, y + 1), (x - 1, y - 1)$$

and are denoted by $N_D(p)$. These points, together with the 4-neighbors, are called the 8-*neighbors* of p, denoted by $N_8(p)$. As before, some of the neighbor locations in $N_D(p)$ and $N_8(p)$ fall outside the image if (x, y) is on the border of the image.

2.5.2 Adjacency, Connectivity, Regions, and Boundaries

Let V be the set of intensity values used to define adjacency. In a binary image, $V = \{1\}$ if we are referring to adjacency of pixels with value 1. In a gray-scale image, the idea is the same, but set V typically contains more elements. For example, in the adjacency of pixels with a range of possible intensity values 0 to 255, set V could be any subset of these 256 values. We consider three types of adjacency:

We use the symbols $\cap$ and $\cup$ to denote set intersection and union, respectively. Given sets A and B, recall that their *intersection* is the set of elements that are members of both A and B. The *union* of these two sets is the set of elements that are members of A, of B, or of both. We discuss sets in more detail in Section 2.6.4.

(a) 4-*adjacency*. Two pixels p and q with values from V are 4-adjacent if q is in the set $N_4(p)$.

(b) 8-*adjacency*. Two pixels p and q with values from V are 8-adjacent if q is in the set $N_8(p)$.

(c) *m-adjacency* (mixed adjacency). Two pixels p and q with values from V are m-adjacent if

- **(i)** q is in $N_4(p)$, *or*
- **(ii)** q is in $N_D(p)$ *and* the set $N_4(p) \cap N_4(q)$ has no pixels whose values are from V.

Mixed adjacency is a modification of 8-adjacency. It is introduced to eliminate the ambiguities that often arise when 8-adjacency is used. For example, consider the pixel arrangement shown in Fig. 2.25(a) for $V = \{1\}$. The three pixels at the top of Fig. 2.25(b) show multiple (ambiguous) 8-adjacency, as indicated by the dashed lines. This ambiguity is removed by using m-adjacency, as shown in Fig. 2.25(c).

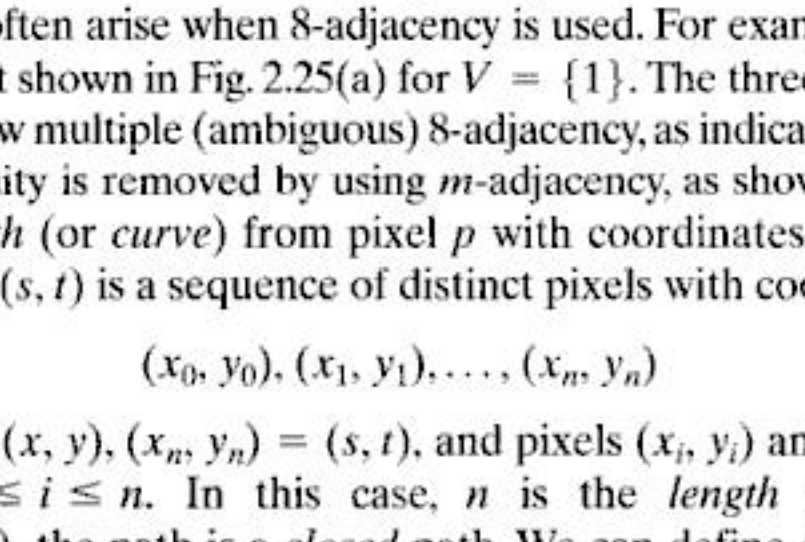
A (*digital*) *path* (or *curve*) from pixel p with coordinates (x, y) to pixel q with coordinates (s, t) is a sequence of distinct pixels with coordinates

$$(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$$

where $(x_0, y_0) = (x, y)$, $(x_n, y_n) = (s, t)$, and pixels (x_i, y_i) and (x_{i-1}, y_{i-1}) are adjacent for $1 \leq i \leq n$. In this case, n is the *length* of the path. If $(x_0, y_0) = (x_n, y_n)$, the path is a *closed* path. We can define 4-, 8-, or m-paths depending on the type of adjacency specified. For example, the paths shown in Fig. 2.25(b) between the top right and bottom right points are 8-paths, and the path in Fig. 2.25(c) is an m-path.

Let S represent a subset of pixels in an image. Two pixels p and q are said to be *connected* in S if there exists a path between them consisting entirely of pixels in S. For any pixel p in S, the *set* of pixels that are connected to it in S is called a *connected component* of S. If it only has one connected component, then set S is called a *connected set*.

Let R be a subset of pixels in an image. We call R a *region* of the image if R is a connected set. Two regions, R_i and R_j are said to be *adjacent* if their union forms a connected set. Regions that are not adjacent are said to be *disjoint*. We consider 4- and 8-adjacency when referring to regions. For our definition to make sense, the type of adjacency used must be specified. For example, the two regions (of 1s) in Fig. 2.25(d) are adjacent only if 8-adjacency is used (according to the definition in the previous paragraph, a 4-path between the two regions does not exist, so their union is not a connected set).



a b c
d e f

FIGURE 2.25 (a) An arrangement of pixels. (b) Pixels that are 8-adjacent (adjacency is shown by dashed lines; note the ambiguity). (c) m-adjacency. (d) Two regions (of 1s) that are adjacent if 8-adjecency is used. (e) The circled point is part of the boundary of the 1-valued pixels only if 8-adjacency between the region and background is used. (f) The inner boundary of the 1-valued region does not form a closed path, but its outer boundary does.

Suppose that an image contains K disjoint regions, R_k, $k = 1, 2, \ldots, K$, none of which touches the image border.[†] Let R_u denote the union of all the K regions, and let $(R_u)^c$ denote its complement (recall that the *complement* of a set S is the set of points that are not in S). We call all the points in R_u the *foreground*, and all the points in $(R_u)^c$ the *background* of the image.

The *boundary* (also called the *border* or *contour*) of a region R is the set of points that are adjacent to points in the complement of R. Said another way, the border of a region is the set of pixels in the region that have at least one background neighbor. Here again, we must specify the connectivity being used to define adjacency. For example, the point circled in Fig. 2.25(e) is not a member of the border of the 1-valued region if 4-connectivity is used between the region and its background. As a rule, adjacency between points in a region and its background is defined in terms of 8-connectivity to handle situations like this.

The preceding definition sometimes is referred to as the *inner border* of the region to distinguish it from its *outer border*, which is the corresponding border in the background. This distinction is important in the development of border-following algorithms. Such algorithms usually are formulated to follow the outer boundary in order to guarantee that the result will form a closed path. For instance, the inner border of the 1-valued region in Fig. 2.25(f) is the region itself. This border does not satisfy the definition of a closed path given earlier. On the other hand, the outer border of the region does form a closed path around the region.

If R happens to be an entire image (which we recall is a rectangular set of pixels), then its boundary is defined as the set of pixels in the first and last rows and columns of the image. This extra definition is required because an image has no neighbors beyond its border. Normally, when we refer to a region, we are referring to a subset of an image, and any pixels in the boundary of the region that happen to coincide with the border of the image are included implicitly as part of the region boundary.

The concept of an *edge* is found frequently in discussions dealing with regions and boundaries. There is a key difference between these concepts, however. The boundary of a finite region forms a closed path and is thus a "global" concept. As discussed in detail in Chapter 10, edges are formed from pixels with derivative values that exceed a preset threshold. Thus, the idea of an edge is a "local" concept that is based on a measure of intensity-level discontinuity at a point. It is possible to link edge points into edge segments, and sometimes these segments are linked in such a way that they correspond to boundaries, but this is not always the case. The one exception in which edges and boundaries correspond is in binary images. Depending on the type of connectivity and edge operators used (we discuss these in Chapter 10), the edge extracted from a binary region will be the same as the region boundary.

[†]We make this assumption to avoid having to deal with special cases. This is done without loss of generality because if one or more regions touch the border of an image, we can simply pad the image with a 1-pixel-wide border of background values.

This is intuitive. Conceptually, until we arrive at Chapter 10, it is helpful to think of edges as intensity discontinuities and boundaries as closed paths.

2.5.3 Distance Measures

For pixels p, q, and z, with coordinates (x, y), (s, t), and (v, w), respectively, D is a *distance function* or *metric* if

(a) $D(p, q) \geq 0 \quad (D(p, q) = 0 \quad \text{iff} \quad p = q)$,

(b) $D(p, q) = D(q, p)$, and

(c) $D(p, z) \leq D(p, q) + D(q, z)$.

The *Euclidean distance* between p and q is defined as

$$D_e(p, q) = \left[(x - s)^2 + (y - t)^2\right]^{\frac{1}{2}} \tag{2.5-1}$$

For this distance measure, the pixels having a distance less than or equal to some value r from (x, y) are the points contained in a disk of radius r centered at (x, y).

The D_4 *distance* (called the *city-block distance*) between p and q is defined as

$$D_4(p, q) = |x - s| + |y - t| \tag{2.5-2}$$

In this case, the pixels having a D_4 distance from (x, y) less than or equal to some value r form a diamond centered at (x, y). For example, the pixels with D_4 distance ≤ 2 from (x, y) (the center point) form the following contours of constant distance:

$$\begin{array}{ccccc} & & 2 & & \\ & 2 & 1 & 2 & \\ 2 & 1 & 0 & 1 & 2 \\ & 2 & 1 & 2 & \\ & & 2 & & \end{array}$$

The pixels with $D_4 = 1$ are the 4-neighbors of (x, y).

The D_8 *distance* (called the *chessboard distance*) between p and q is defined as

$$D_8(p, q) = \max(|x - s|, |y - t|) \tag{2.5-3}$$

In this case, the pixels with D_8 distance from (x, y) less than or equal to some value r form a square centered at (x, y). For example, the pixels with D_8 distance ≤ 2 from (x, y) (the center point) form the following contours of constant distance:

$$\begin{array}{ccccc} 2 & 2 & 2 & 2 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{array}$$

The pixels with $D_8 = 1$ are the 8-neighbors of (x, y).

Note that the D_4 and D_8 distances between p and q are independent of any paths that might exist between the points because these distances involve only the coordinates of the points. If we elect to consider m-adjacency, however, the D_m distance between two points is defined as the shortest m-path between the points. In this case, the distance between two pixels will depend on the values of the pixels along the path, as well as the values of their neighbors. For instance, consider the following arrangement of pixels and assume that p, p_2, and p_4 have value 1 and that p_1 and p_3 can have a value of 0 or 1:

$$\begin{matrix} & p_3 & p_4 \\ p_1 & p_2 & \\ p & & \end{matrix}$$

Suppose that we consider adjacency of pixels valued 1 (i.e., $V = \{1\}$). If p_1 and p_3 are 0, the length of the shortest m-path (the D_m distance) between p and p_4 is 2. If p_1 is 1, then p_2 and p will no longer be m-adjacent (see the definition of m-adjacency) and the length of the shortest m-path becomes 3 (the path goes through the points $pp_1p_2p_4$). Similar comments apply if p_3 is 1 (and p_1 is 0); in this case, the length of the shortest m-path also is 3. Finally, if both p_1 and p_3 are 1, the length of the shortest m-path between p and p_4 is 4. In this case, the path goes through the sequence of points $pp_1p_2p_3p_4$.

2.6 An Introduction to the Mathematical Tools Used in Digital Image Processing

Before proceeding, you may find it helpful to download and study the review material available in the Tutorials section of the book Web site. The review covers introductory material on matrices and vectors, linear systems, set theory, and probability.

This section has two principal objectives: (1) to introduce you to the various mathematical tools we use throughout the book; and (2) to help you begin developing a "feel" for how these tools are used by applying them to a variety of basic image-processing tasks, some of which will be used numerous times in subsequent discussions. We expand the scope of the tools and their application as necessary in the following chapters.

2.6.1 Array versus Matrix Operations

An *array* operation involving one or more images is carried out on a *pixel-by-pixel* basis. We mentioned earlier in this chapter that images can be viewed equivalently as matrices. In fact, there are many situations in which operations between images are carried out using matrix theory (see Section 2.6.6). It is for this reason that a clear distinction must be made between array and matrix operations. For example, consider the following 2×2 images:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

The *array product* of these two images is

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} \\ a_{21}b_{21} & a_{22}b_{22} \end{bmatrix}$$

On the other hand, the *matrix product* is given by

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} & a_{11}b_{12} + a_{12}b_{22} \\ a_{21}b_{11} + a_{22}b_{21} & a_{21}b_{12} + a_{22}b_{22} \end{bmatrix}$$

We assume array operations throughout the book, unless stated otherwise. For example, when we refer to raising an image to a power, we mean that each individual pixel is raised to that power; when we refer to dividing an image by another, we mean that the division is between corresponding pixel pairs, and so on.

2.6.2 Linear versus Nonlinear Operations

One of the most important classifications of an image-processing method is whether it is linear or nonlinear. Consider a general operator, H, that produces an output image, $g(x, y)$, for a given input image, $f(x, y)$:

$$H\big[f(x, y)\big] = g(x, y) \tag{2.6-1}$$

H is said to be a *linear operator* if

$$\begin{aligned} H\big[a_i f_i(x, y) + a_j f_j(x, y)\big] &= a_i H\big[f_i(x, y)\big] + a_j H\big[f_j(x, y)\big] \\ &= a_i g_i(x, y) + a_j g_j(x, y) \end{aligned} \tag{2.6-2}$$

where a_i, a_j, $f_i(x, y)$, and $f_j(x, y)$ are arbitrary constants and images (of the same size), respectively. Equation (2.6-2) indicates that the output of a linear operation due to the sum of two inputs is the same as performing the operation on the inputs individually and then summing the results. In addition, the output of a linear operation to a constant times an input is the same as the output of the operation due to the original input multiplied by that constant. The first property is called the property of *additivity* and the second is called the property of *homogeneity*.

As a simple example, suppose that H is the sum operator, Σ; that is, the function of this operator is simply to sum its inputs. To test for linearity, we start with the left side of Eq. (2.6-2) and attempt to prove that it is equal to the right side:

$$\begin{aligned} \sum\big[a_i f_i(x, y) + a_j f_j(x, y)\big] &= \sum a_i f_i(x, y) + \sum a_j f_j(x, y) \\ &= a_i \sum f_i(x, y) + a_j \sum f_j(x, y) \\ &= a_i g_i(x, y) + a_j g_j(x, y) \end{aligned}$$

These are array summations, not the sums of all the elements of the images. As such, the sum of a single image is the image itself.

where the first step follows from the fact that summation is distributive. So, an expansion of the left side is equal to the right side of Eq. (2.6-2), and we conclude that the sum operator is linear.

On the other hand, consider the max operation, whose function is to find the maximum value of the pixels in an image. For our purposes here, the simplest way to prove that this operator is nonlinear, is to find an example that fails the test in Eq. (2.6-2). Consider the following two images

$$f_1 = \begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix} \quad \text{and} \quad f_2 = \begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}$$

and suppose that we let $a_1 = 1$ and $a_2 = -1$. To test for linearity, we again start with the left side of Eq. (2.6-2):

$$\max\left\{(1)\begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix} + (-1)\begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}\right\} = \max\left\{\begin{bmatrix} -6 & -3 \\ -2 & -4 \end{bmatrix}\right\}$$
$$= -2$$

Working next with the right side, we obtain

$$(1)\max\left\{\begin{bmatrix} 0 & 2 \\ 2 & 3 \end{bmatrix}\right\} + (-1)\max\left\{\begin{bmatrix} 6 & 5 \\ 4 & 7 \end{bmatrix}\right\} = 3 + (-1)7$$
$$= -4$$

The left and right sides of Eq. (2.6-2) are not equal in this case, so we have proved that in general the max operator is nonlinear.

As you will see in the next three chapters, especially in Chapters 4 and 5, linear operations are exceptionally important because they are based on a large body of theoretical and practical results that are applicable to image processing. Nonlinear systems are not nearly as well understood, so their scope of application is more limited. However, you will encounter in the following chapters several nonlinear image processing operations whose performance far exceeds what is achievable by their linear counterparts.

2.6.3 Arithmetic Operations

Arithmetic operations between images are array operations which, as discussed in Section 2.6.1, means that arithmetic operations are carried out between corresponding pixel pairs. The four arithmetic operations are denoted as

$$\begin{aligned} s(x, y) &= f(x, y) + g(x, y) \\ d(x, y) &= f(x, y) - g(x, y) \\ p(x, y) &= f(x, y) \times g(x, y) \\ v(x, y) &= f(x, y) \div g(x, y) \end{aligned} \tag{2.6-3}$$

It is understood that the operations are performed between corresponding pixel pairs in f and g for $x = 0, 1, 2, \ldots, M - 1$ and $y = 0, 1, 2, \ldots, N - 1$

where, as usual, M and N are the row and column sizes of the images. Clearly, s, d, p, and v are images of size $M \times N$ also. Note that image arithmetic in the manner just defined involves images of the same size. The following examples are indicative of the important role played by arithmetic operations in digital image processing.

EXAMPLE 2.5: Addition (averaging) of noisy images for noise reduction.

■ Let $g(x, y)$ denote a corrupted image formed by the addition of noise, $\eta(x, y)$, to a noiseless image $f(x, y)$; that is,

$$g(x, y) = f(x, y) + \eta(x, y) \tag{2.6-4}$$

where the assumption is that at every pair of coordinates (x, y) the noise is uncorrelated[†] and has zero average value. The objective of the following procedure is to reduce the noise content by adding a set of noisy images, $\{g_i(x, y)\}$. This is a technique used frequently for image enhancement.

If the noise satisfies the constraints just stated, it can be shown (Problem 2.20) that if an image $\bar{g}(x, y)$ is formed by averaging K different noisy images,

$$\bar{g}(x, y) = \frac{1}{K}\sum_{i=1}^{K} g_i(x, y) \tag{2.6-5}$$

then it follows that

$$E\{\bar{g}(x, y)\} = f(x, y) \tag{2.6-6}$$

and

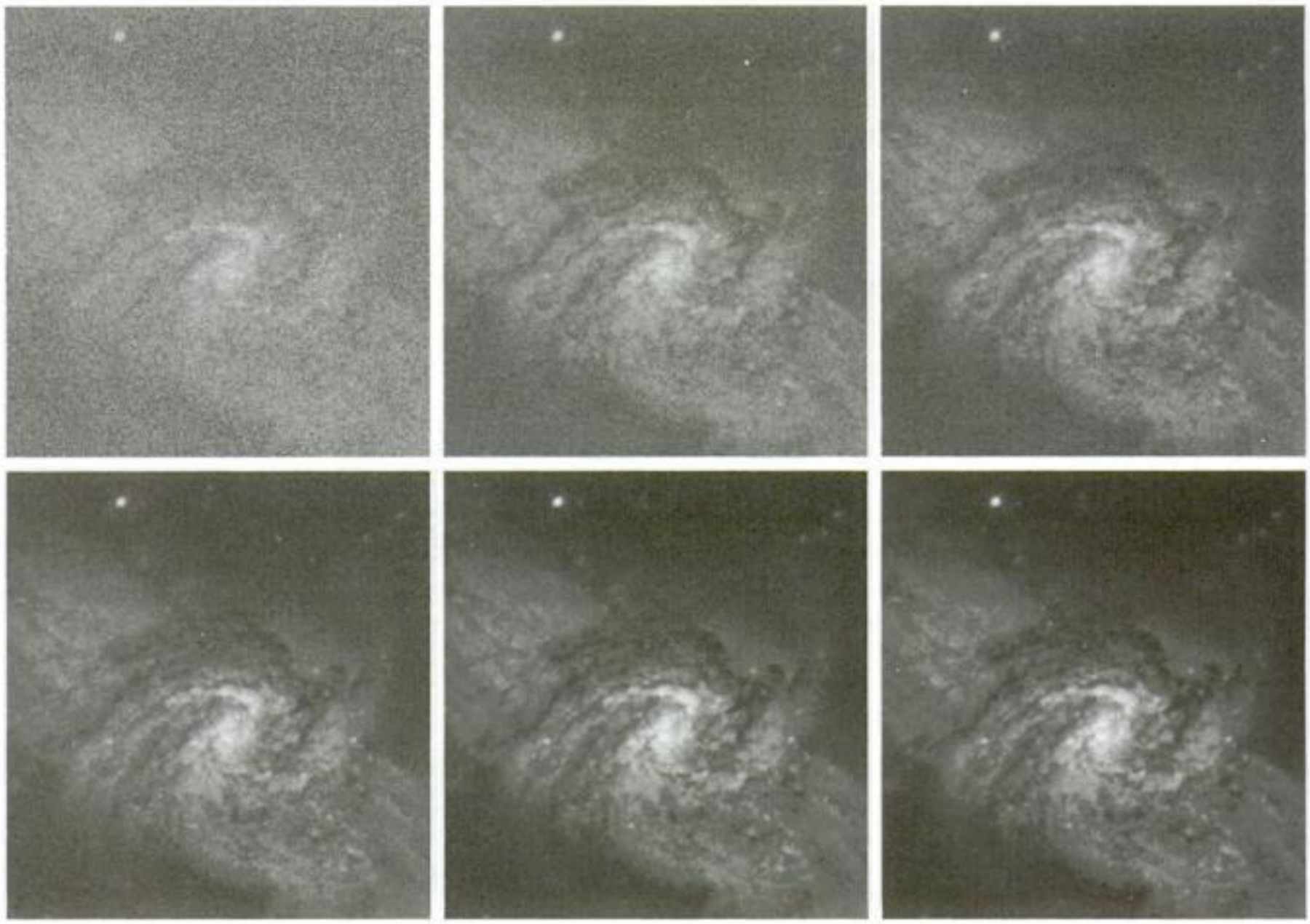
$$\sigma^2_{\bar{g}(x,y)} = \frac{1}{K}\sigma^2_{\eta(x,y)} \tag{2.6-7}$$

where $E\{\bar{g}(x, y)\}$ is the expected value of $\bar{g}$, and $\sigma^2_{\bar{g}(x,y)}$ and $\sigma^2_{\eta(x,y)}$ are the variances of $\bar{g}$ and η, respectively, all at coordinates (x, y). The standard deviation (square root of the variance) at any point in the average image is

$$\sigma_{\bar{g}(x,y)} = \frac{1}{\sqrt{K}}\sigma_{\eta(x,y)} \tag{2.6-8}$$

As K increases, Eqs. (2.6-7) and (2.6-8) indicate that the variability (as measured by the variance or the standard deviation) of the pixel values at each location (x, y) decreases. Because $E\{\bar{g}(x, y)\} = f(x, y)$, this means that $\bar{g}(x, y)$ approaches $f(x, y)$ as the number of noisy images used in the averaging process increases. In practice, the images $g_i(x, y)$ must be *registered* (aligned) in order to avoid the introduction of blurring and other artifacts in the output image.

[†]Recall that the variance of a random variable z with mean m is defined as $E[(z - m)^2]$, where $E\{\cdot\}$ is the expected value of the argument. The covariance of two random variables z_i and z_j is defined as $E[(z_i - m_i)(z_j - m_j)]$. If the variables are *uncorrelated*, their covariance is 0.



a b c
d e f

FIGURE 2.26 (a) Image of Galaxy Pair NGC 3314 corrupted by additive Gaussian noise. (b)–(f) Results of averaging 5, 10, 20, 50, and 100 noisy images, respectively. (Original image courtesy of NASA.)

The images shown in this example are from a galaxy pair called NGC 3314, taken by NASA's Hubble Space Telescope. NGC 3314 lies about 140 million light-years from Earth, in the direction of the southern-hemisphere constellation Hydra. The bright stars forming a pinwheel shape near the center of the front galaxy were formed from interstellar gas and dust.

An important application of image averaging is in the field of astronomy, where imaging under very low light levels frequently causes sensor noise to render single images virtually useless for analysis. Figure 2.26(a) shows an 8-bit image in which corruption was simulated by adding to it Gaussian noise with zero mean and a standard deviation of 64 intensity levels. This image, typical of noisy images taken under low light conditions, is useless for all practical purposes. Figures 2.26(b) through (f) show the results of averaging 5, 10, 20, 50, and 100 images, respectively. We see that the result in Fig. 2.26(e), obtained with $K = 50$, is reasonably clean. The image Fig. 2.26(f), resulting from averaging 100 noisy images, is only a slight improvement over the image in Fig. 2.26(e).

Addition is a discrete version of continuous integration. In astronomical observations, a process equivalent to the method just described is to use the integrating capabilities of CCD (see Section 2.3.3) or similar sensors for noise reduction by observing the same scene over long periods of time. Cooling also is used to reduce sensor noise. The net effect, however, is analogous to averaging a set of noisy digital images. ■

EXAMPLE 2.6: Image subtraction for enhancing differences.

■ A frequent application of image subtraction is in the enhancement of *differences* between images. For example, the image in Fig. 2.27(b) was obtained by setting to zero the least-significant bit of every pixel in Fig. 2.27(a). Visually, these images are indistinguishable. However, as Fig. 2.27(c) shows, subtracting one image from the other clearly shows their differences. Black (0) values in this difference image indicate locations where there is no difference between the images in Figs. 2.27(a) and (b).

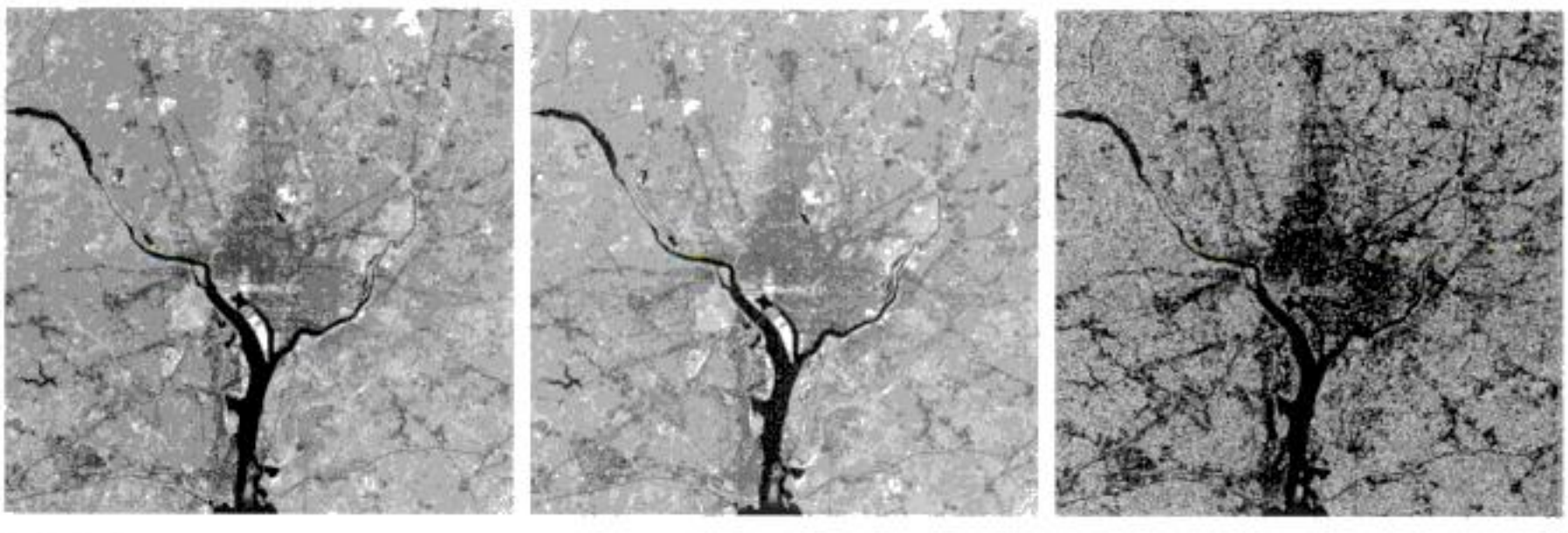
As another illustration, we discuss briefly an area of medical imaging called *mask mode radiography*, a commercially successful and highly beneficial use of image subtraction. Consider image differences of the form

$$g(x, y) = f(x, y) - h(x, y) \tag{2.6-9}$$

Change detection via image subtraction is used also in image segmentation, which is the topic of Chapter 10.

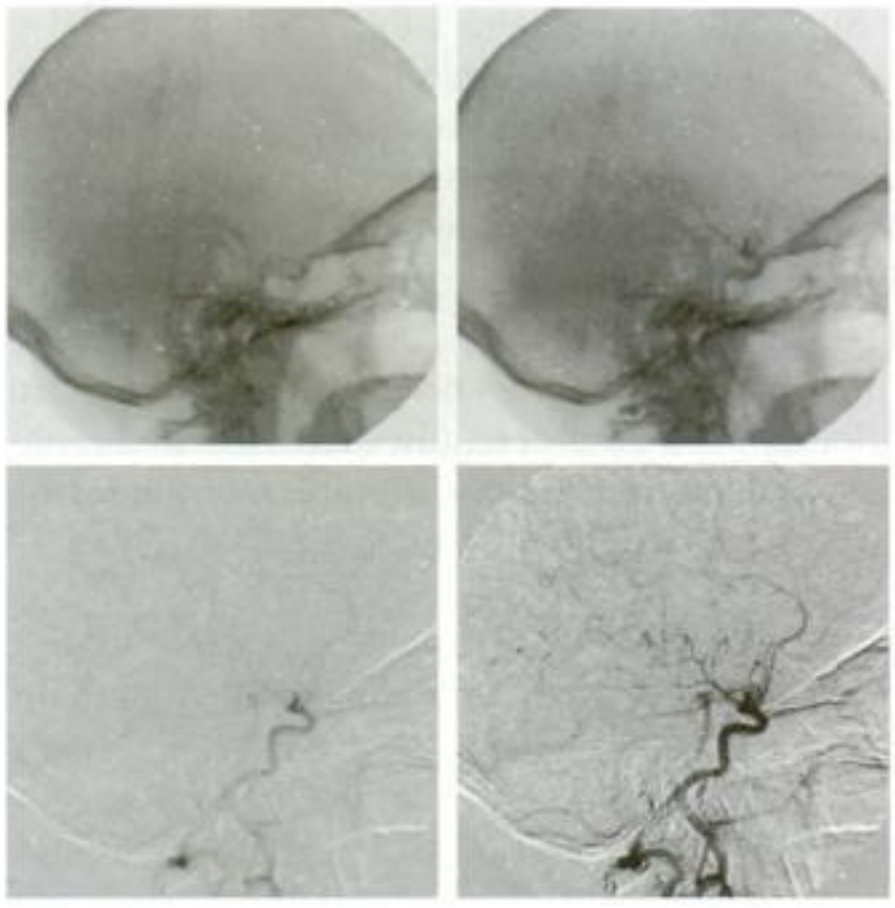
In this case $h(x, y)$, the *mask*, is an X-ray image of a region of a patient's body captured by an intensified TV camera (instead of traditional X-ray film) located opposite an X-ray source. The procedure consists of injecting an X-ray contrast medium into the patient's bloodstream, taking a series of images called *live images* [samples of which are denoted as $f(x, y)$] of the same anatomical region as $h(x, y)$, and subtracting the mask from the series of incoming live images after injection of the contrast medium. The net effect of subtracting the mask from each sample live image is that the areas that are different between $f(x, y)$ and $h(x, y)$ appear in the output image, $g(x, y)$, as enhanced detail. Because images can be captured at TV rates, this procedure in essence gives a movie showing how the contrast medium propagates through the various arteries in the area being observed.

Figure 2.28(a) shows a mask X-ray image of the top of a patient's head prior to injection of an iodine medium into the bloodstream, and Fig. 2.28(b) is a sample of a live image taken after the medium was injected. Figure 2.28(c) is



a b c

FIGURE 2.27 (a) Infrared image of the Washington, D.C. area. (b) Image obtained by setting to zero the least significant bit of every pixel in (a). (c) Difference of the two images, scaled to the range [0, 255] for clarity.



a b
c d

FIGURE 2.28
Digital subtraction angiography. (a) Mask image. (b) A live image. (c) Difference between (a) and (b). (d) Enhanced difference image. (Figures (a) and (b) courtesy of The Image Sciences Institute, University Medical Center, Utrecht, The Netherlands.)

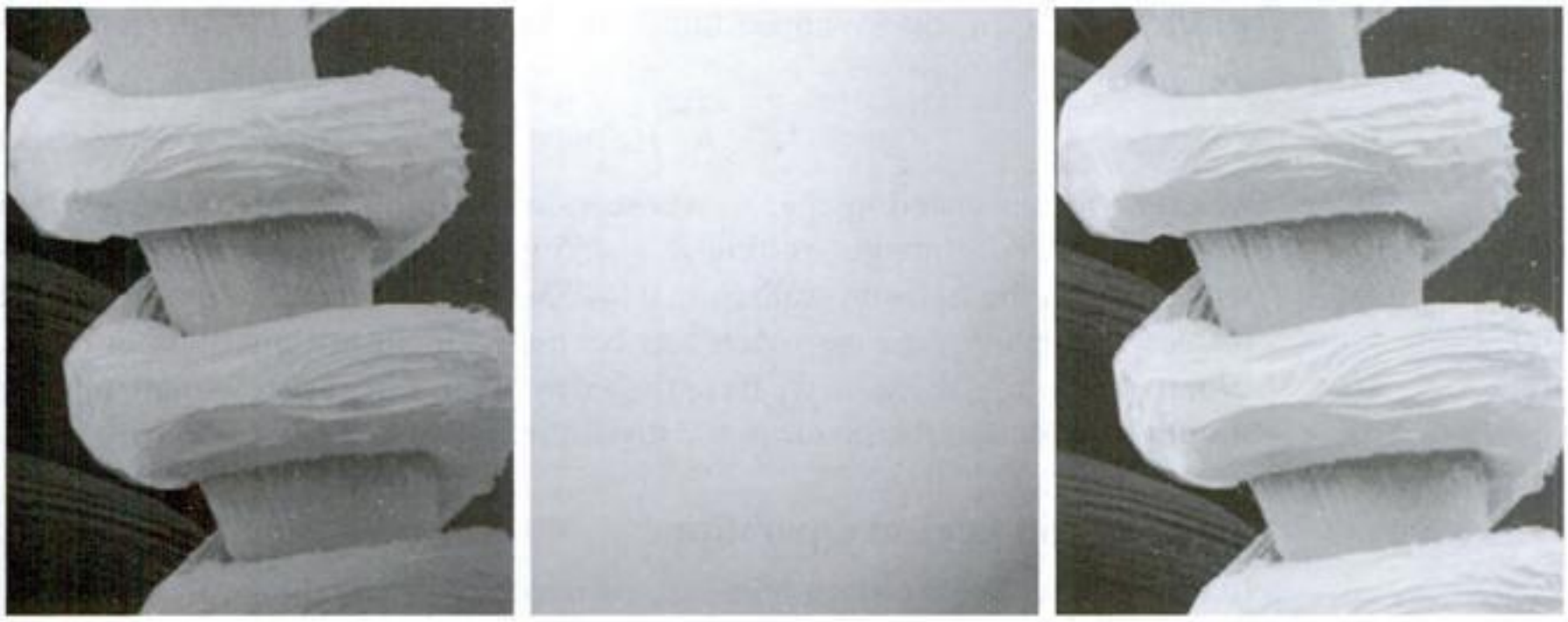
the difference between (a) and (b). Some fine blood vessel structures are visible in this image. The difference is clear in Fig. 2.28(d), which was obtained by enhancing the contrast in (c) (we discuss contrast enhancement in the next chapter). Figure 2.28(d) is a clear "map" of how the medium is propagating through the blood vessels in the subject's brain. ■

EXAMPLE 2.7: Using image multiplication and division for shading correction.

■ An important application of image multiplication (and division) is *shading correction*. Suppose that an imaging sensor produces images that can be modeled as the product of a "perfect image," denoted by $f(x, y)$, times a shading function, $h(x, y)$; that is, $g(x, y) = f(x, y)h(x, y)$. If $h(x, y)$ is known, we can obtain $f(x, y)$ by multiplying the sensed image by the inverse of $h(x, y)$ (i.e., dividing g by h). If $h(x, y)$ is not known, but access to the imaging system is possible, we can obtain an approximation to the shading function by imaging a target of constant intensity. When the sensor is not available, we often can estimate the shading pattern directly from the image, as we discuss in Section 9.6. Figure 2.29 shows an example of shading correction.

Another common use of image multiplication is in *masking*, also called *region of interest* (ROI), operations. The process, illustrated in Fig. 2.30, consists simply of multiplying a given image by a mask image that has 1s in the ROI and 0s elsewhere. There can be more than one ROI in the mask image, and the shape of the ROI can be arbitrary, although rectangular shapes are used frequently for ease of implementation. ■

A few comments about implementing image arithmetic operations are in order before we leave this section. In practice, most images are displayed using 8 bits (even 24-bit color images consist of three separate 8-bit channels). Thus, we expect image values to be in the range from 0 to 255. When images

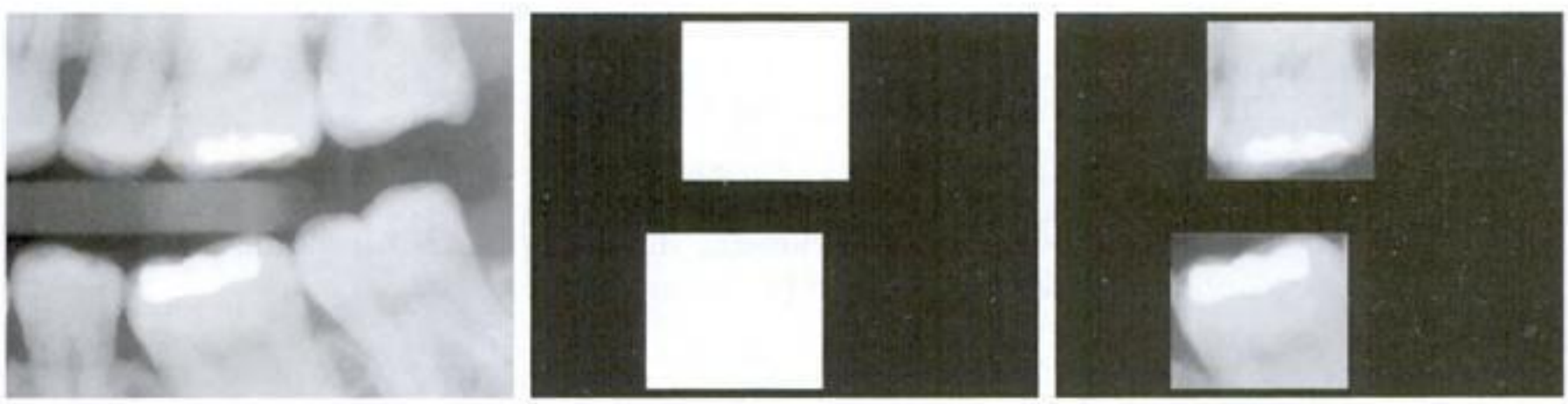


a b c

FIGURE 2.29 Shading correction. (a) Shaded SEM image of a tungsten filament and support, magnified approximately 130 times. (b) The shading pattern. (c) Product of (a) by the reciprocal of (b). (Original image courtesy of Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene.)

are saved in a standard format, such as TIFF or JPEG, conversion to this range is automatic. However, the approach used for the conversion depends on the system used. For example, the values in the difference of two 8-bit images can range from a minimum of −255 to a maximum of 255, and the values of a sum image can range from 0 to 510. Many software packages simply set all negative values to 0 and set to 255 all values that exceed this limit when converting images to 8 bits. Given an image f, an approach that guarantees that the full range of an arithmetic operation between images is "captured" into a fixed number of bits is as follows. First, we perform the operation

$$f_m = f - \min(f) \tag{2.6-10}$$



a b c

FIGURE 2.30 (a) Digital dental X-ray image. (b) ROI mask for isolating teeth with fillings (white corresponds to 1 and black corresponds to 0). (c) Product of (a) and (b).

which creates an image whose minimum value is 0. Then, we perform the operation

$$f_s = K\left[\, f_m / \max(f_m)\right] \tag{2.6-11}$$

which creates a scaled image, f_s, whose values are in the range $[0, K]$. When working with 8-bit images, setting $K = 255$ gives us a scaled image whose intensities span the full 8-bit scale from 0 to 255. Similar comments apply to 16-bit images or higher. This approach can be used for all arithmetic operations. When performing division, we have the extra requirement that a small number should be added to the pixels of the divisor image to avoid division by 0.

2.6.4 Set and Logical Operations

In this section, we introduce briefly some important set and logical operations. We also introduce the concept of a fuzzy set.

Basic set operations

Let A be a set composed of *ordered pairs* of real numbers. If $a = (a_1, a_2)$ is an *element* of A, then we write

$$a \in A \tag{2.6-12}$$

Similarly, if a is not an element of A, we write

$$a \notin A \tag{2.6-13}$$

The set with no elements is called the *null* or *empty set* and is denoted by the symbol $\varnothing$.

A set is specified by the contents of two braces: $\{\cdot\}$. For example, when we write an expression of the form $C = \{w | w = -d, d \in D\}$, we mean that set C is the set of elements, w, such that w is formed by multiplying each of the elements of set D by -1. One way in which sets are used in image processing is to let the elements of sets be the *coordinates* of pixels (ordered pairs of integers) representing regions (objects) in an image.

If every element of a set A is also an element of a set B, then A is said to be a *subset* of B, denoted as

$$A \subseteq B \tag{2.6-14}$$

The *union* of two sets A and B, denoted by

$$C = A \cup B \tag{2.6-15}$$

is the set of elements belonging to either A, B, *or* both. Similarly, the *intersection* of two sets A and B, denoted by

$$D = A \cap B \tag{2.6-16}$$

is the set of elements belonging to both A *and* B. Two sets A and B are said to be *disjoint* or *mutually exclusive* if they have no common elements, in which case,

$$A \cap B = \varnothing \tag{2.6-17}$$

The *set universe*, U, is the set of all elements in a given application. By definition, all set elements in a given application are members of the universe defined for that application. For example, if you are working with the set of real numbers, then the set universe is the real line, which contains all the real numbers. In image processing, we typically define the universe to be the rectangle containing all the pixels in an image.

The *complement* of a set A is the set of elements that are not in A:

$$A^c = \{w | w \notin A\} \tag{2.6-18}$$

The *difference* of two sets A and B, denoted $A - B$, is defined as

$$A - B = \{w | w \in A, w \notin B\} = A \cap B^c \tag{2.6-19}$$

We see that this is the set of elements that belong to A, but not to B. We could, for example, define A^c in terms of U and the set difference operation: $A^c = U - A$.

Figure 2.31 illustrates the preceding concepts, where the universe is the set of coordinates contained within the rectangle shown, and sets A and B are the sets of coordinates contained within the boundaries shown. The result of the set operation indicated in each figure is shown in gray.[†]

In the preceding discussion, set membership is based on position (coordinates). An implicit assumption when working with images is that the intensity of all pixels in the sets is the same, as we have not defined set operations involving intensity values (e.g., we have not specified what the intensities in the intersection of two sets is). The only way that the operations illustrated in Fig. 2.31 can make sense is if the images containing the sets are binary, in which case we can talk about set membership based on coordinates, the assumption being that all member of the sets have the same intensity. We discuss this in more detail in the following subsection.

When dealing with gray-scale images, the preceding concepts are not applicable, because we have to specify the intensities of all the pixels resulting from a set operation. In fact, as you will see in Sections 3.8 and 9.6, the union and intersection operations for gray-scale values usually are defined as the max and min of corresponding pixel pairs, respectively, while the complement is defined as the pairwise differences between a constant and the intensity of every pixel in an image. The fact that we deal with corresponding pixel pairs tells us that gray-scale set operations are array operations, as defined in Section 2.6.1. The following example is a brief illustration of set operations involving gray-scale images. We discuss these concepts further in the two sections mentioned above.

[†]The operations in Eqs. (2.6-12)–(2.6-19) are the basis for the algebra of sets, which starts with properties such as the *commutative laws*: $A \cup B = B \cup A$ and $A \cap B = B \cap A$, and from these develops a broad theory based on set operations. A treatment of the algebra of sets is beyond the scope of the present discussion, but you should be aware of its existence.

a b c
d e

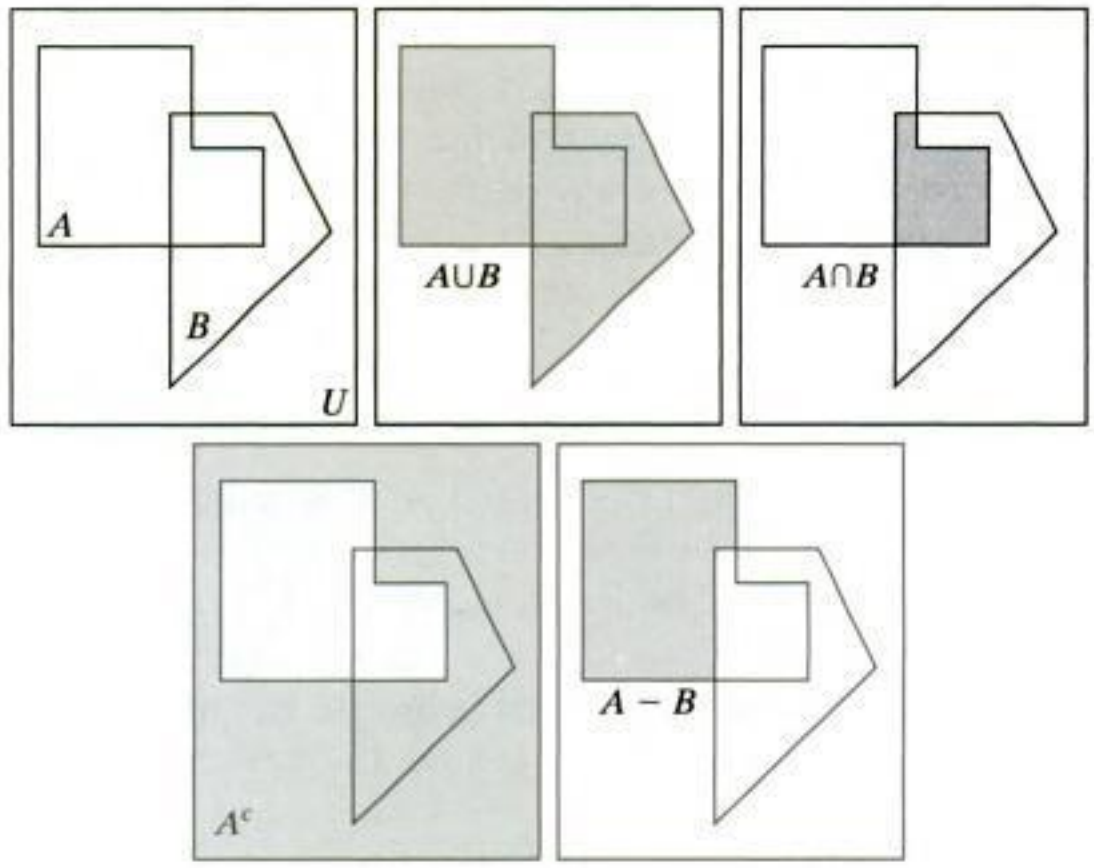


FIGURE 2.31
(a) Two sets of coordinates, A and B, in 2-D space. (b) The union of A and B. (c) The intersection of A and B. (d) The complement of A. (e) The difference between A and B. In (b)–(e) the shaded areas represent the members of the set operation indicated.

EXAMPLE 2.8:
Set operations involving image intensities.

■ Let the elements of a gray-scale image be represented by a set A whose elements are triplets of the form (x, y, z), where x and y are spatial coordinates and z denotes intensity, as mentioned in Section 2.4.2. We can define the *complement* of A as the set $A^c = \{(x, y, K - z) | (x, y, z) \in A\}$, which simply denotes the set of pixels of A whose intensities have been subtracted from a constant K. This constant is equal to $2^k - 1$, where k is the number of intensity bits used to represent z. Let A denote the 8-bit gray-scale image in Fig. 2.32(a), and suppose that we want to form the negative of A using set

a b c

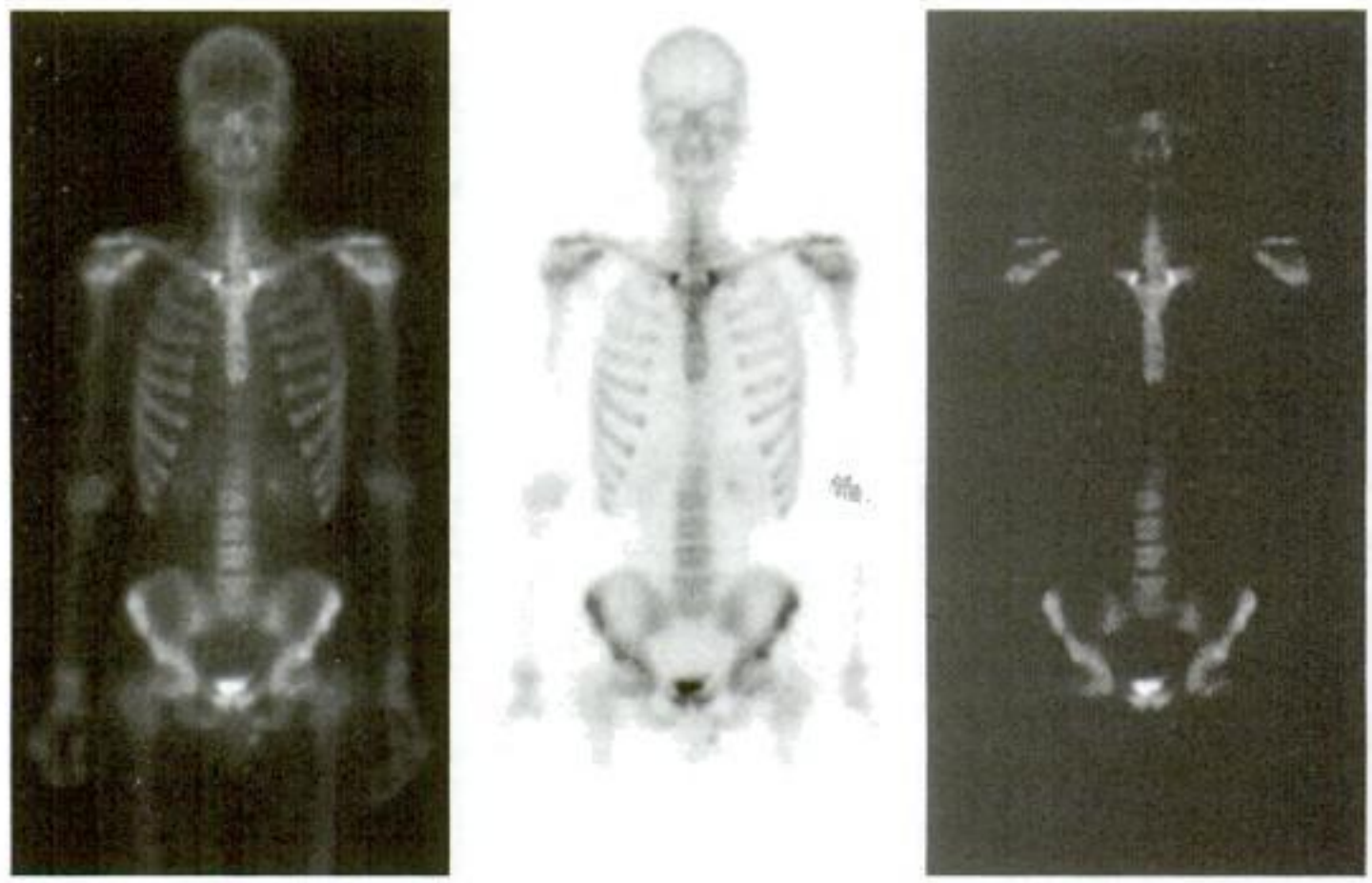


FIGURE 2.32 Set operations involving gray-scale images. (a) Original image. (b) Image negative obtained using set complementation. (c) The union of (a) and a constant image. (Original image courtesy of G.E. Medical Systems.)

operations. We simply form the set $A_n = A^c = \{(x, y, 255 - z) | (x, y, z) \in A\}$. Note that the coordinates are carried over, so A_n is an image of the same size as A. Figure 2.32(b) shows the result.

The union of two gray-scale sets A and B may be defined as the set

$$A \cup B = \left\{ \max_z (a, b) | a \in A, b \in B \right\}$$

That is, the union of two gray-scale sets (images) is an array formed from the maximum intensity between pairs of spatially corresponding elements. Again, note that coordinates carry over, so the union of A and B is an image of the same size as these two images. As an illustration, suppose that A again represents the image in Fig. 2.32(a), and let B denote a rectangular array of the same size as A, but in which all values of z are equal to 3 times the mean intensity, m, of the elements of A. Figure 2.32(c) shows the result of performing the set union, in which all values exceeding $3m$ appear as values from A and all other pixels have value $3m$, which is a mid-gray value. ■

Logical operations

When dealing with binary images, we can think of *foreground* (1-valued) and *background* (0-valued) sets of pixels. Then, if we define regions (objects) as being composed of foreground pixels, the set operations illustrated in Fig. 2.31 become operations between the coordinates of objects in a binary image. When dealing with binary images, it is common practice to refer to union, intersection, and complement as the OR, AND, and NOT *logical* operations, where "logical" arises from logic theory in which 1 and 0 denote true and false, respectively.

Consider two regions (sets) A and B composed of foreground pixels. The OR of these two sets is the set of elements (coordinates) belonging either to A or B or to both. The AND operation is the set of elements that are common to A and B. The NOT operation of a set A is the set of elements not in A. Because we are dealing with images, if A is a given set of foreground pixels, NOT(A) is the set of all pixels in the image that are not in A, these pixels being background pixels and possibly other foreground pixels. We can think of this operation as turning all elements in A to 0 (black) and all the elements not in A to 1 (white). Figure 2.33 illustrates these operations. Note in the fourth row that the result of the operation shown is the set of foreground pixels that belong to A but not to B, which is the definition of set difference in Eq. (2.6-19). The last row in the figure is the XOR (exclusive OR) operation, which is the set of foreground pixels belonging to A or B, but not both. Observe that the preceding operations are between regions, which clearly can be irregular and of different sizes. This is as opposed to the gray-scale operations discussed earlier, which are array operations and thus require sets whose spatial dimensions are the same. That is, gray-scale set operations involve complete images, as opposed to regions of images.

We need be concerned in theory only with the cability to implement the AND, OR, and NOT logic operators because these three operators are *functionally*

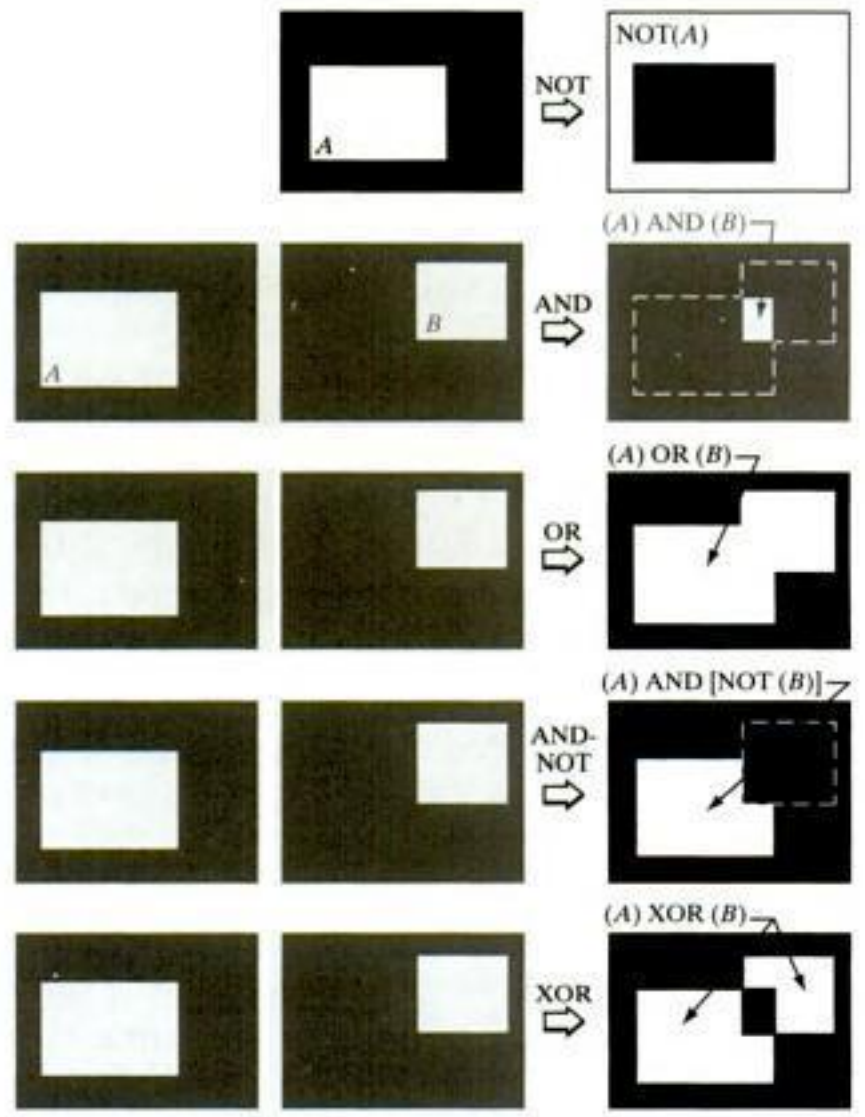


FIGURE 2.33
Illustration of logical operations involving foreground (white) pixels. Black represents binary 0s and white binary 1s. The dashed lines are shown for reference only. They are not part of the result.

complete. In other words, any other logic operator can be implemented by using only these three basic functions, as in the fourth row of Fig. 2.33, where we implemented the set difference operation using AND and NOT. Logic operations are used extensively in image morphology, the topic of Chapter 9.

Fuzzy sets

The preceding set and logical results are *crisp* concepts, in the sense that elements either are or are not members of a set. This presents a serious limitation in some applications. Consider a simple example. Suppose that we wish to categorize all people in the world as being young or not young. Using crisp sets, let U denote the set of all people and let A be a subset of U, which we call the *set of young people*. In order to form set A, we need a *membership function* that assigns a value of 1 or 0 to every element (person) in U. If the value assigned to an element of U is 1, then that element is a member of A; otherwise it is not. Because we are dealing with a bi-valued logic, the membership function simply defines a threshold at or below which a person is considered young, and above which a person is considered not young. Suppose that we define as young any person of age 20 or younger. We see an immediate difficulty. A person whose age is 20 years and 1 sec would not be a member of the set of young people. This limitation arises regardless of the age threshold we use to classify a person as being young. What we need is more flexibility in what we mean by "young," that is, we need a *gradual* transition from young to not young. The theory of *fuzzy sets* implements this concept by utilizing membership functions

that are gradual between the limit values of 1 (definitely young) to 0 (definitely not young). Using fuzzy sets, we can make a statement such as a person being 50% young (in the middle of the transition between young and not young). In other words, age is an imprecise concept, and fuzzy logic provides the tools to deal with such concepts. We explore fuzzy sets in detail in Section 3.8.

2.6.5 Spatial Operations

Spatial operations are performed directly on the pixels of a given image. We classify spatial operations into three broad categories: (1) single-pixel operations, (2) neighborhood operations, and (3) geometric spatial transformations.

Single-pixel operations

The simplest operation we perform on a digital image is to alter the values of its individual pixels based on their intensity. This type of process may be expressed as a transformation function, T, of the form:

$$s = T(z) \tag{2.6-20}$$

where z is the intensity of a pixel in the original image and s is the (mapped) intensity of the corresponding pixel in the processed image. For example, Fig. 2.34 shows the transformation used to obtain the negative of an 8-bit image, such as the image in Fig. 2.32(b), which we obtained using set operations. We discuss in Chapter 3 a number of techniques for specifying intensity transformation functions.

Neighborhood operations

Let S_{xy} denote the set of coordinates of a neighborhood centered on an arbitrary point (x, y) in an image, f. Neighborhood processing generates a corresponding pixel at the same coordinates in an output (processed) image, g, such that the value of that pixel is determined by a specified operation involving the pixels in the input image with coordinates in S_{xy}. For example, suppose that the specified operation is to compute the average value of the pixels in a rectangular neighborhood of size $m \times n$ centered on (x, y). The locations of pixels

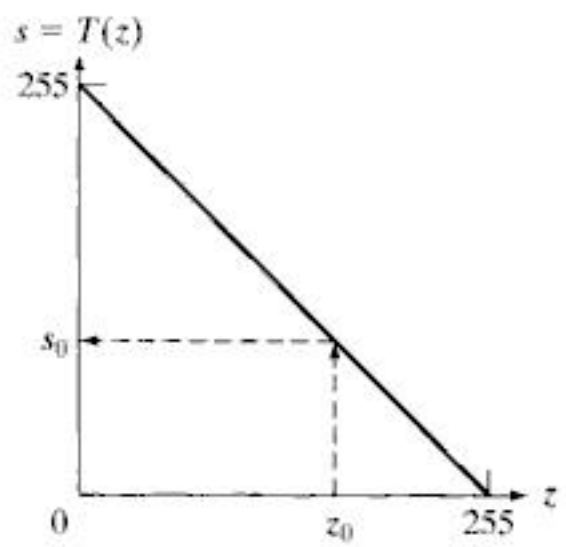
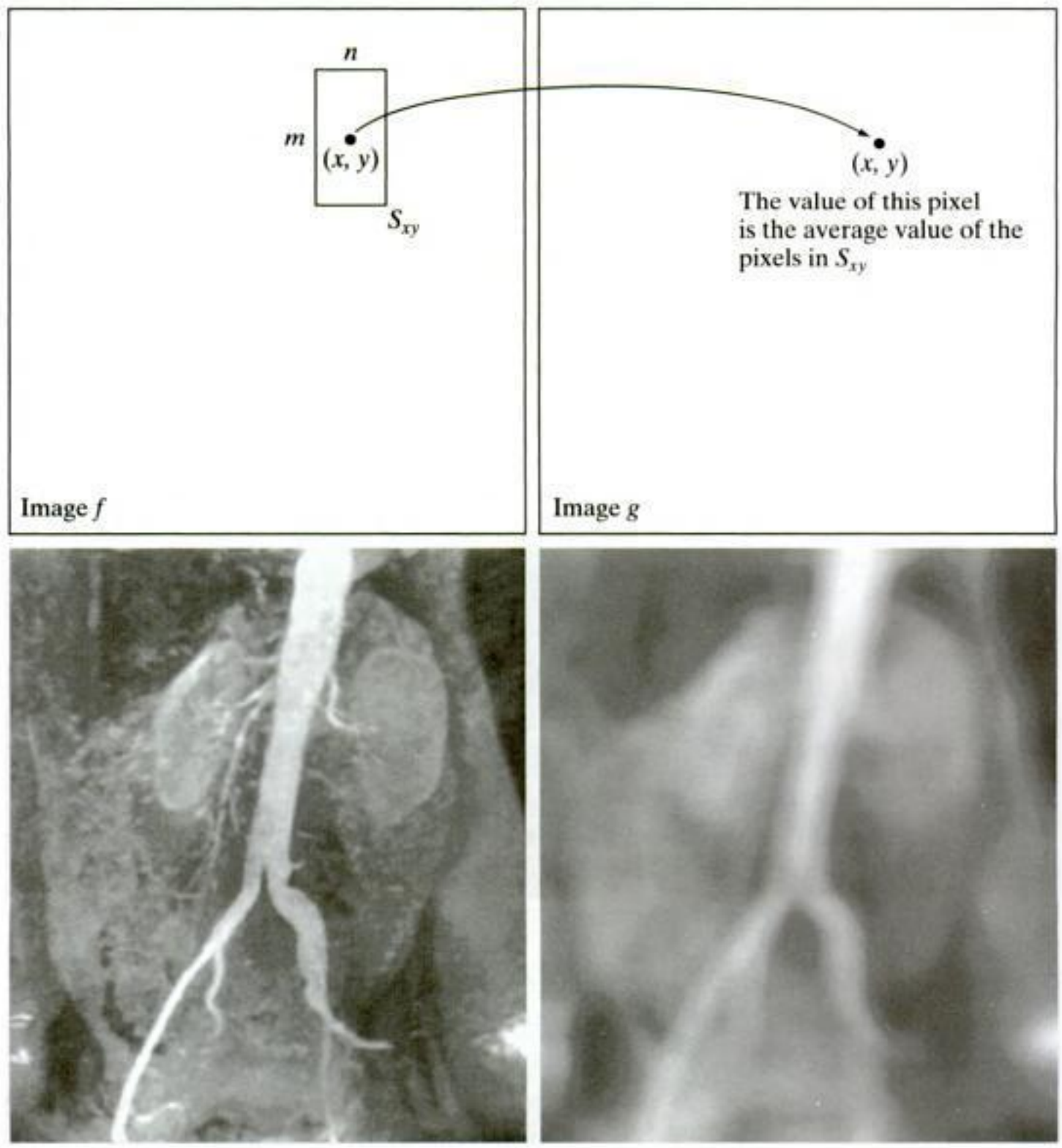


FIGURE 2.34 Intensity transformation function used to obtain the negative of an 8-bit image. The dashed arrows show transformation of an arbitrary input intensity value z_0 into its corresponding output value s_0.

a b
c d

FIGURE 2.35
Local averaging using neighborhood processing. The procedure is illustrated in (a) and (b) for a rectangular neighborhood. (c) The aortic angiogram discussed in Section 1.3.2. (d) The result of using Eq. (2.6-21) with $m = n = 41$. The images are of size 790 × 686 pixels.



in this region constitute the set S_{xy}. Figures 2.35(a) and (b) illustrate the process. We can express this operation in equation form as

$$g(x, y) = \frac{1}{mn} \sum_{(r,c) \in S_{xy}} f(r, c) \tag{2.6-21}$$

where r and c are the row and column coordinates of the pixels whose coordinates are members of the set S_{xy}. Image g is created by varying the coordinates (x, y) so that the center of the neighborhood moves from pixel to pixel in image f, and repeating the neighborhood operation at each new location. For instance, the image in Fig. 2.35(d) was created in this manner using a neighborhood of size 41 × 41. The net effect is to perform local blurring in the original image. This type of process is used, for example, to eliminate small details and thus render "blobs" corresponding to the largest regions of an image. We

discuss neighborhood processing in Chapters 3 and 5, and in several other places in the book.

Geometric spatial transformations and image registration

Geometric transformations modify the spatial relationship between pixels in an image. These transformations often are called *rubber-sheet* transformations because they may be viewed as analogous to "printing" an image on a sheet of rubber and then stretching the sheet according to a predefined set of rules. In terms of digital image processing, a geometric transformation consists of two basic operations: (1) a spatial transformation of coordinates and (2) intensity interpolation that assigns intensity values to the spatially transformed pixels.

The transformation of coordinates may be expressed as

$$(x, y) = T\{(v, w)\} \tag{2.6-22}$$

where (v, w) are pixel coordinates in the original image and (x, y) are the corresponding pixel coordinates in the transformed image. For example, the transformation $(x, y) = T\{(v, w)\} = (v/2, w/2)$ shrinks the original image to half its size in both spatial directions. One of the most commonly used spatial coordinate transformations is the *affine transform* (Wolberg [1990]), which has the general form

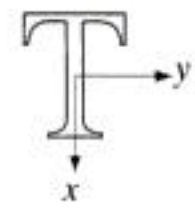
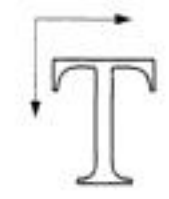
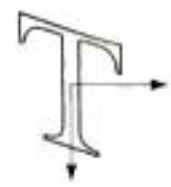
$$[x \;\; y \;\; 1] = [v \;\; w \;\; 1]\,\mathbf{T} = [v \;\; w \;\; 1]\begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix} \tag{2.6-23}$$

This transformation can scale, rotate, translate, or sheer a set of coordinate points, depending on the value chosen for the elements of matrix **T**. Table 2.2 illustrates the matrix values used to implement these transformations. The real power of the matrix representation in Eq. (2.6-23) is that it provides the framework for concatenating together a sequence of operations. For example, if we want to resize an image, rotate it, and move the result to some location, we simply form a 3×3 matrix equal to the product of the scaling, rotation, and translation matrices from Table 2.2.

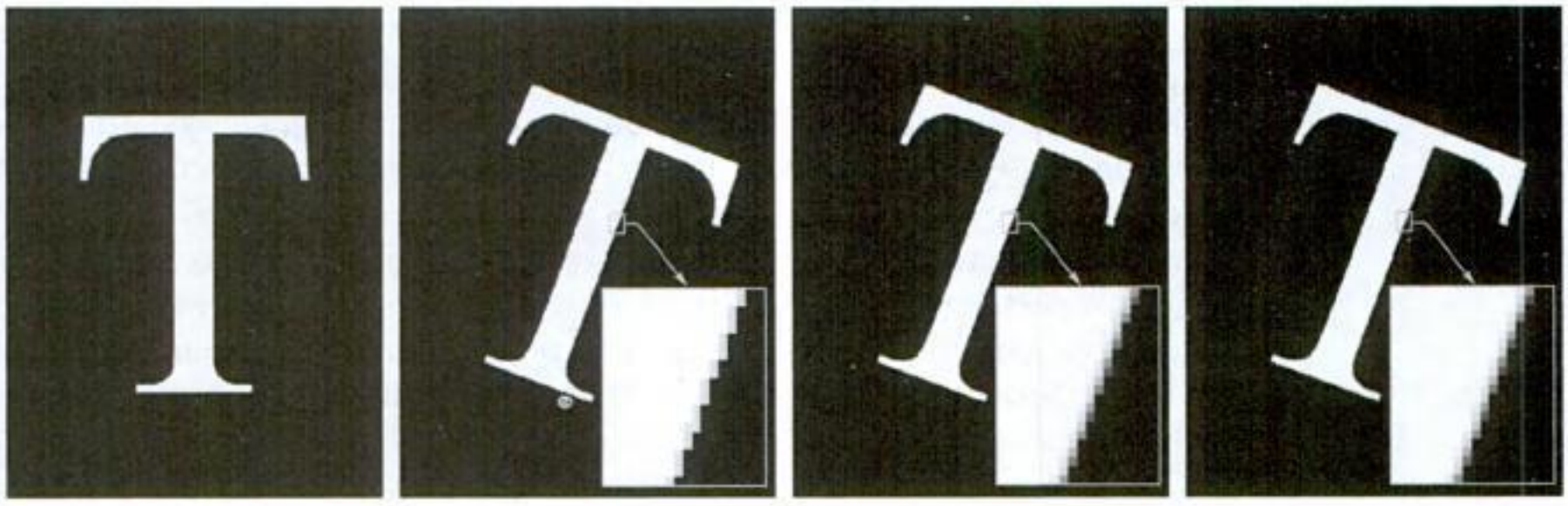
The preceding transformations relocate pixels on an image to new locations. To complete the process, we have to assign intensity values to those locations. This task is accomplished using intensity interpolation. We already discussed this topic in Section 2.4.4. We began that section with an example of zooming an image and discussed the issue of intensity assignment to new pixel locations. Zooming is simply scaling, as detailed in the second row of Table 2.2, and an analysis similar to the one we developed for zooming is applicable to the problem of assigning intensity values to the relocated pixels resulting from the other transformations in Table 2.2. As in Section 2.4.4, we consider nearest neighbor, bilinear, and bicubic interpolation techniques when working with these transformations.

In practice, we can use Eq. (2.6-23) in two basic ways. The first, called a *forward mapping*, consists of scanning the pixels of the input image and, at

TABLE 2.2
Affine transformations based on Eq. (2.6-23).

Transformation Name	Affine Matrix, T	Coordinate Equations	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = w$	
Scaling	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = c_x v$ $y = c_y w$	
Rotation	$\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v\cos\theta - w\sin\theta$ $y = v\sin\theta + w\cos\theta$	
Translation	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	$x = v + t_x$ $y = w + t_y$	
Shear (vertical)	$\begin{bmatrix} 1 & 0 & 0 \\ s_v & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v + s_v w$ $y = w$	
Shear (horizontal)	$\begin{bmatrix} 1 & s_h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x = v$ $y = s_h v + w$	

each location, (v, w), computing the spatial location, (x, y), of the corresponding pixel in the output image using Eq. (2.6-23) directly. A problem with the forward mapping approach is that two or more pixels in the input image can be transformed to the same location in the output image, raising the question of how to combine multiple output values into a single output pixel. In addition, it is possible that some output locations may not be assigned a pixel at all. The second approach, called *inverse mapping*, scans the output pixel locations and, at each location, (x, y), computes the corresponding location in the input image using $(v, w) = T^{-1}(x, y)$. It then interpolates (using one of the techniques discussed in Section 2.4.4) among the nearest input pixels to determine the intensity of the output pixel value. Inverse mappings are more efficient to implement than forward mappings and are used in numerous commercial implementations of spatial transformations (for example, MATLAB uses this approach).



a b c d

FIGURE 2.36 (a) A 300 dpi image of the letter T. (b) Image rotated 21° using nearest neighbor interpolation to assign intensity values to the spatially transformed pixels. (c) Image rotated 21° using bilinear interpolation. (d) Image rotated 21° using bicubic interpolation. The enlarged sections show edge detail for the three interpolation approaches.

EXAMPLE 2.9: Image rotation and intensity interpolation.

■ The objective of this example is to illustrate image rotation using an affine transform. Figure 2.36(a) shows a 300 dpi image and Figs. 2.36(b)–(d) are the results of rotating the original image by 21°, using nearest neighbor, bilinear, and bicubic interpolation, respectively. Rotation is one of the most demanding geometric transformations in terms of preserving straight-line features. As we see in the figure, nearest neighbor interpolation produced the most jagged edges and, as in Section 2.4.4, bilinear interpolation yielded significantly improved results. As before, using bicubic interpolation produced slightly sharper results. In fact, if you compare the enlarged detail in Figs. 2.36(c) and (d), you will notice in the middle of the subimages that the number of vertical gray "blocks" that provide the intensity transition from light to dark in Fig. 2.36(c) is larger than the corresponding number of blocks in (d), indicting that the latter is a sharper edge. Similar results would be obtained with the other spatial transformations in Table 2.2 that require interpolation (the identity transformation does not, and neither does the translation transformation if the increments are an integer number of pixels). This example was implemented using the inverse mapping approach discussed in the preceding paragraph. ■

Image registration is an important application of digital image processing used to align two or more images of the same scene. In the preceding discussion, the form of the transformation function required to achieve a desired geometric transformation was known. In image registration, we have available the input and output images, but the specific transformation that produced the output image from the input generally is unknown. The problem, then, is to estimate the transformation function and then use it to register the two images. To clarify terminology, the input image is the image that we wish to transform, and what we call the *reference* image is the image against which we want to register the input.

For example, it may be of interest to align (register) two or more images taken at approximately the same time, but using different imaging systems, such as an MRI (magnetic resonance imaging) scanner and a PET (positron emission tomography) scanner. Or, perhaps the images were taken at different times using the same instrument, such as satellite images of a given location taken several days, months, or even years apart. In either case, combining the images or performing quantitative analysis and comparisons between them requires compensating for geometric distortions caused by differences in viewing angle, distance, and orientation; sensor resolution; shift in object positions; and other factors.

One of the principal approaches for solving the problem just discussed is to use *tie points* (also called *control points*), which are corresponding points whose locations are known precisely in the input and reference images. There are numerous ways to select tie points, ranging from interactively selecting them to applying algorithms that attempt to detect these points automatically. In some applications, imaging systems have physical artifacts (such as small metallic objects) embedded in the imaging sensors. These produce a set of *known* points (called *reseau marks*) directly on all images captured by the system, which can be used as guides for establishing tie points.

The problem of estimating the transformation function is one of modeling. For example, suppose that we have a set of four tie points each in an input and a reference image. A simple model based on a bilinear approximation is given by

$$x = c_1 v + c_2 w + c_3 v w + c_4 \tag{2.6-24}$$

and

$$y = c_5 v + c_6 w + c_7 v w + c_8 \tag{2.6-25}$$

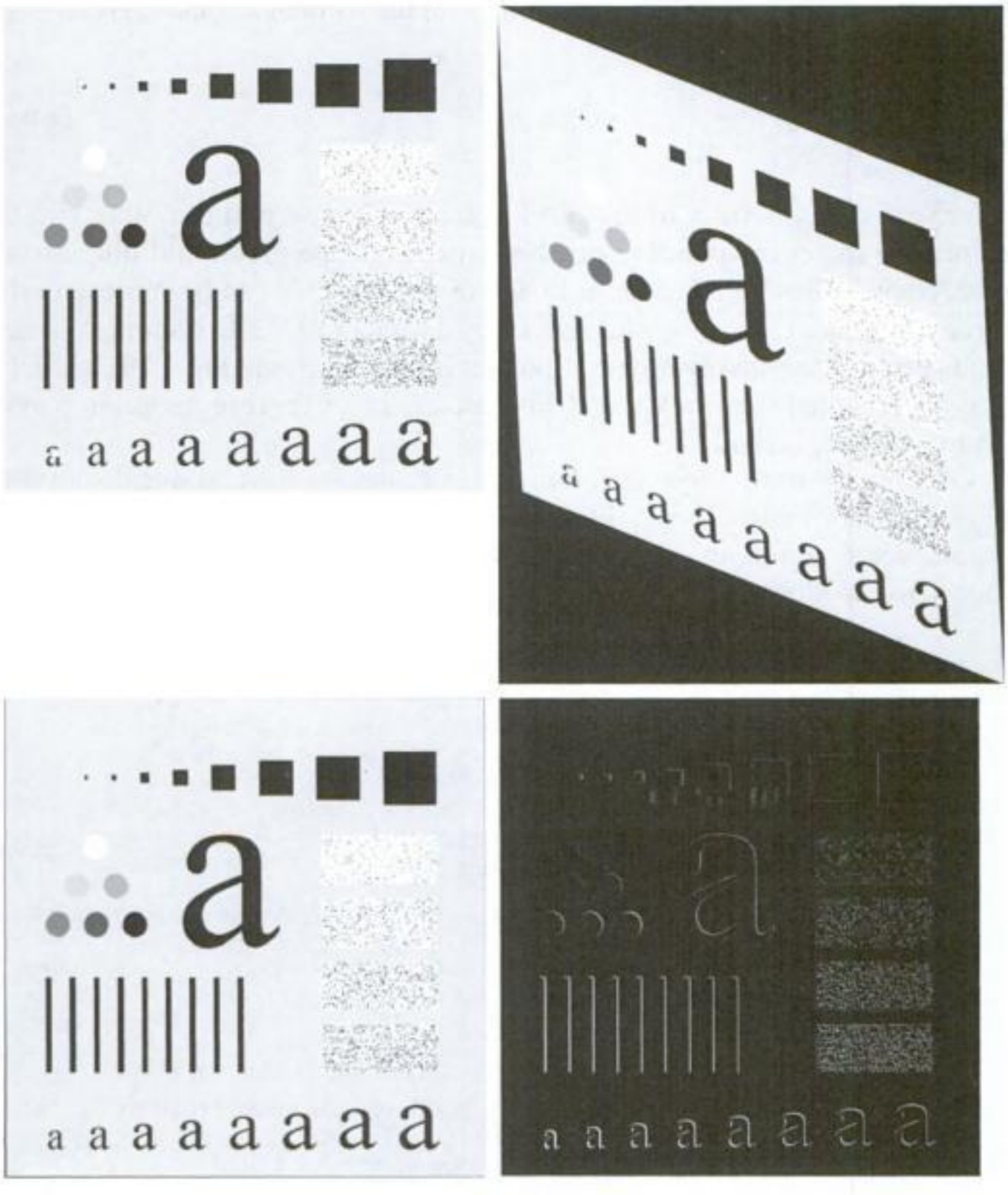
where, during the estimation phase, (v, w) and (x, y) are the coordinates of tie points in the input and reference images, respectively. If we have four pairs of corresponding tie points in both images, we can write eight equations using Eqs. (2.6-24) and (2.6-25) and use them to solve for the eight unknown coefficients, $c_1, c_2, \ldots, c_8$. These coefficients constitute the model that transforms the pixels of one image into the locations of the pixels of the other to achieve registration.

Once we have the coefficients, Eqs. (2.6-24) and (2.6-25) become our vehicle for transforming all the pixels in the input image to generate the desired new image, which, if the tie points were selected correctly, should be registered with the reference image. In situations where four tie points are insufficient to obtain satisfactory registration, an approach used frequently is to select a larger number of tie points and then treat the quadrilaterals formed by groups of four tie points as subimages. The subimages are processed as above, with all the pixels within a quadrilateral being transformed using the coefficients determined from those tie points. Then we move to another set of four tie points and repeat the procedure until all quadrilateral regions have been processed. Of course, it is possible to use regions that are more complex than quadrilaterals and employ more complex models, such as polynomials fitted by least

squares algorithms. In general, the number of control points and sophistication of the model required to solve a problem is dependent on the severity of the geometric distortion. Finally, keep in mind that the transformation defined by Eqs. (2.6-24) and (2.6-25), or any other model for that matter, simply maps the spatial coordinates of the pixels in the input image. We still need to perform intensity interpolation using any of the methods discussed previously to assign intensity values to those pixels.

EXAMPLE 2.10: Image registration.

■ Figure 2.37(a) shows a reference image and Fig. 2.37(b) shows the same image, but distorted geometrically by vertical and horizontal shear. Our objective is to use the reference image to obtain tie points and then use the tie points to register the images. The tie points we selected (manually) are shown as small white squares near the corners of the images (we needed only four tie



a b
c d

FIGURE 2.37 Image registration. (a) Reference image. (b) Input (geometrically distorted image). Corresponding tie points are shown as small white squares near the corners. (c) Registered image (note the errors in the border). (d) Difference between (a) and (c), showing more registration errors.

points because the distortion is linear shear in both directions). Figure 2.37(c) shows the result of using these tie points in the procedure discussed in the preceding paragraphs to achieve registration. We note that registration was not perfect, as is evident by the black edges in Fig. 2.37(c). The difference image in Fig. 2.37(d) shows more clearly the slight lack of registration between the reference and corrected images. The reason for the discrepancies is error in the manual selection of the tie points. It is difficult to achieve perfect matches for tie points when distortion is so severe. ■

Consult the Tutorials section in the book Web site for a brief tutorial on vectors and matrices.

2.6.6 Vector and Matrix Operations

Multispectral image processing is a typical area in which vector and matrix operations are used routinely. For example, you will learn in Chapter 6 that color images are formed in RGB color space by using red, green, and blue component images, as Fig. 2.38 illustrates. Here we see that *each* pixel of an RGB image has three components, which can be organized in the form of a *column vector*

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \tag{2.6-26}$$

where z_1 is the intensity of the pixel in the red image, and the other two elements are the corresponding pixel intensities in the green and blue images, respectively. Thus an RGB color image of size $M \times N$ can be represented by three component images of this size, or by a total of MN 3-D vectors. A general multispectral case involving n component images (e.g., see Fig. 1.10) will result in n-dimensional vectors. We use this type of vector representation in parts of Chapters 6, 10, 11, and 12.

Once pixels have been represented as vectors we have at our disposal the tools of vector-matrix theory. For example, the *Euclidean distance*, D, between a pixel vector $\mathbf{z}$ and an arbitrary point $\mathbf{a}$ in n-dimensional space is defined as the vector product

$$\begin{aligned} D(\mathbf{z}, \mathbf{a}) &= \left[(\mathbf{z} - \mathbf{a})^T(\mathbf{z} - \mathbf{a})\right]^{\frac{1}{2}} \\ &= \left[(z_1 - a_1)^2 + (z_2 - a_2)^2 + \cdots + (z_n - a_n)^2\right]^{\frac{1}{2}} \end{aligned} \tag{2.6-27}$$

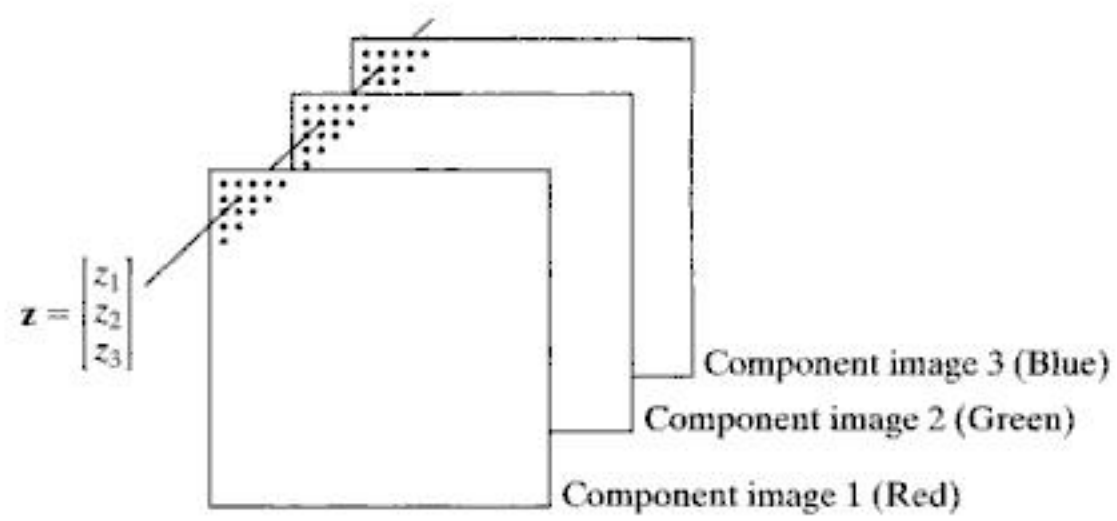


FIGURE 2.38 Formation of a vector from corresponding pixel values in three RGB component images.

We see that this is a generalization of the 2-D Euclidean distance defined in Eq. (2.5-1). Equation (2.6-27) sometimes is referred to as a *vector norm*, denoted by $\|\mathbf{z} - \mathbf{a}\|$. We will use distance computations numerous times in later chapters.

Another important advantage of pixel vectors is in linear transformations, represented as

$$\mathbf{w} = \mathbf{A}(\mathbf{z} - \mathbf{a}) \tag{2.6-28}$$

where $\mathbf{A}$ is a matrix of size $m \times n$ and $\mathbf{z}$ and $\mathbf{a}$ are column vectors of size $n \times 1$. As you will learn later, transformations of this type have a number of useful applications in image processing.

As noted in Eq. (2.4-2), entire images can be treated as matrices (or, equivalently, as vectors), a fact that has important implication in the solution of numerous image processing problems. For example, we can express an image of size $M \times N$ as a vector of dimension $MN \times 1$ by letting the first row of the image be the first N elements of the vector, the second row the next N elements, and so on. With images formed in this manner, we can express a broad range of linear processes applied to an image by using the notation

$$\mathbf{g} = \mathbf{Hf} + \mathbf{n} \tag{2.6-29}$$

where $\mathbf{f}$ is an $MN \times 1$ vector representing an input image, $\mathbf{n}$ is an $MN \times 1$ vector representing an $M \times N$ noise pattern, $\mathbf{g}$ is an $MN \times 1$ vector representing a processed image, and $\mathbf{H}$ is an $MN \times MN$ matrix representing a linear process applied to the input image (see Section 2.6.2 regarding linear processes). It is possible, for example, to develop an entire body of generalized techniques for image restoration starting with Eq. (2.6-29), as we discuss in Section 5.9. We touch on the topic of using matrices again in the following section, and show other uses of matrices for image processing in Chapters 5, 8, 11, and 12.

2.6.7 Image Transforms

All the image processing approaches discussed thus far operate directly on the pixels of the input image; that is, they work directly in the *spatial domain*. In some cases, image processing tasks are best formulated by transforming the input images, carrying the specified task in a *transform domain*, and applying the inverse transform to return to the spatial domain. You will encounter a number of different transforms as you proceed through the book. A particularly important class of 2-D linear transforms, denoted $T(u, v)$, can be expressed in the general form

$$T(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) r(x, y, u, v) \tag{2.6-30}$$

where $f(x, y)$ is the input image, $r(x, y, u, v)$ is called the *forward transformation kernel*, and Eq. (2.6-30) is evaluated for $u = 0, 1, 2, \ldots, M - 1$ and $v = 0, 1, 2, \ldots, N - 1$. As before, x and y are spatial variables, while M and N

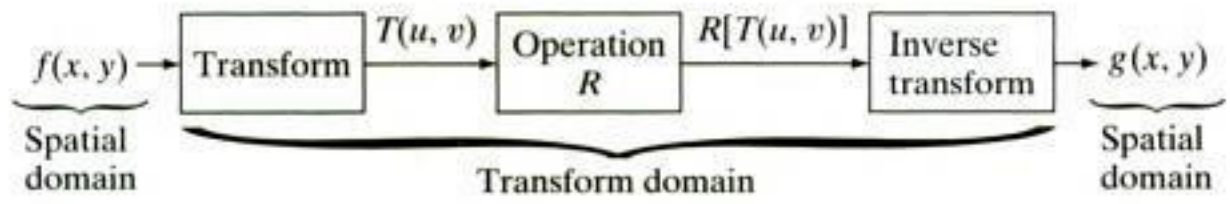


FIGURE 2.39 General approach for operating in the linear transform domain.

are the row and column dimensions of f. Variables u and v are called the *transform variables*. $T(u, v)$ is called the *forward transform* of $f(x, y)$. Given $T(u, v)$, we can recover $f(x, y)$ using the *inverse transform* of $T(u, v)$,

$$f(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} T(u, v)s(x, y, u, v) \tag{2.6-31}$$

for $x = 0, 1, 2, \ldots, M - 1$ and $y = 0, 1, 2, \ldots, N - 1$, where $s(x, y, u, v)$ is called the *inverse transformation kernel*. Together, Eqs. (2.6-30) and (2.6-31) are called a *transform pair*.

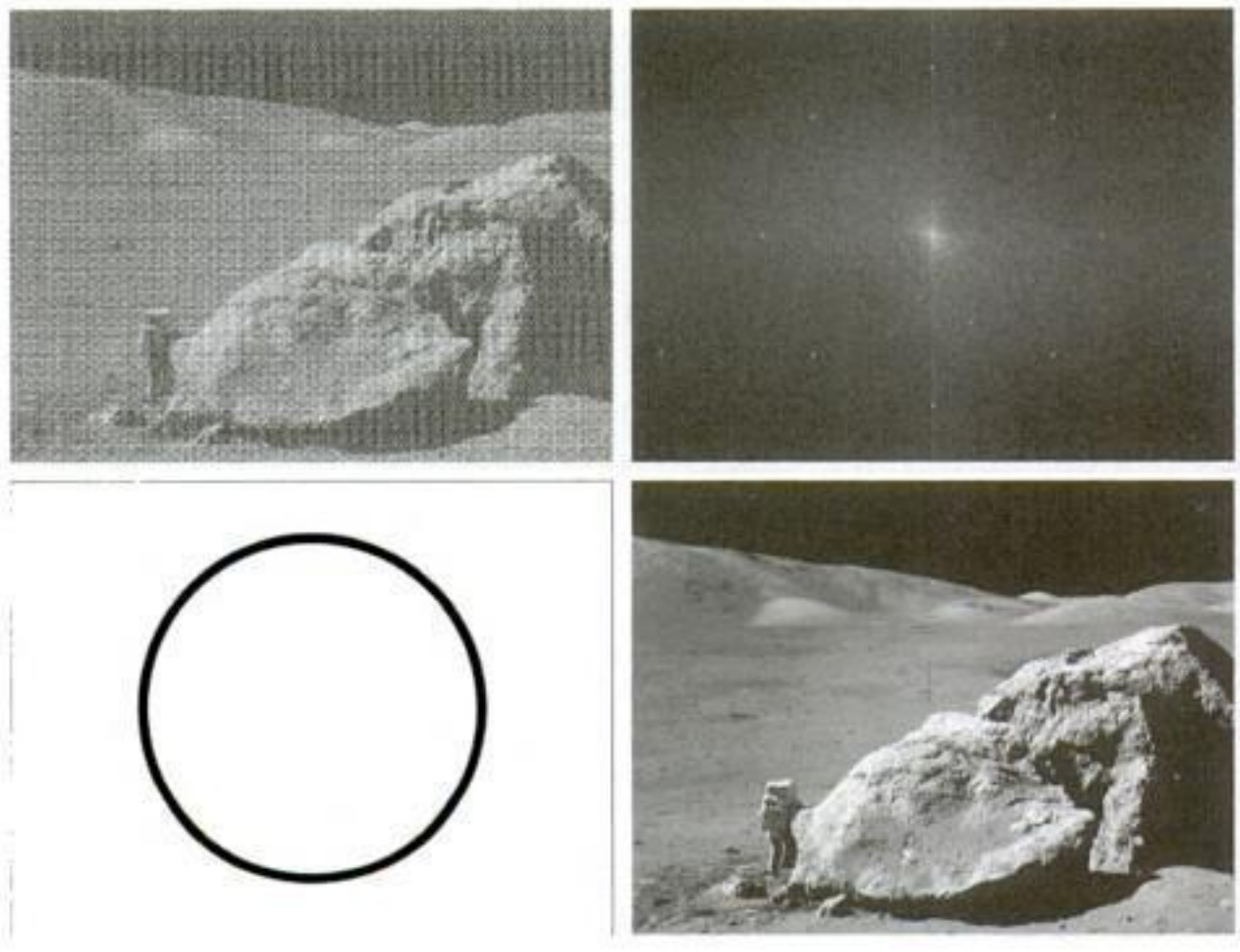
Figure 2.39 shows the basic steps for performing image processing in the linear transform domain. First, the input image is transformed, the transform is then modified by a predefined operation, and, finally, the output image is obtained by computing the inverse of the modified transform. Thus, we see that the process goes from the spatial domain to the transform domain and then back to the spatial domain.

EXAMPLE 2.11: Image processing in the transform domain.

■ Figure 2.40 shows an example of the steps in Fig. 2.39. In this case the transform used was the Fourier transform, which we mention briefly later in this section and discuss in detail in Chapter 4. Figure 2.40(a) is an image corrupted

a b
c d

FIGURE 2.40 (a) Image corrupted by sinusoidal interference. (b) Magnitude of the Fourier transform showing the bursts of energy responsible for the interference. (c) Mask used to eliminate the energy bursts. (d) Result of computing the inverse of the modified Fourier transform. (Original image courtesy of NASA.)



by sinusoidal interference, and Fig. 2.40(b) is the magnitude of its Fourier transform, which is the output of the first stage in Fig. 2.39. As you will learn in Chapter 4, sinusoidal interference in the spatial domain appears as bright bursts of intensity in the transform domain. In this case, the bursts are in a circular pattern that can be seen in Fig. 2.40(b). Figure 2.40(c) shows a mask image (called a *filter*) with white and black representing 1 and 0, respectively. For this example, the operation in the second box of Fig. 2.39 is to multiply the mask by the transform, thus eliminating the bursts responsible for the interference. Figure 2.40(d) shows the final result, obtained by computing the inverse of the modified transform. The interference is no longer visible, and important detail is quite clear. In fact, you can even see the *fiducial marks* (faint crosses) that are used for image alignment. ■

The forward transformation kernel is said to be *separable* if

$$r(x, y, u, v) = r_1(x, u)r_2(y, v) \tag{2.6-32}$$

In addition, the kernel is said to be *symmetric* if $r_1(x, y)$ is functionally equal to $r_2(x, y)$, so that

$$r(x, y, u, v) = r_1(x, u)r_1(y, v) \tag{2.6-33}$$

Identical comments apply to the inverse kernel by replacing r with s in the preceding equations.

The 2-D Fourier transform discussed in Example 2.11 has the following forward and inverse kernels:

$$r(x, y, u, v) = e^{-j2\pi(ux/M+vy/N)} \tag{2.6-34}$$

and

$$s(x, y, u, v) = \frac{1}{MN}e^{j2\pi(ux/M+vy/N)} \tag{2.6-35}$$

respectively, where $j = \sqrt{-1}$, so these kernels are complex. Substituting these kernels into the general transform formulations in Eqs. (2.6-30) and (2.6-31) gives us the *discrete Fourier transform pair*:

$$T(u, v) = \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x, y)e^{-j2\pi(ux/M+vy/N)} \tag{2.6-36}$$

and

$$f(x, y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} T(u, v)e^{j2\pi(ux/M+vy/N)} \tag{2.6-37}$$

These equations are of fundamental importance in digital image processing, and we devote most of Chapter 4 to deriving them starting from basic principles and then using them in a broad range of applications.

It is not difficult to show that the Fourier kernels are separable and symmetric (Problem 2.25), and that separable and symmetric kernels allow 2-D transforms to be computed using 1-D transforms (Problem 2.26). When the

forward and inverse kernels of a transform pair satisfy these two conditions, and $f(x, y)$ is a square image of size $M \times M$, Eqs. (2.6-30) and (2.6-31) can be expressed in matrix form:

$$\mathbf{T} = \mathbf{AFA} \tag{2.6-38}$$

where $\mathbf{F}$ is an $M \times M$ matrix containing the elements of $f(x, y)$ [see Eq. (2.4-2)], $\mathbf{A}$ is an $M \times M$ matrix with elements $a_{ij} = r_1(i, j)$, and $\mathbf{T}$ is the resulting $M \times M$ transform, with values $T(u, v)$ for $u, v = 0, 1, 2, \ldots, M - 1$.

To obtain the inverse transform, we pre- and post-multiply Eq. (2.6-38) by an inverse transformation matrix $\mathbf{B}$:

$$\mathbf{BTB} = \mathbf{BAFAB} \tag{2.6-39}$$

If $\mathbf{B} = \mathbf{A}^{-1}$,

$$\mathbf{F} = \mathbf{BTB} \tag{2.6-40}$$

indicating that $\mathbf{F}$ [whose elements are equal to image $f(x, y)$] can be recovered completely from its forward transform. If $\mathbf{B}$ is not equal to $\mathbf{A}^{-1}$, then use of Eq. (2.6-40) yields an approximation:

$$\hat{\mathbf{F}} = \mathbf{BAFAB} \tag{2.6-41}$$

In addition to the Fourier transform, a number of important transforms, including the Walsh, Hadamard, discrete cosine, Haar, and slant transforms, can be expressed in the form of Eqs. (2.6-30) and (2.6-31) or, equivalently, in the form of Eqs. (2.6-38) and (2.6-40). We discuss several of these and some other types of image transforms in later chapters.

2.6.8 Probabilistic Methods

Consult the Tutorials section in the book Web site for a brief overview of probability theory.

Probability finds its way into image processing work in a number of ways. The simplest is when we treat intensity values as random quantities. For example, let z_i, $i = 0, 1, 2, \ldots, L - 1$, denote the values of all possible intensities in an $M \times N$ digital image. The probability, $p(z_k)$, of intensity level z_k occurring in a given image is estimated as

$$p(z_k) = \frac{n_k}{MN} \tag{2.6-42}$$

where n_k is the number of times that intensity z_k occurs in the image and MN is the total number of pixels. Clearly,

$$\sum_{k=0}^{L-1} p(z_k) = 1 \tag{2.6-43}$$

Once we have $p(z_k)$, we can determine a number of important image characteristics. For example, the mean (average) intensity is given by

$$m = \sum_{k=0}^{L-1} z_k p(z_k) \tag{2.6-44}$$

Similarly, the variance of the intensities is

$$\sigma^2 = \sum_{k=0}^{L-1} (z_k - m)^2 p(z_k) \tag{2.6-45}$$

The variance is a measure of the spread of the values of z about the mean, so it is a useful measure of image contrast. In general, the nth moment of random variable z about the mean is defined as

$$\mu_n(z) = \sum_{k=0}^{L-1} (z_k - m)^n p(z_k) \tag{2.6-46}$$

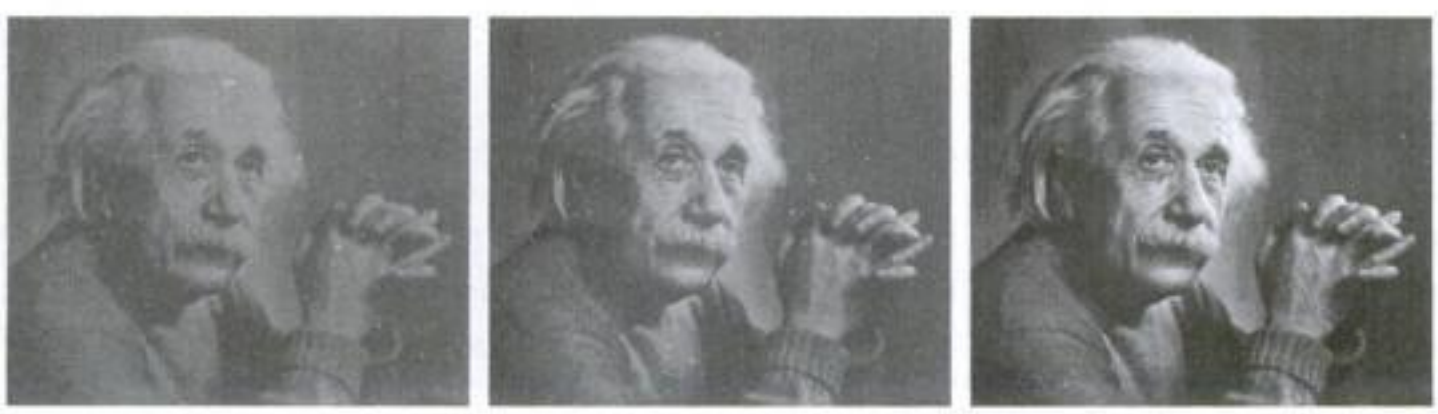
We see that $\mu_0(z) = 1$, $\mu_1(z) = 0$, and $\mu_2(z) = \sigma^2$. Whereas the mean and variance have an immediately obvious relationship to visual properties of an image, higher-order moments are more subtle. For example, a positive third moment indicates that the intensities are biased to values higher than the mean, a negative third moment would indicate the opposite condition, and a zero third moment would tell us that the intensities are distributed approximately equally on both sides of the mean. These features are useful for computational purposes, but they do not tell us much about the appearance of an image in general.

The units of the variance are in intensity values squared. When comparing contrast values, we usually use the standard deviation, σ (square root of the variance), instead because its dimensions are directly in terms of intensity values.

■ Figure 2.41 shows three 8-bit images exhibiting low, medium, and high contrast, respectively. The standard deviations of the pixel intensities in the three images are 14.3, 31.6, and 49.2 intensity levels, respectively. The corresponding variance values are 204.3, 997.8, and 2424.9, respectively. Both sets of values tell the same story but, given that the range of possible intensity values in these images is [0, 255], the standard deviation values relate to this range much more intuitively than the variance. ■

EXAMPLE 2.12: Comparison of standard deviation values as measures of image intensity contrast.

As you will see in progressing through the book, concepts from probability play a central role in the development of image processing algorithms. For example, in Chapter 3 we use the probability measure in Eq. (2.6-42) to derive intensity transformation algorithms. In Chaper 5, we use probability and matrix formulations to develop image restoration algorithms. In Chapter 10, probability is used for image segmentation, and in Chapter 11 we use it for texture description. In Chapter 12, we derive optimum object recognition techniques based on a probabilistic formulation.



a b c

FIGURE 2.41 Images exhibiting (a) low contrast, (b) medium contrast, and (c) high contrast.

Thus far, we have addressed the issue of applying probability to a single random variable (intensity) over a single 2-D image. If we consider sequences of images, we may interpret the third variable as time. The tools needed to handle this added complexity are *stochastic* image processing techniques (the word *stochastic* is derived from a Greek word meaning roughly "to aim at a target," implying randomness in the outcome of the process). We can go a step further and consider an *entire* image (as opposed to a point) to be a spatial random event. The tools needed to handle formulations based on this concept are techniques from *random fields*. We give one example in Section 5.8 of how to treat entire images as random events, but further discussion of stochastic processes and random fields is beyond the scope of this book. The references at the end of this chapter provide a starting point for reading about these topics.

Summary

The material in this chapter is primarily background for subsequent discussions. Our treatment of the human visual system, although brief, provides a basic idea of the capabilities of the eye in perceiving pictorial information. The discussion on light and the electromagnetic spectrum is fundamental in understanding the origin of the many images we use in this book. Similarly, the image model developed in Section 2.3.4 is used in the Chapter 4 as the basis for an image enhancement technique called *homomorphic filtering*.

The sampling and interpolation ideas introduced in Section 2.4 are the foundation for many of the digitizing phenomena you are likely to encounter in practice. We will return to the issue of sampling and many of its ramifications in Chapter 4, after you have mastered the Fourier transform and the frequency domain.

The concepts introduced in Section 2.5 are the basic building blocks for processing techniques based on pixel neighborhoods. For example, as we show in the following chapter, and in Chapter 5, neighborhood processing methods are at the core of many image enhancement and restoration procedures. In Chapter 9, we use neighborhood operations for image morphology; in Chapter 10, we use them for image segmentation; and in Chapter 11 for image description. When applicable, neighborhood processing is favored in commercial applications of image processing because of their operational speed and simplicity of implementation in hardware and/or firmware.

The material in Section 2.6 will serve you well in your journey through the book. Although the level of the discussion was strictly introductory, you are now in a position to conceptualize what it means to process a digital image. As we mentioned in that section, the tools introduced there are expanded as necessary in the following chapters. Rather than dedicate an entire chapter or appendix to develop a comprehensive treatment of mathematical concepts in one place, you will find it considerably more meaningful to learn the necessary extensions of the mathematical tools from Section 2.6 in later chapters, in the context of how they are applied to solve problems in image processing.

References and Further Reading

Additional reading for the material in Section 2.1 regarding the structure of the human eye may be found in Atchison and Smith [2000] and Oyster [1999]. For additional reading on visual perception, see Regan [2000] and Gordon [1997]. The book by Hubel [1988] and the classic book by Cornsweet [1970] also are of interest. Born and Wolf [1999] is a basic reference that discusses light in terms of electromagnetic theory. Electromagnetic energy propagation is covered in some detail by Felsen and Marcuvitz [1994].

The area of image sensing is quite broad and very fast moving. An excellent source of information on optical and other imaging sensors is the Society for Optical Engineering (SPIE). The following are representative publications by the SPIE in this area: Blouke et al. [2001], Hoover and Doty [1996], and Freeman [1987].

The image model presented in Section 2.3.4 is from Oppenheim, Schafer, and Stockham [1968]. A reference for the illumination and reflectance values used in that section is the *IESNA Lighting Handbook* [2000]. For additional reading on image sampling and some of its effects, such as aliasing, see Bracewell [1995]. We discuss this topic in more detail in Chapter 4. The early experiments mentioned in Section 2.4.3 on perceived image quality as a function of sampling and quatization were reported by Huang [1965]. The issue of reducing the number of samples and intensity levels in an image while minimizing the ensuing degradation is still of current interest, as exemplified by Papamarkos and Atsalakis [2000]. For further reading on image shrinking and zooming, see Sid-Ahmed [1995], Unser et al. [1995], Umbaugh [2005], and Lehmann et al. [1999]. For further reading on the topics covered in Section 2.5, see Rosenfeld and Kak [1982], Marchand-Maillet and Sharaiha [2000], and Ritter and Wilson [2001].

Additional reading on linear systems in the context of image processing (Section 2.6.2) may be found in Castleman [1996]. The method of noise reduction by image averaging (Section 2.6.3) was first proposed by Kohler and Howell [1963]. See Peebles [1993] regarding the expected value of the mean and variance of a sum of random variables. Image subtraction (Section 2.6.3) is a generic image processing tool used widely for change detection. For image subtraction to make sense, it is necessary that the images being subtracted be registered or, alternatively, that any artifacts due to motion be identified. Two papers by Meijering et al. [1999, 2001] are illustrative of the types of techniques used to achieve these objectives.

A basic reference for the material in Section 2.6.4 is Cameron [2005]. For more advanced reading on this topic, see Tourlakis [2003]. For an introduction to fuzzy sets, see Section 3.8 and the corresponding references in Chapter 3. For further details on single-point and neighborhood processing (Section 2.6.5), see Sections 3.2 through 3.4 and the references on these topics in Chapter 3. For geometric spatial transformations, see Wolberg [1990].

Noble and Daniel [1988] is a basic reference for matrix and vector operations (Section 2.6.6). See Chapter 4 for a detailed discussion on the Fourier transform (Section 2.6.7), and Chapters 7, 8, and 11 for examples of other types of transforms used in digital image processing. Peebles [1993] is a basic introduction to probability and random variables (Section 2.6.8) and Papoulis [1991] is a more advanced treatment of this topic. For foundation material on the use of stochastic and random fields for image processing, see Rosenfeld and Kak [1982], Jähne [2002], and Won and Gray [2004].

For details of software implementation of many of the techniques illustrated in this chapter, see Gonzalez, Woods, and Eddins [2004].

Problems

Detailed solutions to the problems marked with a star can be found in the book Web site. The site also contains suggested projects based on the material in this chapter.

★**2.1** Using the background information provided in Section 2.1, and thinking purely in geometric terms, estimate the diameter of the smallest printed dot that the eye can discern if the page on which the dot is printed is 0.3 m away from the eyes. Assume for simplicity that the visual system ceases to detect the dot when the image of the dot on the fovea becomes smaller than the diameter of one receptor (cone) in that area of the retina. Assume further that the fovea can be

modeled as a circular array of diameter 1.5 mm and that the cones and spaces between the cones are distributed uniformly throughout this array.

2.2 When you enter a dark theater on a bright day, it takes an appreciable interval of time before you can see well enough to find an empty seat. Which of the visual processes explained in Section 2.1 is at play in this situation?

★2.3 Although it is not shown in Fig. 2.10, alternating current certainly is part of the electromagnetic spectrum. Commercial alternating current in the United States has a frequency of 77 Hz. What is the wavelength in meters of this component of the spectrum?

2.4 You are hired to design the front end of an imaging system for studying the boundary shapes of cells, bacteria, viruses, and protein. The front end consists, in this case, of the illumination source(s) and corresponding imaging camera(s). The diameters of circles required to enclose individual specimens in each of these categories are 25, 0.5, 0.05, and 0.005 μm, respectively.

(a) Can you solve the imaging aspects of this problem with a single sensor and camera? If your answer is yes, specify the illumination wavelength band and the type of camera needed. By "type," we mean the band of the electromagnetic spectrum to which the camera is most sensitive (e.g., infrared).

(b) If your answer in (a) is no, what type of illumination sources and corresponding imaging sensors would you recommend? Specify the light sources and cameras as requested in part (a). Use the *minimum* number of illumination sources and cameras needed to solve the problem.

By "solving the problem," we mean being able to detect circular details of diameter 25, 0.5, 0.05, and 0.005 μm, respectively.

2.5 A CCD camera chip of dimensions 14 × 14 mm, and having 2048 × 2048 elements, is focused on a square, flat area, located 0.5 m away. How many line pairs per mm will this camera be able to resolve? The camera is equipped with a 35-mm lens. (*Hint:* Model the imaging process as in Fig. 2.3, with the focal length of the camera lens substituting for the focal length of the eye.)

★2.6 An automobile manufacturer is automating the placement of certain components on the bumpers of a limited-edition line of sports cars. The components are color coordinated, so the robots need to know the color of each car in order to select the appropriate bumper component. Models come in only four colors: blue, green, red, and white. You are hired to propose a solution based on imaging. How would you solve the problem of automatically determining the color of each car, keeping in mind that *cost* is the most important consideration in your choice of components?

2.7 Suppose that a flat area with center at (x_0, y_0) is illuminated by a light source with intensity distribution

$$i(x, y) = Ke^{-[(x-x_0)^2+(y-y_0)^2]}$$

Assume for simplicity that the reflectance of the area is constant and equal to 1.0, and let $K = 255$. If the resulting image is digitized with k bits of intensity resolution, and the eye can detect an abrupt change of four shades of intensity between adjacent pixels, what value of k will cause visible false contouring?

2.8 Sketch the image in Problem 2.7 for $k = 4$.

★2.9 A common measure of transmission for digital data is the *baud rate*, defined as the number of bits transmitted per second. Generally, transmission is accomplished

in packets consisting of a start bit, a byte (8 bits) of information, and a stop bit. Using these facts, answer the following:

(a) How many minutes would it take to transmit a 2048 × 2048 image with 256 intensity levels using a 33.6K baud modem?

(b) What would the time be at 3000K baud, a representative medium speed of a phone DSL (Digital Subscriber Line) connection?

2.10 High-definition television (HDTV) generates images with 1080 horizontal TV lines interlaced (where every other line is painted on the tube face in each of two fields, each field being 1/60th of a second in duration). The width-to-height aspect ratio of the images is 16:9. The fact that the number of horizontal lines is fixed determines the vertical resolution of the images. A company has designed an image capture system that generates digital images from HDTV images. The resolution of each TV (horizontal) line in their system is in proportion to vertical resolution, with the proportion being the width-to-height ratio of the images. Each pixel in the color image has 24 bits of intensity resolution, 8 bits each for a red, a green, and a blue image. These three "primary" images form a color image. How many bits would it take to store a 90-minute HDTV movie?

★2.11 Consider the two image subsets, S_1 and S_2, shown in the following figure. For $V = \{1\}$, determine whether these two subsets are (a) 4-adjacent, (b) 8-adjacent, or (c) m-adjacent.

					S_1				S_2	
0	0	0	0	0	0	0	1	1	0	0
1	0	0	1	0	0	1	0	0	1	0
1	0	0	1	0	1	1	0	0	0	0
0	0	1	1	1	0	0	0	0	0	1
0	0	1	1	1	0	0	1	1	1	1

★2.12 Develop an algorithm for converting a one-pixel-thick 8-path to a 4-path.

2.13 Develop an algorithm for converting a one-pixel-thick m-path to a 4-path.

2.14 Refer to the discussion at the end of Section 2.5.2, where we defined the background as $(R_u)^c$, the complement of the union of all the regions in an image. In some applications, it is advantageous to define the background as the subset of pixels $(R_u)^c$ that are not region hole pixels (informally, think of holes as sets of background pixels surrounded by region pixels). How would you modify the definition to exclude hole pixels from $(R_u)^c$? An answer such as "the background is the subset of pixels of $(R_u)^c$ that are not hole pixels" is not acceptable. (*Hint:* Use the concept of connectivity.)

2.15 Consider the image segment shown.

★(a) Let $V = \{0, 1, 2\}$ and compute the lengths of the shortest 4-, 8-, and m-path between p and q. If a particular path does not exist between these two points, explain why.

(b) Repeat for $V = \{2, 3, 4\}$.

3	4	1	2	0
0	1	0	4	2 (q)
2	2	3	1	4
(p) 3	0	4	2	1
1	2	0	3	4

2.16 ★**(a)** Give the condition(s) under which the D_4 distance between two points p and q is equal to the shortest 4-path between these points.

(b) Is this path unique?

2.17 Repeat Problem 2.16 for the D_8 distance.

★**2.18** In the next chapter, we will deal with operators whose function is to compute the sum of pixel values in a small subimage area, S. Show that these are linear operators.

2.19 The median, ζ, of a set of numbers is such that half the values in the set are below ζ and the other half are above it. For example, the median of the set of values $\{2, 3, 8, 20, 21, 25, 31\}$ is 20. Show that an operator that computes the median of a subimage area, S, is nonlinear.

★**2.20** Prove the validity of Eqs. (2.6-6) and (2.6-7). [*Hint:* Start with Eq. (2.6-4) and use the fact that the expected value of a sum is the sum of the expected values.]

2.21 Consider two 8-bit images whose intensity levels span the full range from 0 to 255.

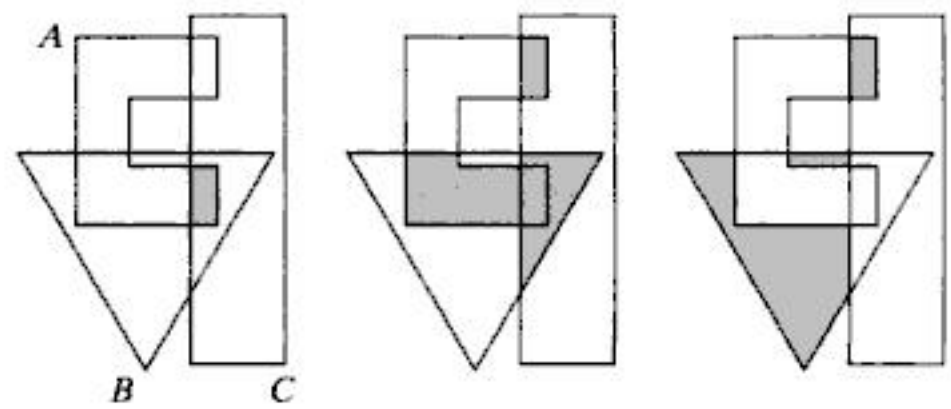
(a) Discuss the limiting effect of repeatedly subtracting image (2) from image (1). Assume that the result is represented also in eight bits.

(b) Would reversing the order of the images yield a different result?

★**2.22** Image subtraction is used often in industrial applications for detecting missing components in product assembly. The approach is to store a "golden" image that corresponds to a correct assembly; this image is then subtracted from incoming images of the same product. Ideally, the differences would be zero if the new products are assembled correctly. Difference images for products with missing components would be nonzero in the area where they differ from the golden image. What conditions do you think have to be met in practice for this method to work?

2.23 ★**(a)** With reference to Fig. 2.31, sketch the set $(A^c - B) \cup (B - A)$

(b) Give expressions for the sets shown shaded in the following figure in terms of sets A, B, and C. The shaded areas in each figure constitute one set, so give one expression for each of the three figures.



2.24 What would be the equations analogous to Eqs. (2.6-24) and (2.6-25) that would result from using triangular instead of quadrilateral regions?

2.25 Prove that the Fourier kernels in Eqs. (2.6-34) and (2.6-35) are separable and symmetric. What are the advantages of using separable transformations on images?

★**2.26** Show that 2-D transforms with separable, symmetric kernels can be computed by (1) computing 1-D transforms along the individual rows (columns) of the input, followed by (2) computing 1-D transforms along the columns (rows) of the result from step (1).

2.27 A plant produces a line of translucent miniature polymer squares. Stringent quality requirements dictate 100% visual inspection, and the plant manager finds the use of human inspectors increasingly expensive. Inspection is semiautomated. At each inspection station, a robotic mechanism places each polymer square over a light located under an optical system that produces a magnified image of the square. The image completely fills a viewing screen measuring 80 × 80 mm. Defects appear as dark circular blobs, and the inspector's job is to look at the screen and reject any sample that has one or more such dark blobs with a diameter of 0.8 mm or larger, as measured on the scale of the screen. The manager believes that if she can find a way to automate the process completely, she will increase profits by 50%. She also believes that success in this project will aid her climb up the corporate ladder. After much investigation, the manager decides that the way to solve the problem is to view each inspection screen with a CCD TV camera and feed the output of the camera into an image processing system capable of detecting the blobs, measuring their diameter, and activating the accept/reject buttons previously operated by an inspector. She is able to find a system that can do the job, as long as the smallest defect occupies an area of at least 2 × 2 pixels in the digital image. The manager hires you to help her specify the camera and lens system, but requires that you use off-the-shelf components. For the lenses, assume that this constraint means any integer multiple of 25 mm or 35 mm, up to 200 mm. For the cameras, it means resolutions of 512 × 512, 1024 × 1024, or 2048 × 2048 pixels. The *individual* imaging elements in these cameras are squares measuring 8 × 8 μm, and the spaces between imaging elements are 2 μm. For this application, the cameras cost much more than the lenses, so the problem should be solved with the lowest-resolution camera possible, based on the choice of lenses. As a consultant, you are to provide a written recommendation, showing in reasonable detail the analysis that led to your conclusion. Use the same imaging geometry suggested in Problem 2.5.

3 *Intensity Transformations and Spatial Filtering*

It makes all the difference whether one sees darkness through the light or brightness through the shadows.

David Lindsay

Preview

The term *spatial domain* refers to the image plane itself, and image processing methods in this category are based on direct manipulation of pixels in an image. This is in contrast to image processing in a *transform domain* which, as introduced in Section 2.6.7 and discussed in more detail in Chapter 4, involves first transforming an image into the transform domain, doing the processing there, and obtaining the inverse transform to bring the results back into the spatial domain. Two principal categories of spatial processing are intensity transformations and spatial filtering. As you will learn in this chapter, intensity transformations operate on single pixels of an image, principally for the purpose of contrast manipulation and image thresholding. Spatial filtering deals with performing operations, such as image sharpening, by working in a neighborhood of every pixel in an image. In the sections that follow, we discuss a number of "classical" techniques for intensity transformations and spatial filtering. We also discuss in some detail fuzzy techniques that allow us to incorporate imprecise, knowledge-based information in the formulation of intensity transformations and spatial filtering algorithms.

3.1 Background

3.1.1 The Basics of Intensity Transformations and Spatial Filtering

All the image processing techniques discussed in this section are implemented in the *spatial domain*, which we know from the discussion in Section 2.4.2 is simply the plane containing the pixels of an image. As noted in Section 2.6.7, spatial domain techniques operate directly on the pixels of an image as opposed, for example, to the frequency domain (the topic of Chapter 4) in which operations are performed on the Fourier transform of an image, rather than on the image itself. As you will learn in progressing through the book, some image processing tasks are easier or more meaningful to implement in the spatial domain while others are best suited for other approaches. Generally, spatial domain techniques are more efficient computationally and require less processing resources to implement.

The spatial domain processes we discuss in this chapter can be denoted by the expression

$$g(x, y) = T[f(x, y)] \tag{3.1-1}$$

where $f(x, y)$ is the input image, $g(x, y)$ is the output image, and T is an operator on f defined over a neighborhood of point (x, y). The operator can apply to a single image (our principal focus in this chapter) or to a set of images, such as performing the pixel-by-pixel sum of a sequence of images for noise reduction, as discussed in Section 2.6.3. Figure 3.1 shows the basic implementation of Eq. (3.1-1) on a single image. The point (x, y) shown is an arbitrary location in the image, and the small region shown containing the point is a neighborhood of (x, y), as explained in Section 2.6.5. Typically, the neighborhood is rectangular, centered on (x, y), and much smaller in size than the image.

Other neighborhood shapes, such as digital approximations to circles, are used sometimes, but rectangular shapes are by far the most prevalent because they are much easier to implement computationally.

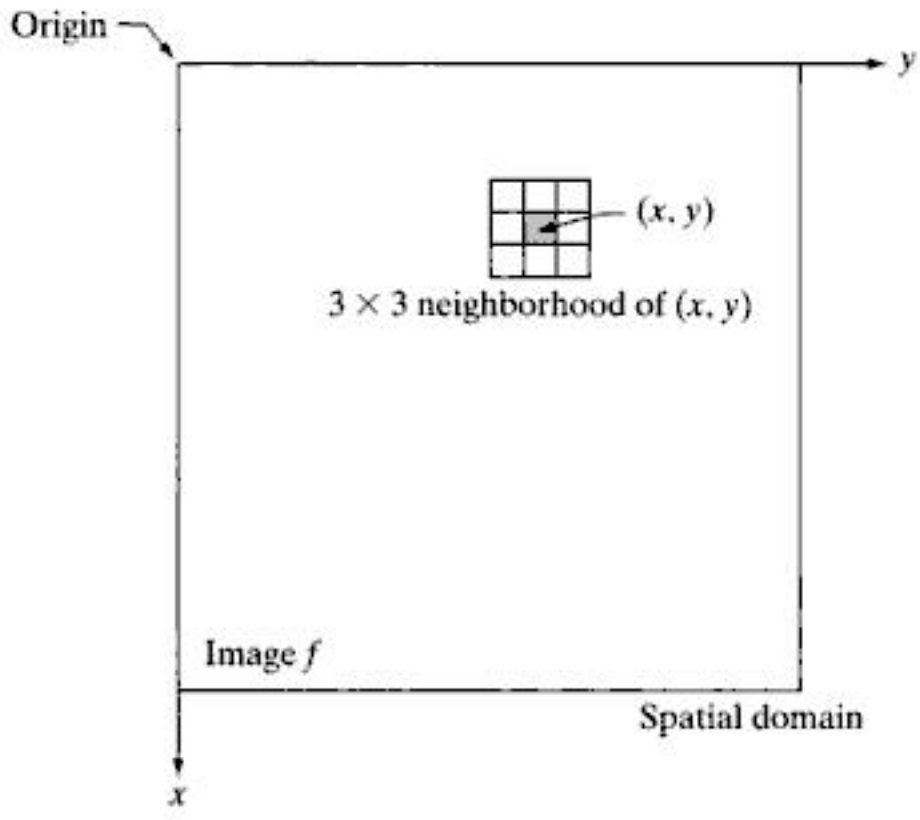


FIGURE 3.1 A 3 × 3 neighborhood about a point (x, y) in an image in the spatial domain. The neighborhood is moved from pixel to pixel in the image to generate an output image.

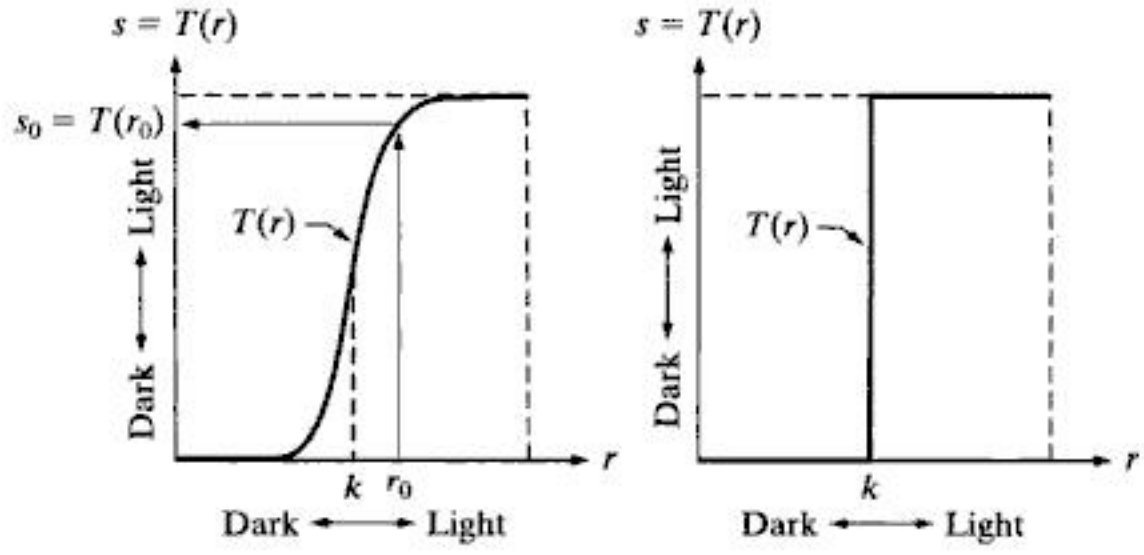
The process that Fig. 3.1 illustrates consists of moving the origin of the neighborhood from pixel to pixel and applying the operator T to the pixels in the neighborhood to yield the output at that location. Thus, for any specific location (x, y), the value of the output image g at those coordinates is equal to the result of applying T to the neighborhood with origin at (x, y) in f. For example, suppose that the neighborhood is a square of size 3×3, and that operator T is defined as "compute the average intensity of the neighborhood." Consider an arbitrary location in an image, say (100, 150). Assuming that the origin of the neighborhood is at its center, the result, $g(100, 150)$, at that location is computed as the sum of $f(100, 150)$ and its 8-neighbors, divided by 9 (i.e., the average intensity of the pixels encompassed by the neighborhood). The origin of the neighborhood is then moved to the next location and the procedure is repeated to generate the next value of the output image g. Typically, the process starts at the top left of the input image and proceeds pixel by pixel in a horizontal scan, one row at a time. When the origin of the neighborhood is at the border of the image, part of the neighborhood will reside outside the image. The procedure is either to ignore the outside neighbors in the computations specified by T, or to pad the image with a border of 0s or some other specified intensity values. The thickness of the padded border depends on the size of the neighborhood. We will return to this issue in Section 3.4.1.

As we discuss in detail in Section 3.4, the procedure just described is called *spatial filtering*, in which the neighborhood, along with a predefined operation, is called a *spatial filter* (also referred to as a *spatial mask*, *kernel*, *template*, or *window*). The type of operation performed in the neighborhood determines the nature of the filtering process.

The smallest possible neighborhood is of size 1×1. In this case, g depends only on the value of f at a single point (x, y) and T in Eq. (3.1-1) becomes an *intensity* (also called *gray-level* or *mapping*) *transformation function* of the form

$$s = T(r) \tag{3.1-2}$$

where, for simplicity in notation, s and r are variables denoting, respectively, the intensity of g and f at any point (x, y). For example, if $T(r)$ has the form in Fig. 3.2(a), the effect of applying the transformation to every pixel of f to generate the corresponding pixels in g would be to produce an image of



a b

FIGURE 3.2 Intensity transformation functions. (a) Contrast-stretching function. (b) Thresholding function.

higher contrast than the original by darkening the intensity levels below k and brightening the levels above k. In this technique, sometimes called *contrast stretching* (see Section 3.2.4), values of r lower than k are compressed by the transformation function into a narrow range of s, toward black. The opposite is true for values of r higher than k. Observe how an intensity value r_0 is mapped to obtain the corresponding value s_0. In the limiting case shown in Fig. 3.2(b), $T(r)$ produces a two-level (binary) image. A mapping of this form is called a *thresholding* function. Some fairly simple, yet powerful, processing approaches can be formulated with intensity transformation functions. In this chapter, we use intensity transformations principally for image enhancement. In Chapter 10, we use them for image segmentation. Approaches whose results depend only on the intensity at a point sometimes are called *point processing* techniques, as opposed to the *neighborhood processing* techniques discussed earlier in this section.

3.1.2 About the Examples in This Chapter

Although intensity transformations and spatial filtering span a broad range of applications, most of the examples in this chapter are applications to image enhancement. *Enhancement* is the process of manipulating an image so that the result is more suitable than the original for a specific application. The word *specific* is important here because it establishes at the outset that enhancement techniques are problem oriented. Thus, for example, a method that is quite useful for enhancing X-ray images may not be the best approach for enhancing satellite images taken in the infrared band of the electromagnetic spectrum. There is no general "theory" of image enhancement. When an image is processed for visual interpretation, the viewer is the ultimate judge of how well a particular method works. When dealing with machine perception, a given technique is easier to quantify. For example, in an automated character-recognition system, the most appropriate enhancement method is the one that results in the best recognition rate, leaving aside other considerations such as computational requirements of one method over another.

Regardless of the application or method used, however, image enhancement is one of the most visually appealing areas of image processing. By its very nature, beginners in image processing generally find enhancement applications interesting and relatively simple to understand. Therefore, using examples from image enhancement to illustrate the spatial processing methods developed in this chapter not only saves having an extra chapter in the book dealing with image enhancement but, more importantly, is an effective approach for introducing newcomers to the details of processing techniques in the spatial domain. As you will see as you progress through the book, the basic material developed in this chapter is applicable to a much broader scope than just image enhancement.

3.2 Some Basic Intensity Transformation Functions

Intensity transformations are among the simplest of all image processing techniques. The values of pixels, before and after processing, will be denoted by r and s, respectively. As indicated in the previous section, these values are related

by an expression of the form $s = T(r)$, where T is a transformation that maps a pixel value r into a pixel value s. Because we are dealing with digital quantities, values of a transformation function typically are stored in a one-dimensional array and the mappings from r to s are implemented via table lookups. For an 8-bit environment, a lookup table containing the values of T will have 256 entries.

As an introduction to intensity transformations, consider Fig. 3.3, which shows three basic types of functions used frequently for image enhancement: linear (negative and identity transformations), logarithmic (log and inverse-log transformations), and power-law (nth power and nth root transformations). The identity function is the trivial case in which output intensities are identical to input intensities. It is included in the graph only for completeness.

3.2.1 Image Negatives

The negative of an image with intensity levels in the range $[0, L-1]$ is obtained by using the negative transformation shown in Fig. 3.3, which is given by the expression

$$s = L - 1 - r \tag{3.2-1}$$

Reversing the intensity levels of an image in this manner produces the equivalent of a photographic negative. This type of processing is particularly suited for enhancing white or gray detail embedded in dark regions of an

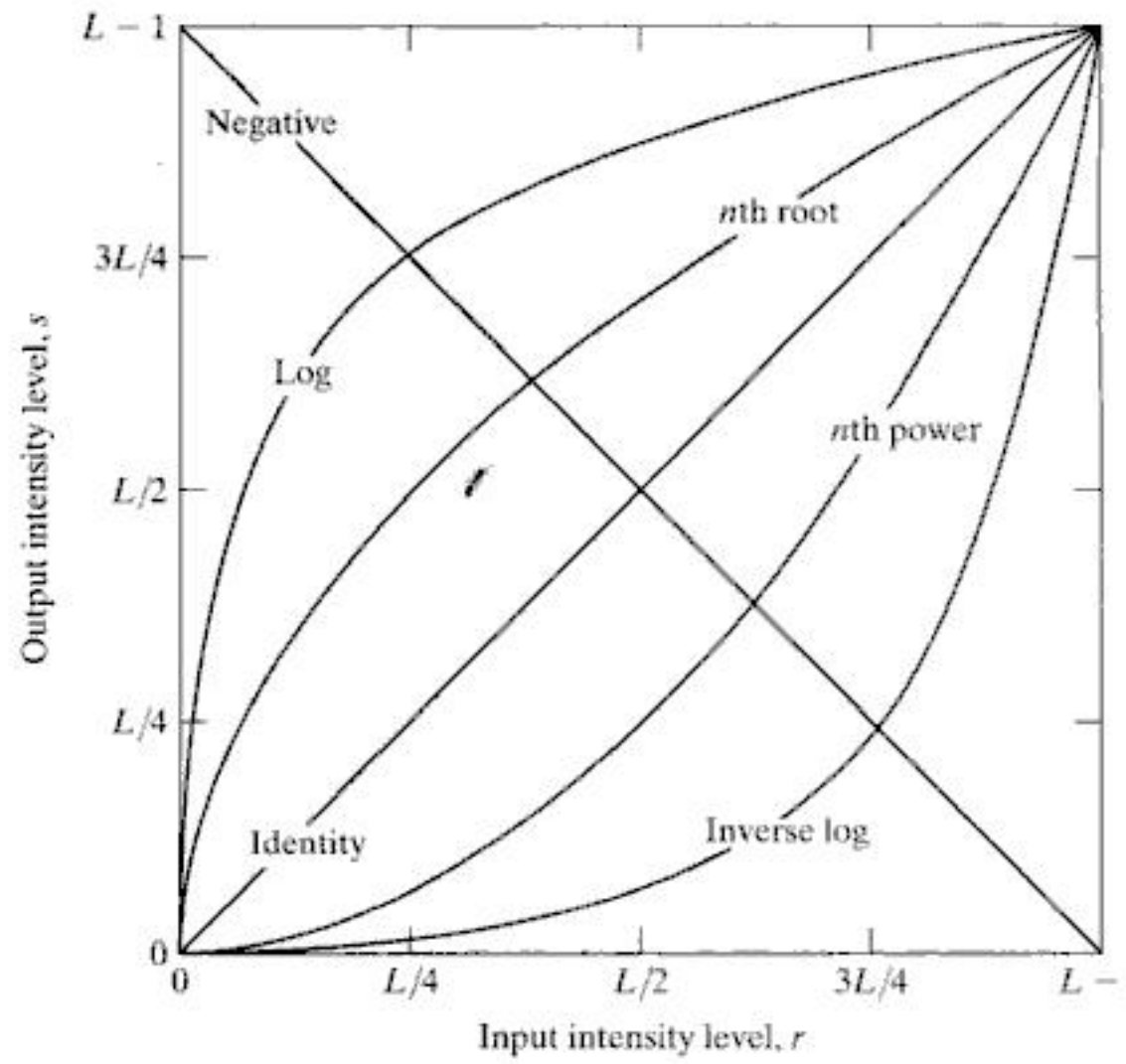
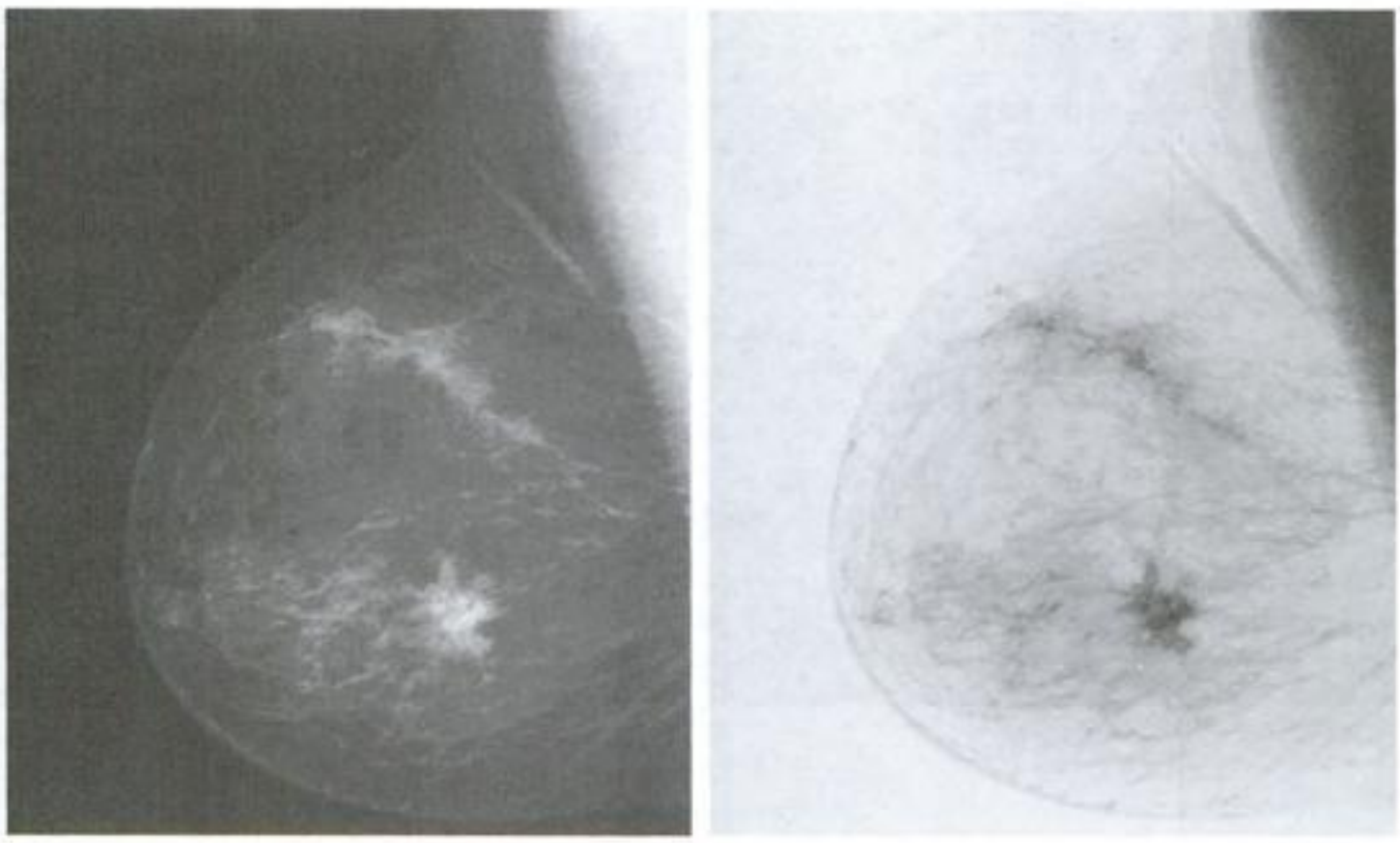


FIGURE 3.3 Some basic intensity transformation functions. All curves were scaled to fit in the range shown.



a b

FIGURE 3.4
(a) Original digital mammogram. (b) Negative image obtained using the negative transformation in Eq. (3.2-1). (Courtesy of G.E. Medical Systems.)

image, especially when the black areas are dominant in size. Figure 3.4 shows an example. The original image is a digital mammogram showing a small lesion. In spite of the fact that the visual content is the same in both images, note how much easier it is to analyze the breast tissue in the negative image in this particular case.

3.2.2 Log Transformations

The general form of the log transformation in Fig. 3.3 is

$$s = c\log(1 + r) \tag{3.2-2}$$

where c is a constant, and it is assumed that $r \geq 0$. The shape of the log curve in Fig. 3.3 shows that this transformation maps a narrow range of low intensity values in the input into a wider range of output levels. The opposite is true of higher values of input levels. We use a transformation of this type to expand the values of dark pixels in an image while compressing the higher-level values. The opposite is true of the inverse log transformation.

Any curve having the general shape of the log functions shown in Fig. 3.3 would accomplish this spreading/compressing of intensity levels in an image, but the power-law transformations discussed in the next section are much more versatile for this purpose. The log function has the important characteristic that it compresses the dynamic range of images with large variations in pixel values. A classic illustration of an application in which pixel values have a large dynamic range is the Fourier spectrum, which will be discussed in Chapter 4. At the moment, we are concerned only with the image characteristics of spectra. It is not unusual to encounter spectrum values that range from 0 to 10^6 or higher. While processing numbers such as these presents no problems for a computer, image display systems generally will not be able to reproduce

a b

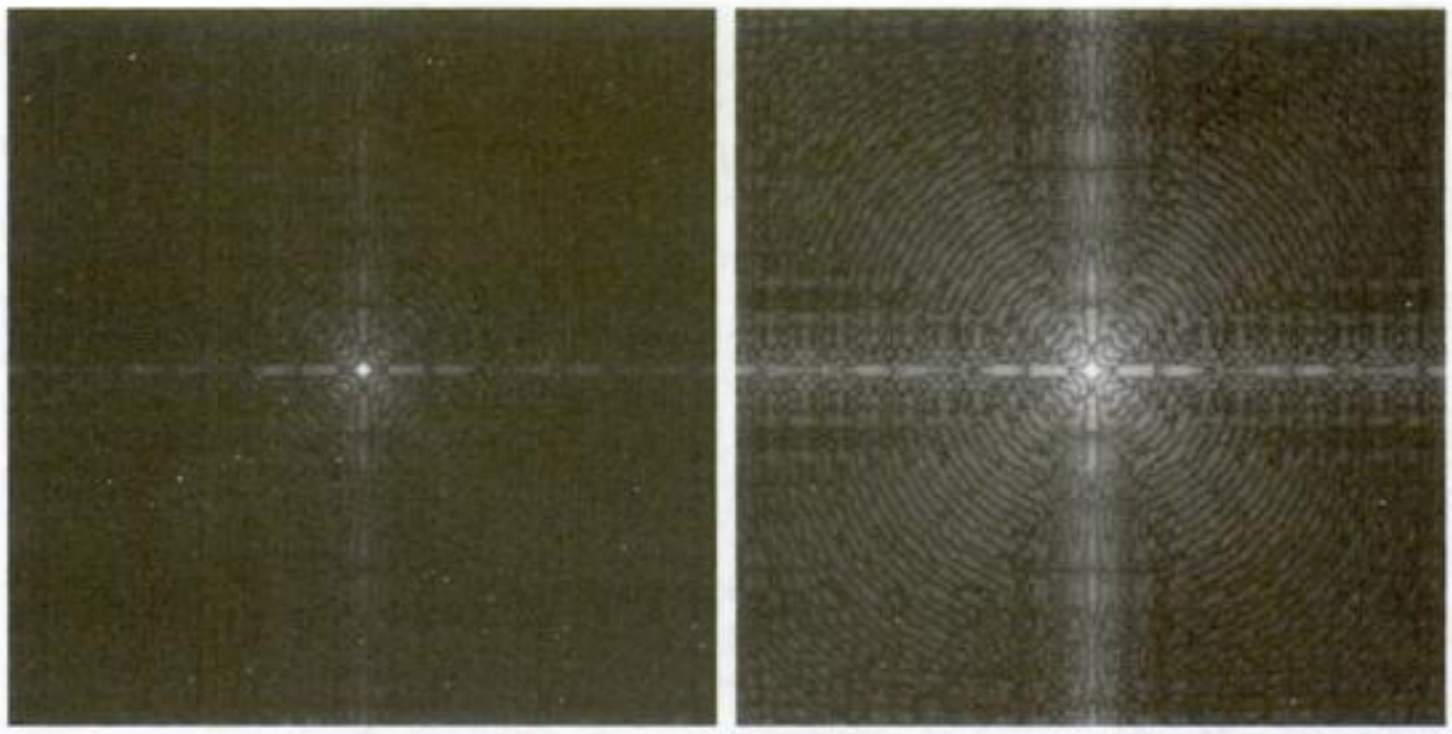


FIGURE 3.5
(a) Fourier spectrum.
(b) Result of applying the log transformation in Eq. (3.2-2) with $c = 1$.

faithfully such a wide range of intensity values. The net effect is that a significant degree of intensity detail can be lost in the display of a typical Fourier spectrum.

As an illustration of log transformations, Fig. 3.5(a) shows a Fourier spectrum with values in the range 0 to 1.5×10^6. When these values are scaled linearly for display in an 8-bit system, the brightest pixels will dominate the display, at the expense of lower (and just as important) values of the spectrum. The effect of this dominance is illustrated vividly by the relatively small area of the image in Fig. 3.5(a) that is not perceived as black. If, instead of displaying the values in this manner, we first apply Eq. (3.2-2) (with $c = 1$ in this case) to the spectrum values, then the range of values of the result becomes 0 to 6.2, which is more manageable. Figure 3.5(b) shows the result of scaling this new range linearly and displaying the spectrum in the same 8-bit display. The wealth of detail visible in this image as compared to an unmodified display of the spectrum is evident from these pictures. Most of the Fourier spectra seen in image processing publications have been scaled in just this manner.

3.2.3 Power-Law (Gamma) Transformations

Power-law transformations have the basic form

$$s = cr^{\gamma} \tag{3.2-3}$$

where c and γ are positive constants. Sometimes Eq. (3.2-3) is written as $s = c(r + \varepsilon)^{\gamma}$ to account for an offset (that is, a measurable output when the input is zero). However, offsets typically are an issue of display calibration and as a result they are normally ignored in Eq. (3.2-3). Plots of s versus r for various values of γ are shown in Fig. 3.6. As in the case of the log transformation, power-law curves with fractional values of γ map a narrow range of dark input values into a wider range of output values, with the opposite being true for higher values of input levels. Unlike the log function, however, we notice

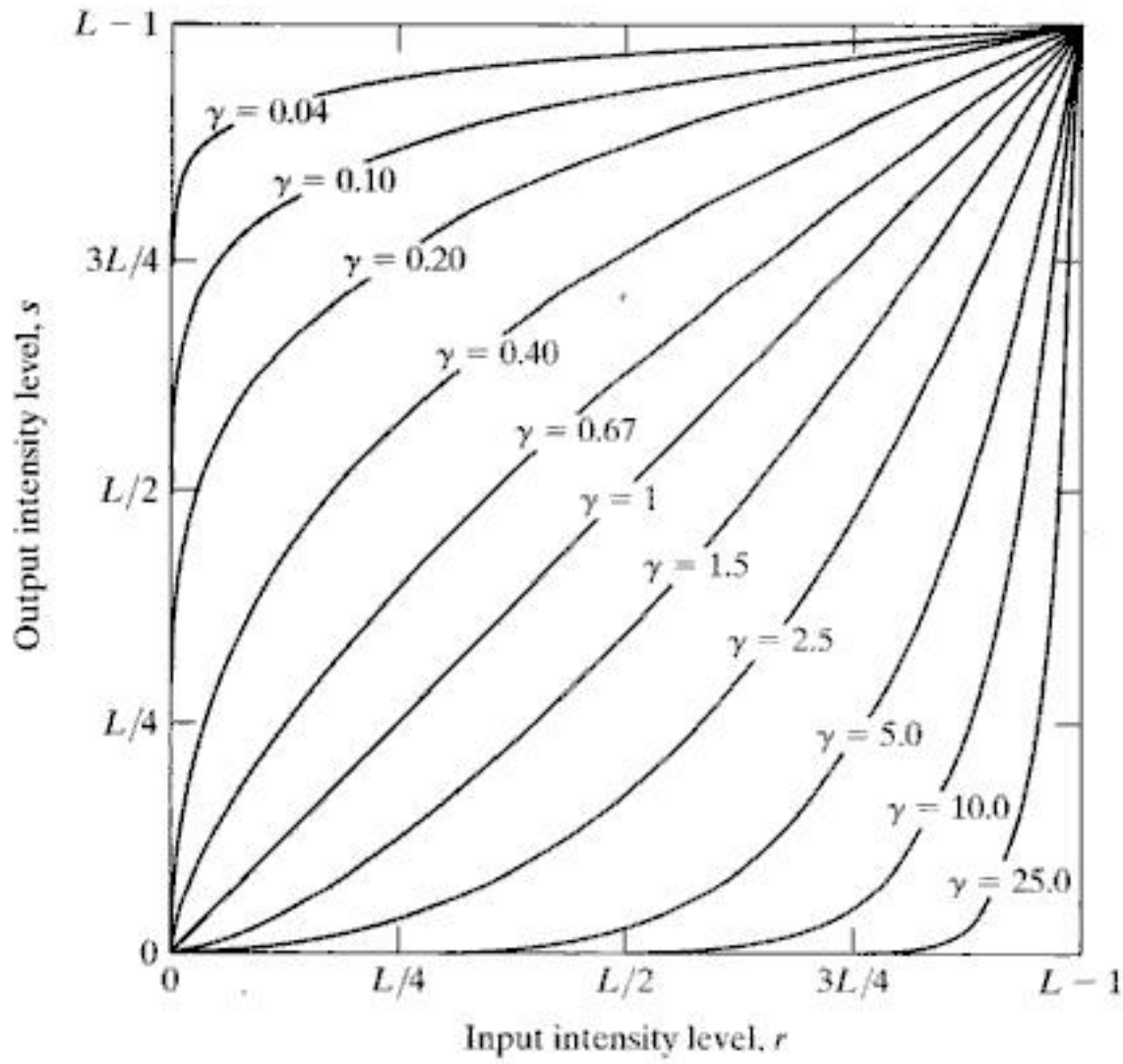
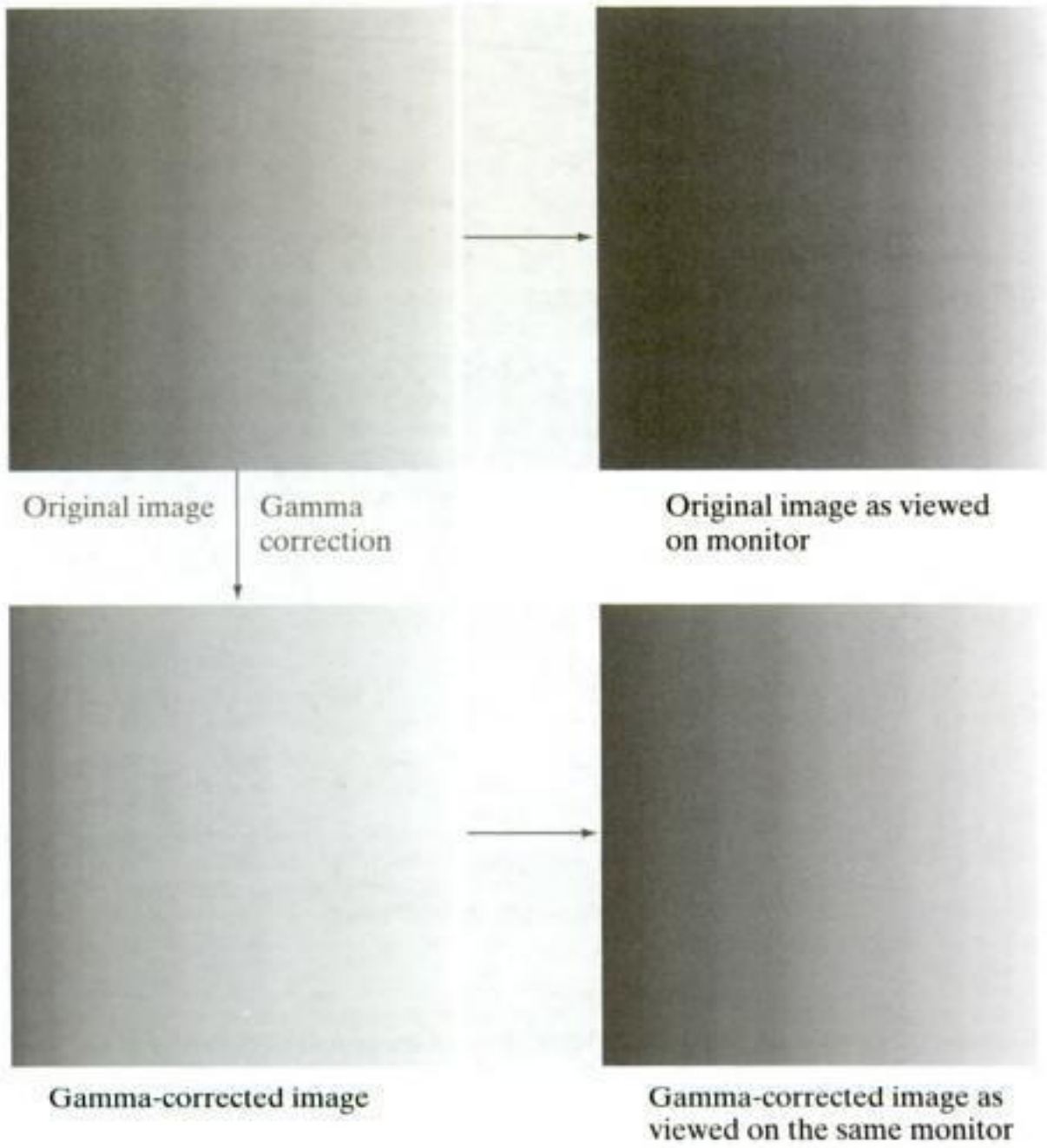


FIGURE 3.6 Plots of the equation $s = cr^{\gamma}$ for various values of γ ($c = 1$ in all cases). All curves were scaled to fit in the range shown.

here a family of possible transformation curves obtained simply by varying γ. As expected, we see in Fig. 3.6 that curves generated with values of $\gamma > 1$ have exactly the opposite effect as those generated with values of $\gamma < 1$. Finally, we note that Eq. (3.2-3) reduces to the identity transformation when $c = \gamma = 1$.

A variety of devices used for image capture, printing, and display respond according to a power law. By convention, the exponent in the power-law equation is referred to as *gamma* [hence our use of this symbol in Eq. (3.2-3)]. The process used to correct these power-law response phenomena is called *gamma correction*. For example, cathode ray tube (CRT) devices have an intensity-to-voltage response that is a power function, with exponents varying from approximately 1.8 to 2.5. With reference to the curve for $\gamma = 2.5$ in Fig. 3.6, we see that such display systems would tend to produce images that are darker than intended. This effect is illustrated in Fig. 3.7. Figure 3.7(a) shows a simple intensity-ramp image input into a monitor. As expected, the output of the monitor appears darker than the input, as Fig. 3.7(b) shows. Gamma correction in this case is straightforward. All we need to do is preprocess the input image before inputting it into the monitor by performing the transformation $s = r^{1/2.5} = r^{0.4}$. The result is shown in Fig. 3.7(c). When input into the same monitor, this gamma-corrected input produces an output that is close in appearance to the original image, as Fig. 3.7(d) shows. A similar analysis would apply to other imaging devices such as scanners and printers. The only difference would be the device-dependent value of gamma (Poynton [1996]).



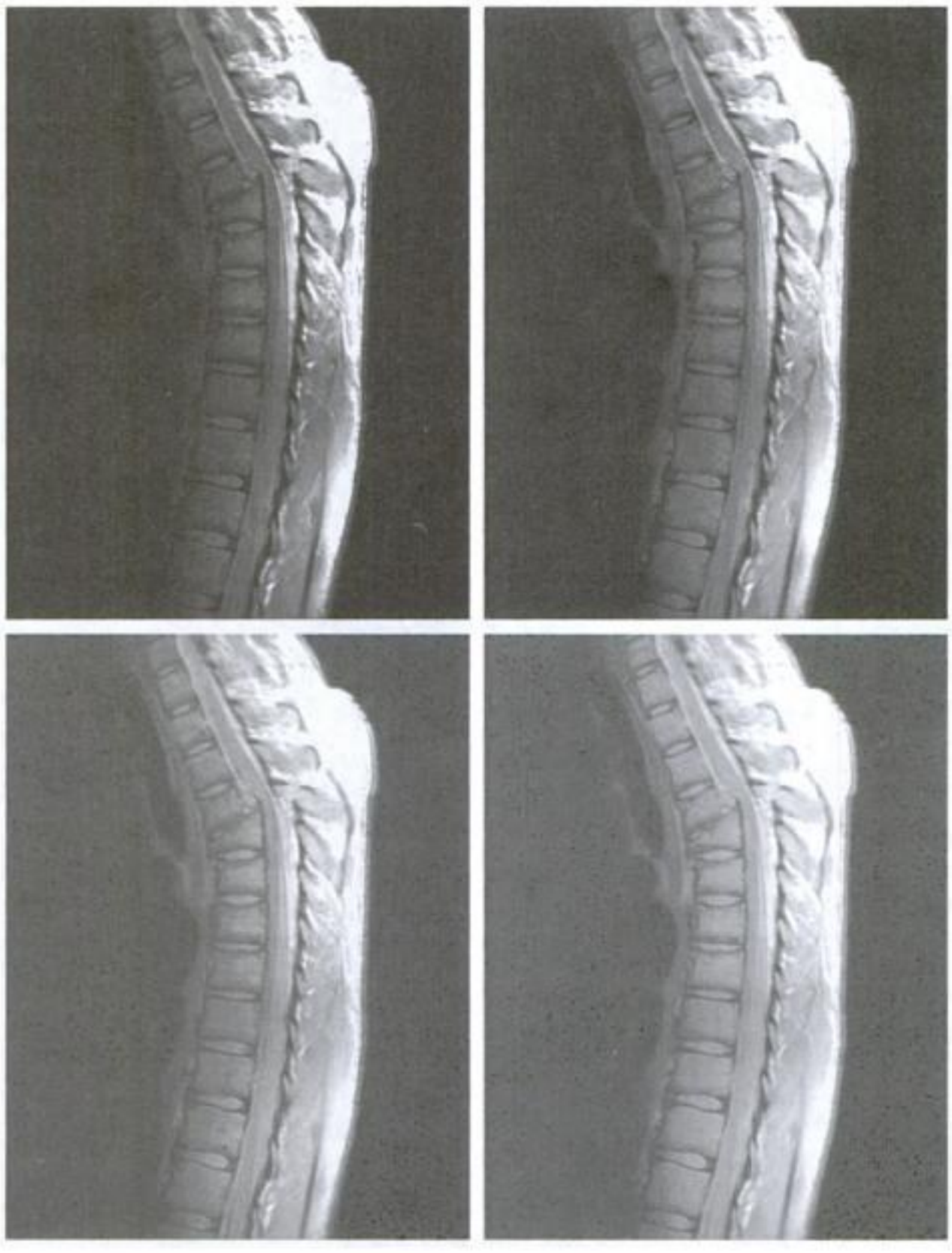
a b
c d

FIGURE 3.7
(a) Intensity ramp image. (b) Image as viewed on a simulated monitor with a gamma of 2.5. (c) Gamma-corrected image. (d) Corrected image as viewed on the same monitor. Compare (d) and (a).

Gamma correction is important if displaying an image accurately on a computer screen is of concern. Images that are not corrected properly can look either bleached out, or, what is more likely, too dark. Trying to reproduce colors accurately also requires some knowledge of gamma correction because varying the value of gamma changes not only the intensity, but also the ratios of red to green to blue in a color image. Gamma correction has become increasingly important in the past few years, as the use of digital images for commercial purposes over the Internet has increased. It is not unusual that images created for a popular Web site will be viewed by millions of people, the majority of whom will have different monitors and/or monitor settings. Some computer systems even have partial gamma correction built in. Also, current image standards do not contain the value of gamma with which an image was created, thus complicating the issue further. Given these constraints, a reasonable approach when storing images in a Web site is to preprocess the images with a gamma that represents an "average" of the types of monitors and computer systems that one expects in the open market at any given point in time.

EXAMPLE 3.1: Contrast enhancement using power-law transformations.

■ In addition to gamma correction, power-law transformations are useful for general-purpose contrast manipulation. Figure 3.8(a) shows a magnetic resonance image (MRI) of an upper thoracic human spine with a fracture dislocation and spinal cord impingement. The fracture is visible near the vertical center of the spine, approximately one-fourth of the way down from the top of the picture. Because the given image is predominantly dark, an expansion of intensity levels is desirable. This can be accomplished with a power-law transformation with a fractional exponent. The other images shown in the figure were obtained by processing Fig. 3.8(a) with the power-law transformation



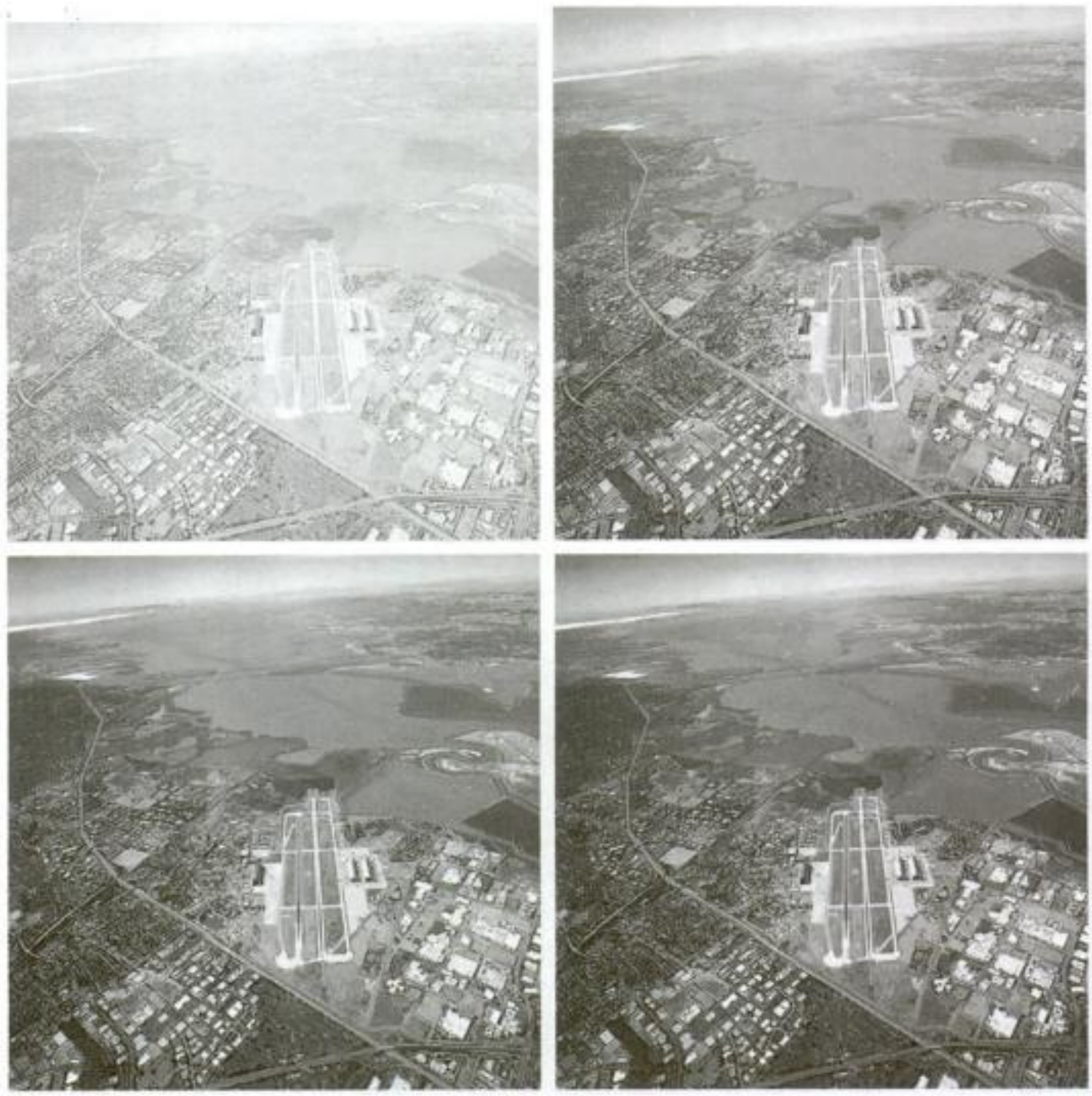
a b
c d

FIGURE 3.8 (a) Magnetic resonance image (MRI) of a fractured human spine. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 0.6, 0.4$, and 0.3, respectively. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

function of Eq. (3.2-3). The values of gamma corresponding to images (b) through (d) are 0.6, 0.4, and 0.3, respectively (the value of c was 1 in all cases). We note that, as gamma decreased from 0.6 to 0.4, more detail became visible. A further decrease of gamma to 0.3 enhanced a little more detail in the background, but began to reduce contrast to the point where the image started to have a very slight "washed-out" appearance, especially in the background. By comparing all results, we see that the best enhancement in terms of contrast and discernable detail was obtained with $\gamma = 0.4$. A value of $\gamma = 0.3$ is an approximate limit below which contrast in this particular image would be reduced to an unacceptable level. ■

EXAMPLE 3.2: Another illustration of power-law transformations.

■ Figure 3.9(a) shows the opposite problem of Fig. 3.8(a). The image to be processed now has a washed-out appearance, indicating that a compression of intensity levels is desirable. This can be accomplished with Eq. (3.2-3) using values of γ greater than 1. The results of processing Fig. 3.9(a) with $\gamma = 3.0$, 4.0, and 5.0 are shown in Figs. 3.9(b) through (d). Suitable results were obtained with gamma values of 3.0 and 4.0, the latter having a slightly



a b
c d

FIGURE 3.9
(a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 3.0$, 4.0, and 5.0, respectively. (Original image for this example courtesy of NASA.)

more appealing appearance because it has higher contrast. The result obtained with $\gamma = 5.0$ has areas that are too dark, in which some detail is lost. The dark region to the left of the main road in the upper left quadrant is an example of such an area. ■

3.2.4 Piecewise-Linear Transformation Functions

A complementary approach to the methods discussed in the previous three sections is to use piecewise linear functions. The principal advantage of piecewise linear functions over the types of functions we have discussed thus far is that the form of piecewise functions can be arbitrarily complex. In fact, as you will see shortly, a practical implementation of some important transformations can be formulated only as piecewise functions. The principal disadvantage of piecewise functions is that their specification requires considerably more user input.

Contrast stretching

One of the simplest piecewise linear functions is a contrast-stretching transformation. Low-contrast images can result from poor illumination, lack of dynamic range in the imaging sensor, or even the wrong setting of a lens aperture during image acquisition. *Contrast stretching* is a process that expands the range of intensity levels in an image so that it spans the full intensity range of the recording medium or display device.

Figure 3.10(a) shows a typical transformation used for contrast stretching. The locations of points (r_1, s_1) and (r_2, s_2) control the shape of the transformation function. If $r_1 = s_1$ and $r_2 = s_2$, the transformation is a linear function that produces no changes in intensity levels. If $r_1 = r_2$, $s_1 = 0$ and $s_2 = L - 1$, the transformation becomes a *thresholding function* that creates a binary image, as illustrated in Fig. 3.2(b). Intermediate values of (r_1, s_1) and (r_2, s_2) produce various degrees of spread in the intensity levels of the output image, thus affecting its contrast. In general, $r_1 \leq r_2$ and $s_1 \leq s_2$ is assumed so that the function is single valued and monotonically increasing. This condition preserves the order of intensity levels, thus preventing the creation of intensity artifacts in the processed image.

Figure 3.10(b) shows an 8-bit image with low contrast. Figure 3.10(c) shows the result of contrast stretching, obtained by setting $(r_1, s_1) = (r_{\min}, 0)$ and $(r_2, s_2) = (r_{\max}, L - 1)$, where $r_{\min}$ and $r_{\max}$ denote the minimum and maximum intensity levels in the image, respectively. Thus, the transformation function stretched the levels linearly from their original range to the full range $[0, L - 1]$. Finally, Fig. 3.10(d) shows the result of using the thresholding function defined previously, with $(r_1, s_1) = (m, 0)$ and $(r_2, s_2) = (m, L - 1)$, where m is the mean intensity level in the image. The original image on which these results are based is a scanning electron microscope image of pollen, magnified approximately 700 times.

Intensity-level slicing

Highlighting a specific range of intensities in an image often is of interest. Applications include enhancing features such as masses of water in satellite imagery and enhancing flaws in X-ray images. The process, often called *intensity-level*

a b
c d

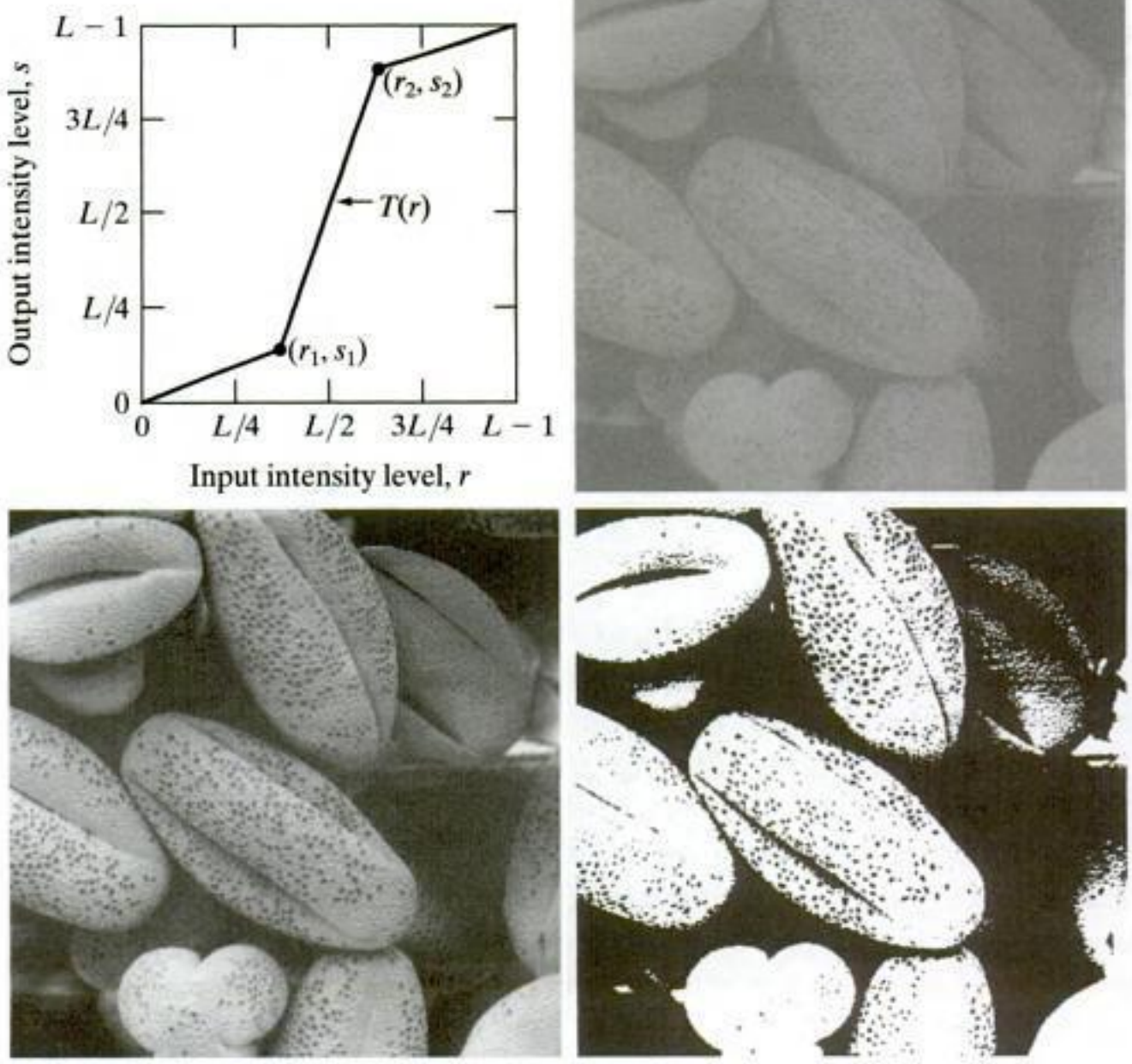


FIGURE 3.10 Contrast stretching. (a) Form of transformation function. (b) A low-contrast image. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

slicing, can be implemented in several ways, but most are variations of two basic themes. One approach is to display in one value (say, white) all the values in the range of interest and in another (say, black) all other intensities. This transformation, shown in Fig. 3.11(a), produces a binary image. The second approach, based on the transformation in Fig. 3.11(b), brightens (or darkens) the desired range of intensities but leaves all other intensity levels in the image unchanged.

a b

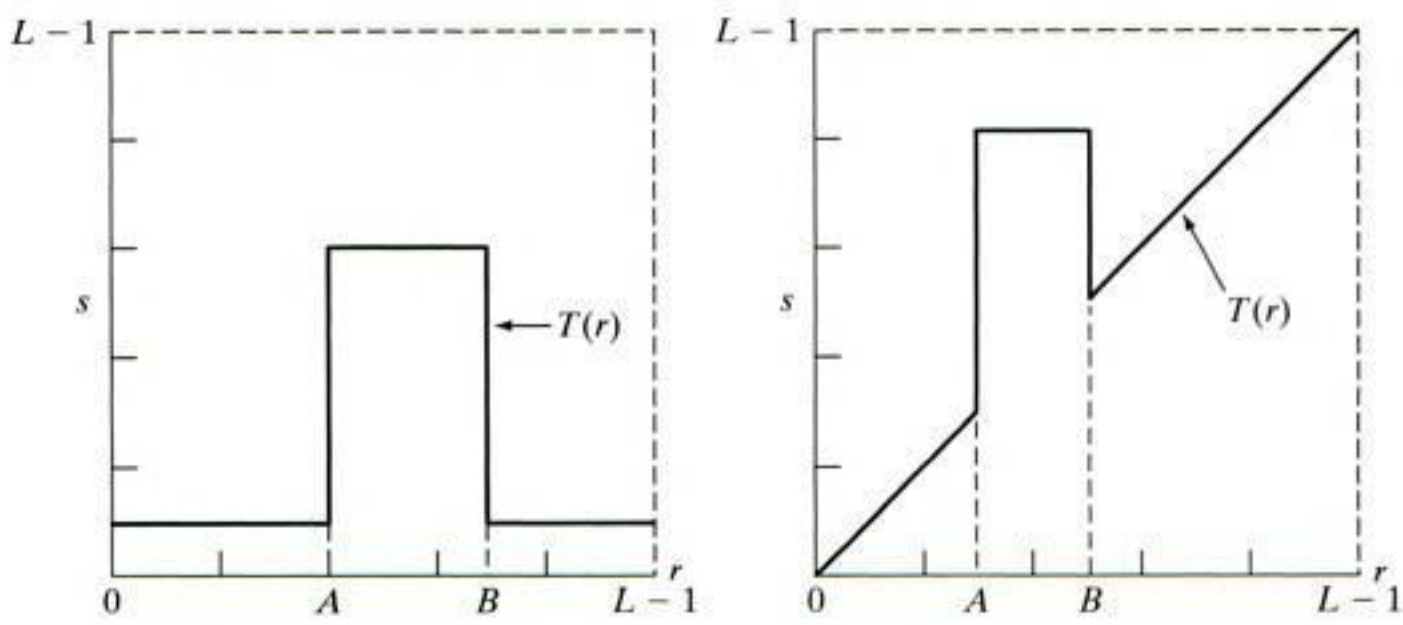


FIGURE 3.11 (a) This transformation highlights intensity range $[A, B]$ and reduces all other intensities to a lower level. (b) This transformation highlights range $[A, B]$ and preserves all other intensity levels.

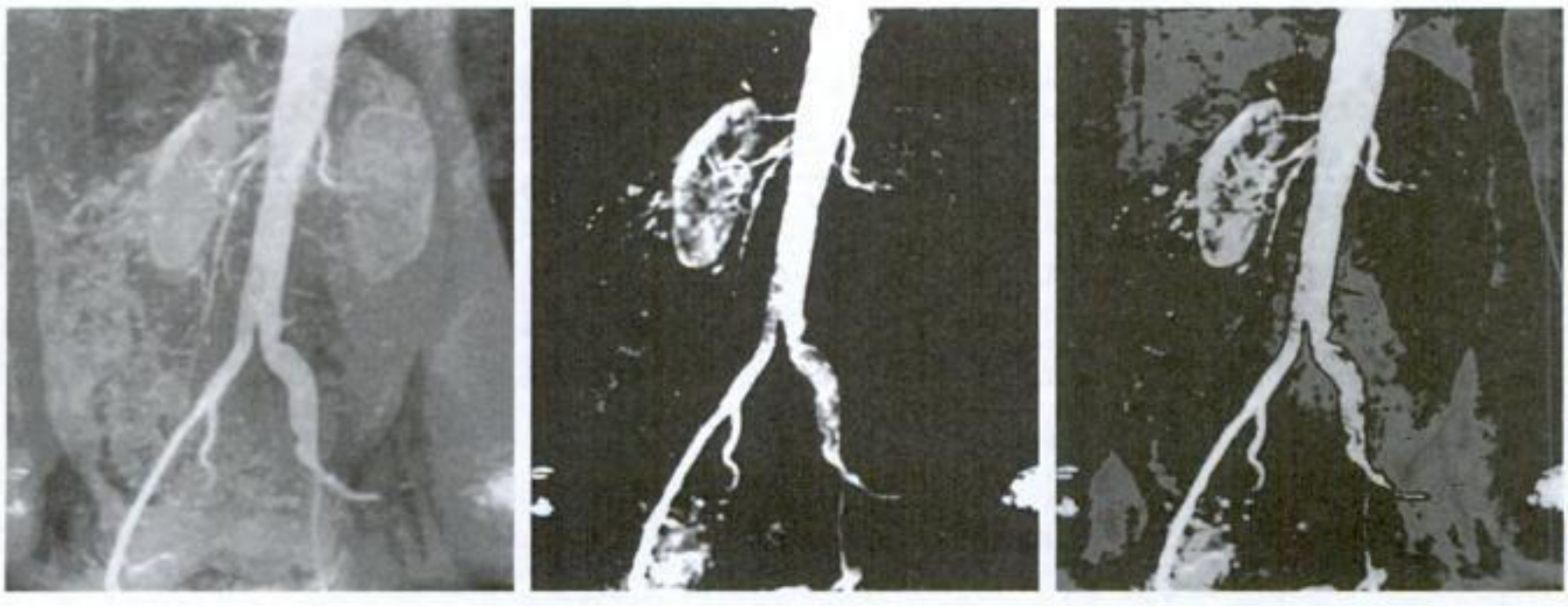
EXAMPLE 3.3: Intensity-level slicing.

■ Figure 3.12(a) is an aortic angiogram near the kidney area (see Section 1.3.2 for a more detailed explanation of this image). The objective of this example is to use intensity-level slicing to highlight the major blood vessels that appear brighter as a result of an injected contrast medium. Figure 3.12(b) shows the result of using a transformation of the form in Fig. 3.11(a), with the selected band near the top of the scale, because the range of interest is brighter than the background. The net result of this transformation is that the blood vessel and parts of the kidneys appear white, while all other intensities are black. This type of enhancement produces a binary image and is useful for studying the *shape* of the flow of the contrast medium (to detect blockages, for example).

If, on the other hand, interest lies in the actual intensity values of the region of interest, we can use the transformation in Fig. 3.11(b). Figure 3.12(c) shows the result of using such a transformation in which a band of intensities in the mid-gray region around the mean intensity was set to black, while all other intensities were left unchanged. Here, we see that the gray-level tonality of the major blood vessels and part of the kidney area were left intact. Such a result might be useful when interest lies in measuring the actual flow of the contrast medium as a function of time in a series of images. ■

Bit-plane slicing

Pixels are digital numbers composed of bits. For example, the intensity of each pixel in a 256-level gray-scale image is composed of 8 bits (i.e., one byte). Instead of highlighting intensity-level ranges, we could highlight the contribution



a b c

FIGURE 3.12 (a) Aortic angiogram. (b) Result of using a slicing transformation of the type illustrated in Fig. 3.11(a), with the range of intensities of interest selected in the upper end of the gray scale. (c) Result of using the transformation in Fig. 3.11(b), with the selected area set to black, so that grays in the area of the blood vessels and kidneys were preserved. (Original image courtesy of Dr. Thomas R. Gest, University of Michigan Medical School.)

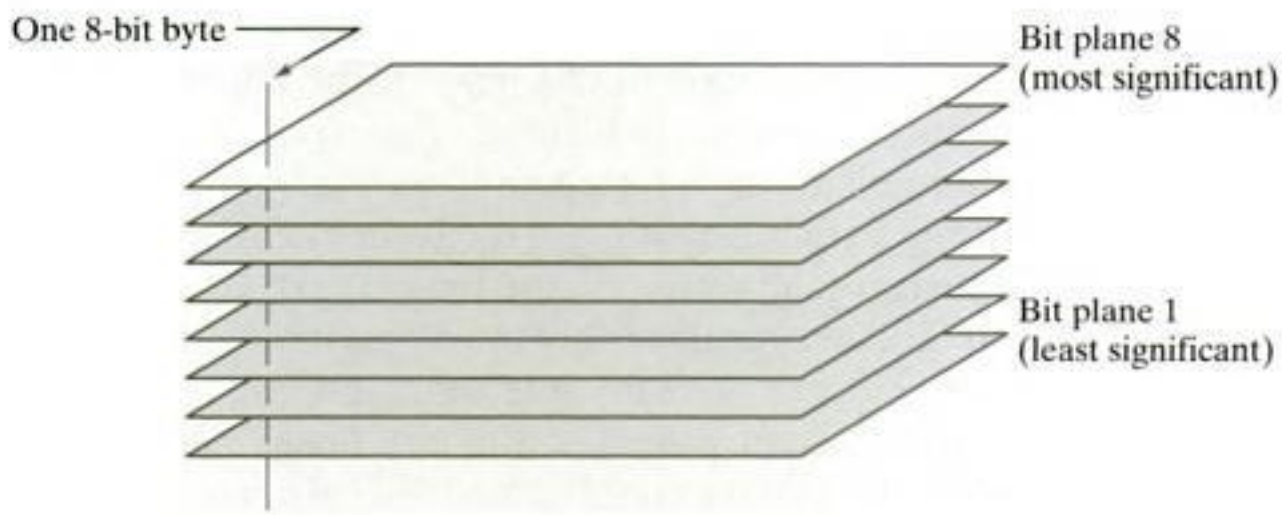


FIGURE 3.13 Bit-plane representation of an 8-bit image.

made to total image appearance by specific bits. As Fig. 3.13 illustrates, an 8-bit image may be considered as being composed of eight 1-bit planes, with plane 1 containing the lowest-order bit of all pixels in the image and plane 8 all the highest-order bits.

Figure 3.14(a) shows an 8-bit gray-scale image and Figs. 3.14(b) through (i) are its eight 1-bit planes, with Fig. 3.14(b) corresponding to the lowest-order bit. Observe that the four higher-order bit planes, especially the last two, contain a significant amount of the visually significant data. The lower-order planes contribute to more subtle intensity details in the image. The original image has a gray border whose intensity is 194. Notice that the corresponding borders of some of the bit planes are black (0), while others are white (1). To see why, consider a

a b c
d e f
g h i

FIGURE 3.14 (a) An 8-bit gray-scale image of size 500 × 1192 pixels. (b) through (i) Bit planes 1 through 8, with bit plane 1 corresponding to the least significant bit. Each bit plane is a binary image.

pixel in, say, the middle of the lower border of Fig. 3.14(a). The corresponding pixels in the bit planes, starting with the highest-order plane, have values 1 1 0 0 0 0 1 0, which is the binary representation of decimal 194. The value of any pixel in the original image can be similarly reconstructed from its corresponding binary-valued pixels in the bit planes.

In terms of intensity transformation functions, it is not difficult to show that the binary image for the 8th bit plane of an 8-bit image can be obtained by processing the input image with a thresholding intensity transformation function that maps all intensities between 0 and 127 to 0 and maps all levels between 128 and 255 to 1. The binary image in Fig. 3.14(i) was obtained in just this manner. It is left as an exercise (Problem 3.4) to obtain the intensity transformation functions for generating the other bit planes.

Decomposing an image into its bit planes is useful for analyzing the relative importance of each bit in the image, a process that aids in determining the adequacy of the number of bits used to quantize the image. Also, this type of decomposition is useful for image compression (the topic of Chapter 8), in which fewer than all planes are used in reconstructing an image. For example, Fig. 3.15(a) shows an image reconstructed using bit planes 8 and 7. The reconstruction is done by multiplying the pixels of the nth plane by the constant 2^{n-1}. This is nothing more than converting the nth significant binary bit to decimal. Each plane used is multiplied by the corresponding constant, and all planes used are added to obtain the gray scale image. Thus, to obtain Fig. 3.15(a), we multiplied bit plane 8 by 128, bit plane 7 by 64, and added the two planes. Although the main features of the original image were restored, the reconstructed image appears flat, especially in the background. This is not surprising because two planes can produce only four distinct intensity levels. Adding plane 6 to the reconstruction helped the situation, as Fig. 3.15(b) shows. Note that the background of this image has perceptible false contouring. This effect is reduced significantly by adding the 5th plane to the reconstruction, as Fig. 3.15(c) illustrates. Using more planes in the reconstruction would not contribute significantly to the appearance of this image. Thus, we conclude that storing the four highest-order bit planes would allow us to reconstruct the original image in acceptable detail. Storing these four planes instead of the original image requires 50% less storage (ignoring memory architecture issues).

a b c

FIGURE 3.15 Images reconstructed using (a) bit planes 8 and 7; (b) bit planes 8, 7, and 6; and (c) bit planes 8, 7, 6, and 5. Compare (c) with Fig. 3.14(a).

3.3 Histogram Processing

The *histogram* of a digital image with intensity levels in the range $[0, L-1]$ is a discrete function $h(r_k) = n_k$, where r_k is the kth intensity value and n_k is the number of pixels in the image with intensity r_k. It is common practice to normalize a histogram by dividing each of its components by the total number of pixels in the image, denoted by the product MN, where, as usual, M and N are the row and column dimensions of the image. Thus, a normalized histogram is given by $p(r_k) = n_k/MN$, for $k = 0, 1, 2, \ldots, L-1$. Loosely speaking, $p(r_k)$ is an estimate of the probability of occurrence of intensity level r_k in an image. The sum of all components of a normalized histogram is equal to 1.

Consult the book Web site for a review of basic probability theory.

Histograms are the basis for numerous spatial domain processing techniques. Histogram manipulation can be used for image enhancement, as shown in this section. In addition to providing useful image statistics, we shall see in subsequent chapters that the information inherent in histograms also is quite useful in other image processing applications, such as image compression and segmentation. Histograms are simple to calculate in software and also lend themselves to economic hardware implementations, thus making them a popular tool for real-time image processing.

As an introduction to histogram processing for intensity transformations, consider Fig. 3.16, which is the pollen image of Fig. 3.10 shown in four basic intensity characteristics: dark, light, low contrast, and high contrast. The right side of the figure shows the histograms corresponding to these images. The horizontal axis of each histogram plot corresponds to intensity values, r_k. The vertical axis corresponds to values of $h(r_k) = n_k$ or $p(r_k) = n_k/MN$ if the values are normalized. Thus, histograms may be viewed graphically simply as plots of $h(r_k) = n_k$ versus r_k or $p(r_k) = n_k/MN$ versus r_k.

We note in the dark image that the components of the histogram are concentrated on the low (dark) side of the intensity scale. Similarly, the components of the histogram of the light image are biased toward the high side of the scale. An image with low contrast has a narrow histogram located typically toward the middle of the intensity scale. For a monochrome image this implies a dull, washed-out gray look. Finally, we see that the components of the histogram in the high-contrast image cover a wide range of the intensity scale and, further, that the distribution of pixels is not too far from uniform, with very few vertical lines being much higher than the others. Intuitively, it is reasonable to conclude that an image whose pixels tend to occupy the entire range of possible intensity levels and, in addition, tend to be distributed uniformly, will have an appearance of high contrast and will exhibit a large variety of gray tones. The net effect will be an image that shows a great deal of gray-level detail and has high dynamic range. It will be shown shortly that it is possible to develop a transformation function that can automatically achieve this effect, based only on information available in the histogram of the input image.

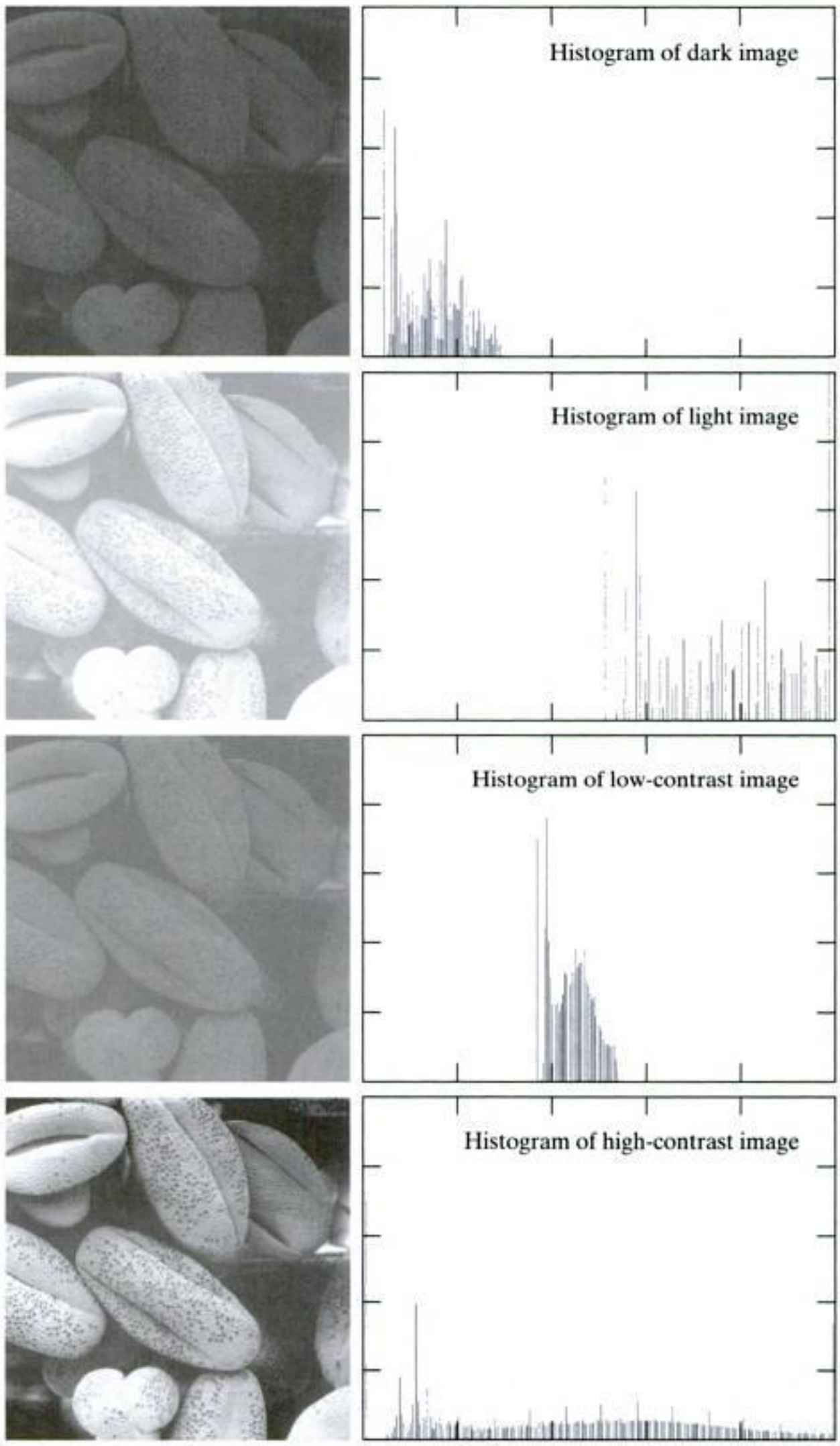


FIGURE 3.16 Four basic image types: dark, light, low contrast, high contrast, and their corresponding histograms.

3.3.1 Histogram Equalization

Consider for a moment continuous intensity values and let the variable r denote the intensities of an image to be processed. As usual, we assume that r is in the range $[0, L-1]$, with $r = 0$ representing black and $r = L-1$ representing white. For r satisfying these conditions, we focus attention on transformations (intensity mappings) of the form

$$s = T(r) \quad 0 \le r \le L-1 \tag{3.3-1}$$

that produce an output intensity level s for every pixel in the input image having intensity r. We assume that:

(a) $T(r)$ is a monotonically[†] increasing function in the interval $0 \le r \le L-1$; and

(b) $0 \le T(r) \le L-1$ for $0 \le r \le L-1$.

In some formulations to be discussed later, we use the inverse

$$r = T^{-1}(s) \quad 0 \le s \le L-1 \tag{3.3-2}$$

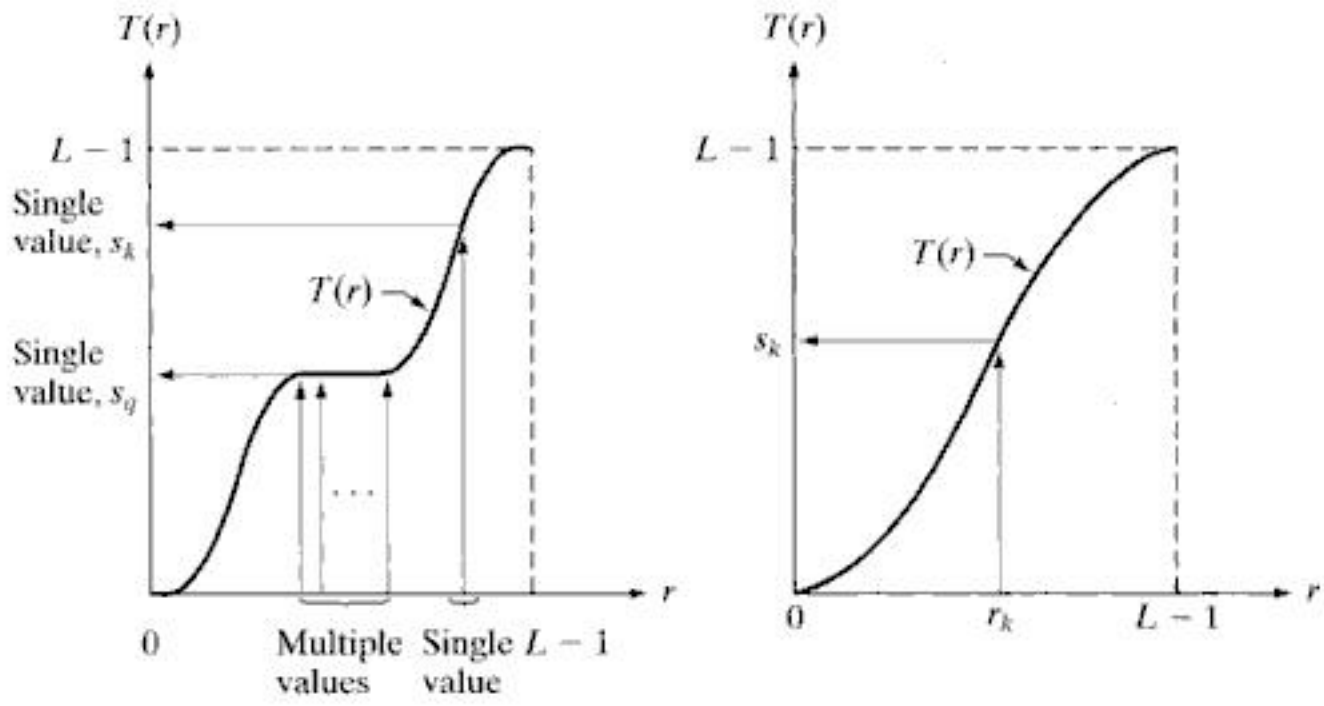
in which case we change condition (a) to

(a') $T(r)$ is a strictly monotonically increasing function in the interval $0 \le r \le L-1$.

The requirement in condition (a) that $T(r)$ be monotonically increasing guarantees that output intensity values will never be less than corresponding input values, thus preventing artifacts created by reversals of intensity. Condition (b) guarantees that the range of output intensities is the same as the input. Finally, condition (a') guarantees that the mappings from s back to r will be one-to-one, thus preventing ambiguities. Figure 3.17(a) shows a function

a b

FIGURE 3.17
(a) Monotonically increasing function, showing how multiple values can map to a single value. (b) Strictly monotonically increasing function. This is a one-to-one mapping, both ways.



[†]Recall that a function $T(r)$ is *monotonically increasing* if $T(r_2) \ge T(r_1)$ for $r_2 > r_1$. $T(r)$ is a *strictly monotonically increasing* function if $T(r_2) > T(r_1)$ for $r_2 > r_1$. Similar definitions apply to monotonically decreasing functions.

that satisfies conditions (a) and (b). Here, we see that it is possible for multiple values to map to a single value and still satisfy these two conditions. That is, a monotonic transformation function performs a one-to-one or many-to-one mapping. This is perfectly fine when mapping from r to s. However, Fig. 3.17(a) presents a problem if we wanted to recover the values of r uniquely from the mapped values (inverse mapping can be visualized by reversing the direction of the arrows). This would be possible for the inverse mapping of s_k in Fig. 3.17(a), but the inverse mapping of s_q is a *range* of values, which, of course, prevents us in general from recovering the original value of r that resulted in s_q. As Fig. 3.17(b) shows, requiring that $T(r)$ be strictly monotonic guarantees that the inverse mappings will be *single valued* (i.e., the mapping is one-to-one in both directions). This is a theoretical requirement that allows us to derive some important histogram processing techniques later in this chapter. Because in practice we deal with integer intensity values, we are forced to round all results to their nearest integer values. Therefore, when strict monotonicity is not satisfied, we address the problem of a nonunique inverse transformation by looking for the closest integer matches. Example 3.8 gives an illustration of this.

The intensity levels in an image may be viewed as random variables in the interval $[0, L-1]$. A fundamental descriptor of a random variable is its probability density function (PDF). Let $p_r(r)$ and $p_s(s)$ denote the PDFs of r and s, respectively, where the subscripts on p are used to indicate that p_r and p_s are different functions in general. A fundamental result from basic probability theory is that if $p_r(r)$ and $T(r)$ are known, and $T(r)$ is continuous and differentiable over the range of values of interest, then the PDF of the transformed (mapped) variable s can be obtained using the simple formula

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right| \tag{3.3-3}$$

Thus, we see that the PDF of the output intensity variable, s, is determined by the PDF of the input intensities and the transformation function used [recall that r and s are related by $T(r)$].

A transformation function of particular importance in image processing has the form

$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw \tag{3.3-4}$$

where w is a dummy variable of integration. The right side of this equation is recognized as the cumulative distribution function (CDF) of random variable r. Because PDFs always are positive, and recalling that the integral of a function is the area under the function, it follows that the transformation function of Eq. (3.3-4) satisfies condition (a) because the area under the function cannot decrease as r increases. When the upper limit in this equation is $r = (L-1)$, the integral evaluates to 1 (the area under a PDF curve always is 1), so the maximum value of s is $(L-1)$ and condition (b) is satisfied also.

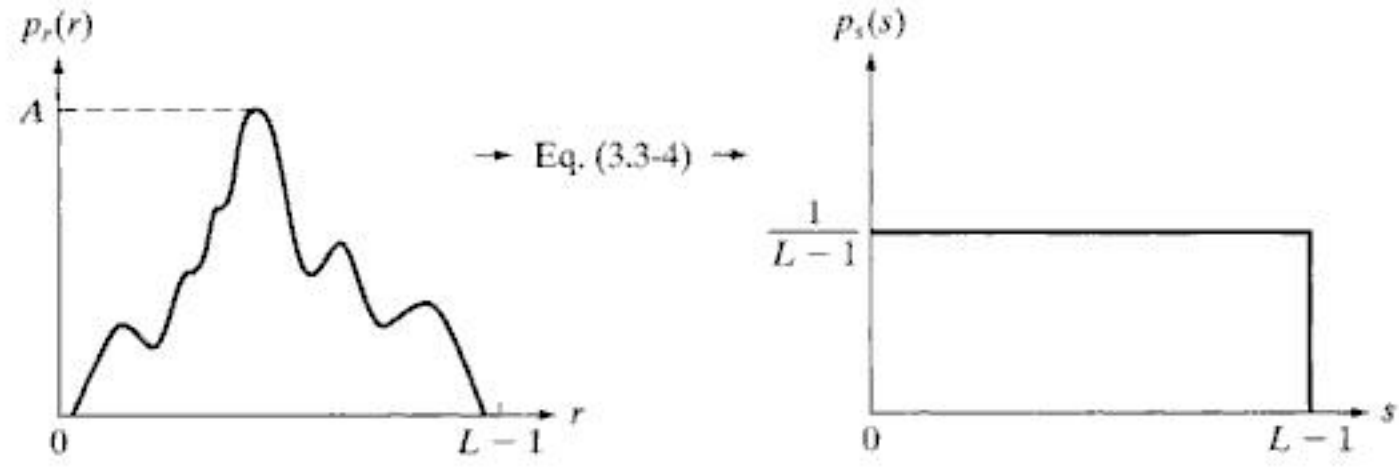
To find the $p_s(s)$ corresponding to the transformation just discussed, we use Eq. (3.3-3). We know from Leibniz's rule in basic calculus that the derivative of a definite integral with respect to its upper limit is the integrand evaluated at the limit. That is,

$$\begin{aligned}\frac{ds}{dr} &= \frac{dT(r)}{dr} \\ &= (L-1)\frac{d}{dr}\left[\int_0^r p_r(w)\,dw\right] \\ &= (L-1)p_r(r)\end{aligned} \tag{3.3-5}$$

Substituting this result for dr/ds in Eq. (3.3-3), and keeping in mind that all probability values are positive, yields

$$\begin{aligned}p_s(s) &= p_r(r)\left|\frac{dr}{ds}\right| \\ &= p_r(r)\left|\frac{1}{(L-1)p_r(r)}\right| \\ &= \frac{1}{L-1} \qquad 0 \le s \le L-1\end{aligned} \tag{3.3-6}$$

We recognize the form of $p_s(s)$ in the last line of this equation as a *uniform* probability density function. Simply stated, we have demonstrated that performing the intensity transformation in Eq. (3.3-4) yields a random variable, s, characterized by a uniform PDF. It is important to note from this equation that $T(r)$ depends on $p_r(r)$ but, as Eq. (3.3-6) shows, the resulting $p_s(s)$ *always* is uniform, *independently* of the form of $p_r(r)$. Figure 3.18 illustrates these concepts.



a b

FIGURE 3.18 (a) An arbitrary PDF. (b) Result of applying the transformation in Eq. (3.3-4) to all intensity levels, r. The resulting intensities, s, have a uniform PDF, independently of the form of the PDF of the r's.

EXAMPLE 3.4: Illustration of Eqs. (3.3-4) and (3.3-6).

■ To fix ideas, consider the following simple example. Suppose that the (continuous) intensity values in an image have the PDF

$$p_r(r) = \begin{cases} \dfrac{2r}{(L-1)^2} & \text{for } 0 \le r \le L-1 \\ 0 & \text{otherwise} \end{cases}$$

From Eq. (3.3-4),

$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw = \frac{2}{L-1}\int_0^r w\,dw = \frac{r^2}{L-1}$$

Suppose next that we form a new image with intensities, s, obtained using this transformation; that is, the s values are formed by squaring the corresponding intensity values of the input image and dividing them by $(L-1)$. For example, consider an image in which $L = 10$, and suppose that a pixel in an arbitrary location (x, y) in the input image has intensity $r = 3$. Then the pixel in that location in the new image is $s = T(r) = r^2/9 = 1$. We can verify that the PDF of the intensities in the new image is uniform simply by substituting $p_r(r)$ into Eq. (3.3-6) and using the fact that $s = r^2/(L-1)$; that is,

$$p_s(s) = p_r(r)\left|\frac{dr}{ds}\right| = \frac{2r}{(L-1)^2}\left|\left[\frac{ds}{dr}\right]^{-1}\right|$$

$$= \frac{2r}{(L-1)^2}\left|\left[\frac{d}{dr}\frac{r^2}{L-1}\right]^{-1}\right|$$

$$= \frac{2r}{(L-1)^2}\left|\frac{(L-1)}{2r}\right| = \frac{1}{L-1}$$

where the last step follows from the fact that r is nonnegative and we assume that $L > 1$. As expected, the result is a uniform PDF. ■

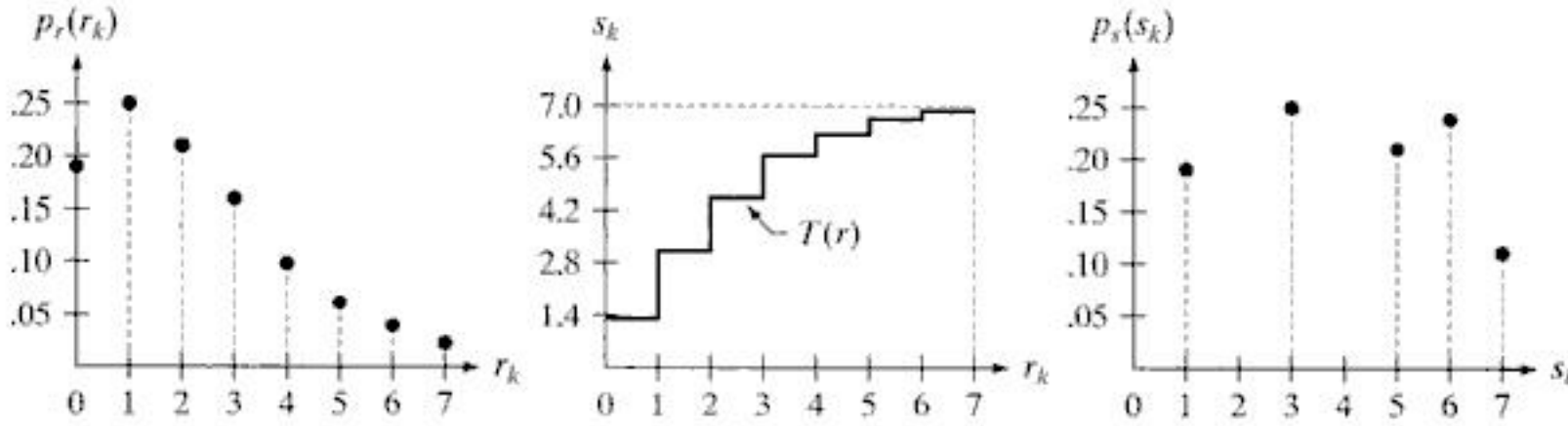
For discrete values, we deal with probabilities (histogram values) and summations instead of probability density functions and integrals.[†] As mentioned earlier, the probability of occurrence of intensity level r_k in a digital image is approximated by

$$p_r(r_k) = \frac{n_k}{MN} \qquad k = 0, 1, 2, \ldots, L-1 \tag{3.3-7}$$

where MN is the total number of pixels in the image, n_k is the number of pixels that have intensity r_k, and L is the number of possible intensity levels in the image (e.g., 256 for an 8-bit image). As noted in the beginning of this section, a plot of $p_r(r_k)$ versus r_k is commonly referred to as a *histogram*.

[†]The conditions of monotonicity stated earlier apply also in the discrete case. We simply restrict the values of the variables to be discrete.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a b c

FIGURE 3.19 Illustration of histogram equalization of a 3-bit (8 intensity levels) image. (a) Original histogram. (b) Transformation function. (c) Equalized histogram.

At this point, the s values still have fractions because they were generated by summing probability values, so we round them to the nearest integer:

$$s_0 = 1.33 \rightarrow 1 \qquad s_4 = 6.23 \rightarrow 6$$
$$s_1 = 3.08 \rightarrow 3 \qquad s_5 = 6.65 \rightarrow 7$$
$$s_2 = 4.55 \rightarrow 5 \qquad s_6 = 6.86 \rightarrow 7$$
$$s_3 = 5.67 \rightarrow 6 \qquad s_7 = 7.00 \rightarrow 7$$

These are the values of the equalized histogram. Observe that there are only five distinct intensity levels. Because $r_0 = 0$ was mapped to $s_0 = 1$, there are 790 pixels in the histogram equalized image with this value (see Table 3.1). Also, there are in this image 1023 pixels with a value of $s_1 = 3$ and 850 pixels with a value of $s_2 = 5$. However both r_3 and r_4 were mapped to the same value, 6, so there are $(656 + 329) = 985$ pixels in the equalized image with this value. Similarly, there are $(245 + 122 + 81) = 448$ pixels with a value of 7 in the histogram equalized image. Dividing these numbers by $MN = 4096$ yielded the equalized histogram in Fig. 3.19(c).

Because a histogram is an approximation to a PDF, and no new allowed intensity levels are created in the process, perfectly flat histograms are rare in practical applications of histogram equalization. Thus, unlike its continuous counterpart, it cannot be proved (in general) that discrete histogram equalization results in a uniform histogram. However, as you will see shortly, using Eq. (3.3-8) has the general tendency to spread the histogram of the input image so that the intensity levels of the equalized image span a wider range of the intensity scale. The net result is contrast enhancement. ■

We discussed earlier in this section the many advantages of having intensity values that cover the entire gray scale. In addition to producing intensities that have this tendency, the method just derived has the additional advantage that it is fully "automatic." In other words, given an image, the process of histogram equalization consists simply of implementing Eq. (3.3-8), which is based on information that can be extracted directly from the given image, without the

need for further parameter specifications. We note also the simplicity of the computations required to implement the technique.

The *inverse transformation* from s back to r is denoted by

$$r_k = T^{-1}(s_k) \qquad k = 0, 1, 2, \ldots, L-1 \tag{3.3-9}$$

It can be shown (Problem 3.10) that this inverse transformation satisfies conditions (a′) and (b) only if none of the levels, r_k, $k = 0, 1, 2, \ldots, L-1$, are missing from the input image, which in turn means that none of the components of the image histogram are zero. Although the inverse transformation is not used in histogram equalization, it plays a central role in the histogram-matching scheme developed in the next section.

EXAMPLE 3.6: Histogram equalization.

■ The left column in Fig. 3.20 shows the four images from Fig. 3.16, and the center column shows the result of performing histogram equalization on each of these images. The first three results from top to bottom show significant improvement. As expected, histogram equalization did not have much effect on the fourth image because the intensities of this image already span the full intensity scale. Figure 3.21 shows the transformation functions used to generate the equalized images in Fig. 3.20. These functions were generated using Eq. (3.3-8). Observe that transformation (4) has a nearly linear shape, indicating that the inputs were mapped to nearly equal outputs.

The third column in Fig. 3.20 shows the histograms of the equalized images. It is of interest to note that, while all these histograms are different, the histogram-equalized images themselves are visually very similar. This is not unexpected because the basic difference between the images on the left column is one of contrast, not content. In other words, because the images have the same content, the increase in contrast resulting from histogram equalization was enough to render any intensity differences in the equalized images visually indistinguishable. Given the significant contrast differences between the original images, this example illustrates the power of histogram equalization as an adaptive contrast enhancement tool. ■

3.3.2 Histogram Matching (Specification)

As indicated in the preceding discussion, histogram equalization automatically determines a transformation function that seeks to produce an output image that has a uniform histogram. When automatic enhancement is desired, this is a good approach because the results from this technique are predictable and the method is simple to implement. We show in this section that there are applications in which attempting to base enhancement on a uniform histogram is not the best approach. In particular, it is useful sometimes to be able to specify the shape of the histogram that we wish the processed image to have. The method used to generate a processed image that has a specified histogram is called *histogram matching* or *histogram specification*.

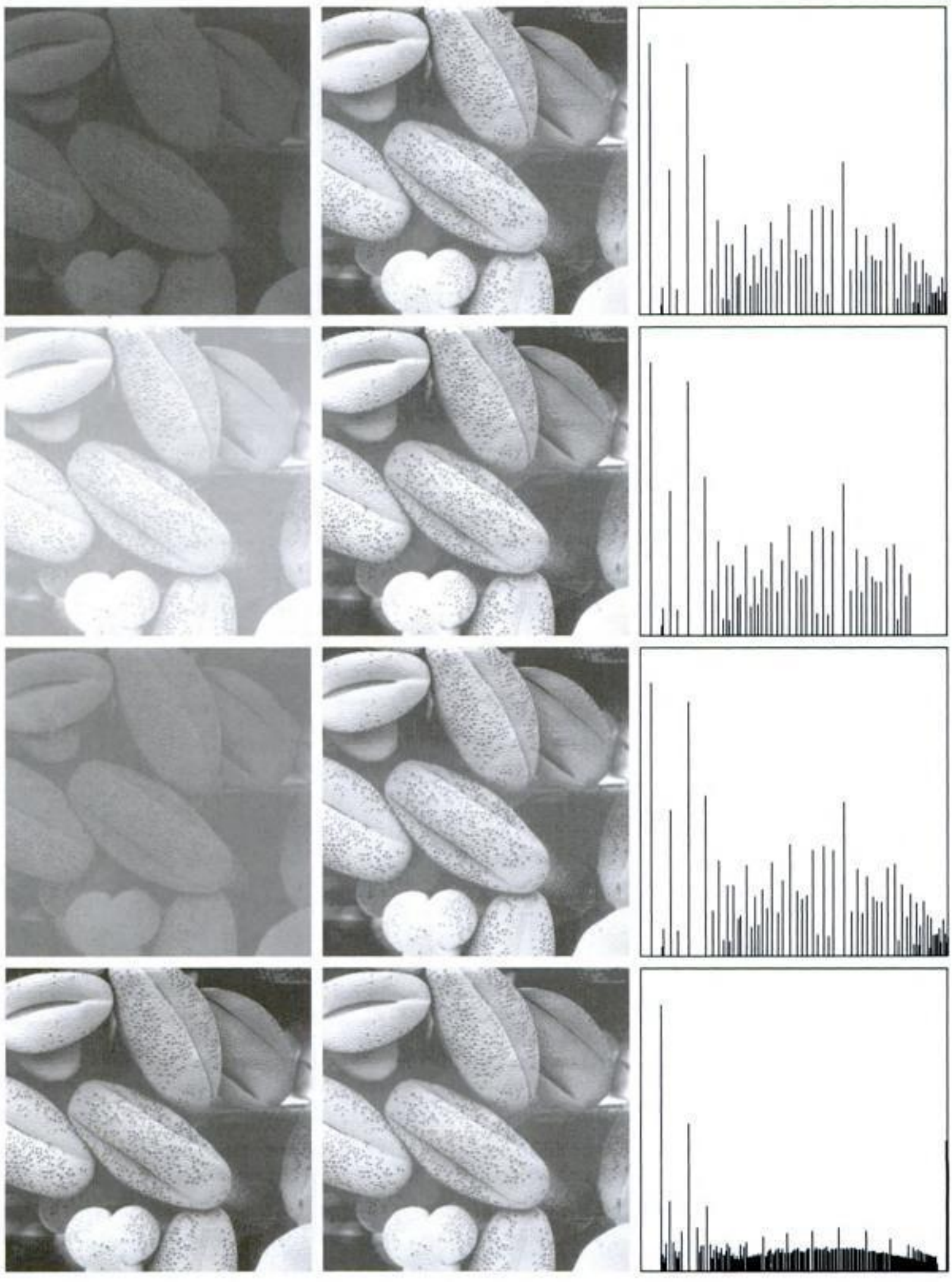


FIGURE 3.20 Left column: images from Fig. 3.16. Center column: corresponding histogram-equalized images. Right column: histograms of the images in the center column.

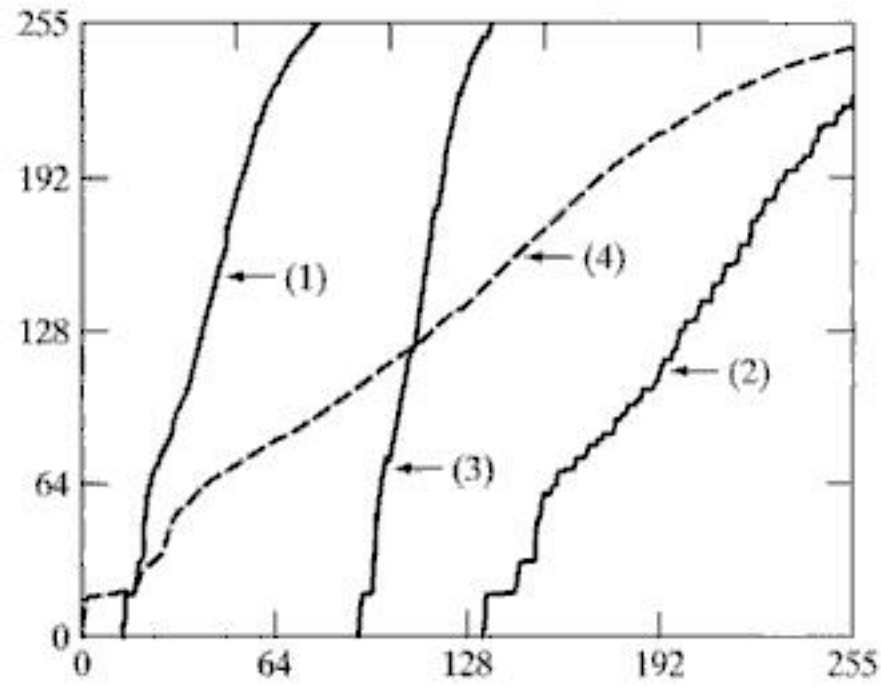


FIGURE 3.21 Transformation functions for histogram equalization. Transformations (1) through (4) were obtained from the histograms of the images (from top to bottom) in the left column of Fig. 3.20 using Eq. (3.3-8).

Let us return for a moment to continuous intensities r and z (considered continuous random variables), and let $p_r(r)$ and $p_z(z)$ denote their corresponding continuous probability density functions. In this notation, r and z denote the intensity levels of the input and output (processed) images, respectively. We can estimate $p_r(r)$ from the given input image, while $p_z(z)$ is the *specified* probability density function that we wish the output image to have.

Let s be a random variable with the property

$$s = T(r) = (L - 1)\int_0^r p_r(w)\,dw \tag{3.3-10}$$

where, as before, w is a dummy variable of integration. We recognize this expression as the continuous version of histogram equalization given in Eq. (3.3-4).

Suppose next that we define a random variable z with the property

$$G(z) = (L - 1)\int_0^z p_z(t)\,dt = s \tag{3.3-11}$$

where t is a dummy variable of integration. It then follows from these two equations that $G(z) = T(r)$ and, therefore, that z must satisfy the condition

$$z = G^{-1}[T(r)] = G^{-1}(s) \tag{3.3-12}$$

The transformation $T(r)$ can be obtained from Eq. (3.3-10) once $p_r(r)$ has been estimated from the input image. Similarly, the transformation function $G(z)$ can be obtained using Eq. (3.3-11) because $p_z(z)$ is given.

Equations (3.3-10) through (3.3-12) show that an image whose intensity levels have a specified probability density function can be obtained from a given image by using the following procedure:

1. Obtain $p_r(r)$ from the input image and use Eq. (3.3-10) to obtain the values of s.
2. Use the specified PDF in Eq. (3.3-11) to obtain the transformation function $G(z)$.

3. Obtain the inverse transformation $z = G^{-1}(s)$; because z is obtained from s, this process is a *mapping* from s to z, the latter being the desired values.
4. Obtain the output image by first equalizing the input image using Eq. (3.3-10); the pixel values in this image are the s values. For each pixel with value s in the equalized image, perform the inverse mapping $z = G^{-1}(s)$ to obtain the corresponding pixel in the output image. When all pixels have been thus processed, the PDF of the output image will be equal to the specified PDF.

EXAMPLE 3.7: Histogram specification.

■ Assuming continuous intensity values, suppose that an image has the intensity PDF $p_r(r) = 2r/(L-1)^2$ for $0 \le r \le (L-1)$ and $p_r(r) = 0$ for other values of r. Find the transformation function that will produce an image whose intensity PDF is $p_z(z) = 3z^2/(L-1)^3$ for $0 \le z \le (L-1)$ and $p_z(z) = 0$ for other values of z.

First, we find the histogram equalization transformation for the interval $[0, L-1]$:

$$s = T(r) = (L-1)\int_0^r p_r(w)\,dw = \frac{2}{(L-1)}\int_0^r w\,dw = \frac{r^2}{(L-1)}$$

By definition, this transformation is 0 for values outside the range $[0, L-1]$. Squaring the values of the input intensities and dividing them by $(L-1)^2$ will produce an image whose intensities, s, have a uniform PDF because this is a histogram-equalization transformation, as discussed earlier.

We are interested in an image with a specified histogram, so we find next

$$G(z) = (L-1)\int_0^z p_z(w)\,dw = \frac{3}{(L-1)^2}\int_0^z w^2\,dw = \frac{z^3}{(L-1)^2}$$

over the interval $[0, L-1]$; this function is 0 elsewhere by definition. Finally, we require that $G(z) = s$, but $G(z) = z^3/(L-1)^2$; so $z^3/(L-1)^2 = s$, and we have

$$z = \left[(L-1)^2 s\right]^{1/3}$$

So, if we multiply every histogram equalized pixel by $(L-1)^2$ and raise the product to the power 1/3, the result will be an image whose intensities, z, have the PDF $p_z(z) = 3z^2/(L-1)^3$ in the interval $[0, L-1]$, as desired.

Because $s = r^2/(L-1)$ we can generate the z's directly from the intensities, r, of the input image:

$$z = \left[(L-1)^2 s\right]^{1/3} = \left[(L-1)^2 \frac{r^2}{(L-1)}\right]^{1/3} = \left[(L-1)r^2\right]^{1/3}$$

Thus, squaring the value of each pixel in the original image, multiplying the result by $(L-1)$, and raising the product to the power 1/3 will yield an image

whose intensity levels, z, have the specified PDF. We see that the intermediate step of equalizing the input image can be skipped; all we need is to obtain the transformation function $T(r)$ that maps r to s. Then, the two steps can be combined into a single transformation from r to z. ■

As the preceding example shows, histogram specification is straightforward in principle. In practice, a common difficulty is finding meaningful analytical expressions for $T(r)$ and G^{-1}. Fortunately, the problem is simplified significantly when dealing with discrete quantities. The price paid is the same as for histogram equalization, where only an approximation to the desired histogram is achievable. In spite of this, however, some very useful results can be obtained, even with crude approximations.

The discrete formulation of Eq. (3.3-10) is the histogram equalization transformation in Eq. (3.3-8), which we repeat here for convenience:

$$\begin{aligned} s_k &= T(r_k) = (L-1)\sum_{j=0}^{k} p_r(r_j) \\ &= \frac{(L-1)}{MN}\sum_{j=0}^{k} n_j \qquad k = 0, 1, 2, \ldots, L-1 \end{aligned} \tag{3.3-13}$$

where, as before, MN is the total number of pixels in the image, n_j is the number of pixels that have intensity value r_j, and L is the total number of possible intensity levels in the image. Similarly, given a specific value of s_k, the discrete formulation of Eq. (3.3-11) involves computing the transformation function

$$G(z_q) = (L-1)\sum_{i=0}^{q} p_z(z_i) \tag{3.3-14}$$

for a value of q, so that

$$G(z_q) = s_k \tag{3.3-15}$$

where $p_z(z_i)$, is the ith value of the specified histogram. As before, we find the desired value z_q by obtaining the inverse transformation:

$$z_q = G^{-1}(s_k) \tag{3.3-16}$$

In other words, this operation gives a value of z for each value of s; thus, it performs a *mapping* from s to z.

In practice, we do not need to compute the inverse of G. Because we deal with intensity levels that are integers (e.g., 0 to 255 for an 8-bit image), it is a simple matter to compute all the possible values of G using Eq. (3.3-14) for $q = 0, 1, 2, \ldots, L-1$. These values are scaled and rounded to their nearest integer values spanning the range $[0, L-1]$. The values are stored in a table. Then, given a particular value of s_k, we look for the closest match in the values stored in the table. If, for example, the 64th entry in the table is the closest to s_k, then $q = 63$ (recall that we start counting at 0) and z_{63} is the best solution to Eq. (3.3-15). Thus, the given value s_k would be associated with z_{63} (i.e., that

specific value of s_k would *map* to z_{63}). Because the zs are intensities used as the basis for specifying the histogram $p_z(z)$, it follows that $z_0 = 0$, $z_1 = 1, \ldots, z_{L-1} = L - 1$, so z_{63} would have the intensity value 63. By repeating this procedure, we would find the mapping of each value of s_k to the value of z_q that is the closest solution to Eq. (3.3-15). These mappings are the solution to the histogram-specification problem.

Recalling that the s_ks are the values of the histogram-equalized image, we may summarize the histogram-specification procedure as follows:

1. Compute the histogram $p_r(r)$ of the given image, and use it to find the histogram equalization transformation in Eq. (3.3-13). Round the resulting values, s_k, to the integer range $[0, L - 1]$.
2. Compute all values of the transformation function G using the Eq. (3.3-14) for $q = 0, 1, 2, \ldots, L - 1$, where $p_z(z_i)$ are the values of the specified histogram. Round the values of G to integers in the range $[0, L - 1]$. Store the values of G in a table.
3. For every value of s_k, $k = 0, 1, 2, \ldots, L - 1$, use the stored values of G from step 2 to find the corresponding value of z_q so that $G(z_q)$ is closest to s_k and store these mappings from s to z. When more than one value of z_q satisfies the given s_k (i.e., the mapping is not unique), choose the smallest value by convention.
4. Form the histogram-specified image by first histogram-equalizing the input image and then mapping every equalized pixel value, s_k, of this image to the corresponding value z_q in the histogram-specified image using the mappings found in step 3. As in the continuous case, the intermediate step of equalizing the input image is conceptual. It can be skipped by combining the two transformation functions, T and G^{-1}, as Example 3.8 shows.

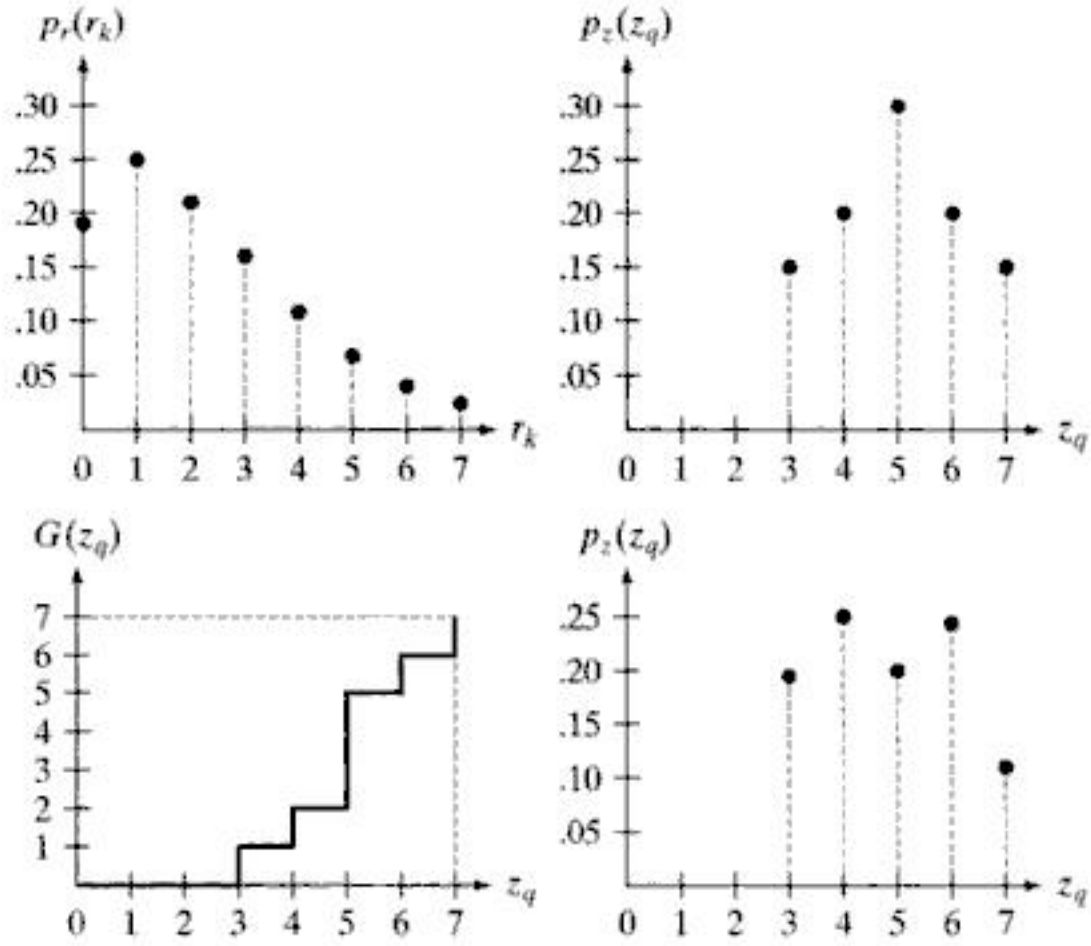
As mentioned earlier, for G^{-1} to satisfy conditions (a′) and (b), G has to be strictly monotonic, which, according to Eq. (3.3-14), means that none of the values $p_z(z_i)$ of the specified histogram can be zero (Problem 3.10). When working with discrete quantities, the fact that this condition may not be satisfied is not a serious implementation issue, as step 3 above indicates. The following example illustrates this numerically.

EXAMPLE 3.8: A simple example of histogram specification.

■ Consider again the 64×64 hypothetical image from Example 3.5, whose histogram is repeated in Fig. 3.22(a). It is desired to transform this histogram so that it will have the values specified in the second column of Table 3.2. Figure 3.22(b) shows a sketch of this histogram.

The first step in the procedure is to obtain the scaled histogram-equalized values, which we did in Example 3.5:

$$s_0 = 1 \quad s_2 = 5 \quad s_4 = 7 \quad s_6 = 7$$
$$s_1 = 3 \quad s_3 = 6 \quad s_5 = 7 \quad s_7 = 7$$



a b
c d

FIGURE 3.22
(a) Histogram of a 3-bit image. (b) Specified histogram. (c) Transformation function obtained from the specified histogram. (d) Result of performing histogram specification. Compare (b) and (d).

In the next step, we compute all the values of the transformation function, G, using Eq. (3.3-14):

$$G(z_0) = 7\sum_{j=0}^{0} p_z(z_j) = 0.00$$

Similarly,

$$G(z_1) = 7\sum_{j=0}^{1} p_z(z_j) = 7\left[p(z_0) + p(z_1)\right] = 0.00$$

and

$$G(z_2) = 0.00 \quad G(z_4) = 2.45 \quad G(z_6) = 5.95$$
$$G(z_3) = 1.05 \quad G(z_5) = 4.55 \quad G(z_7) = 7.00$$

TABLE 3.2
Specified and actual histograms (the values in the third column are from the computations performed in the body of Example 3.8).

z_q	**Specified** $p_z(z_q)$	**Actual** $p_z(z_k)$
$z_0 = 0$	0.00	0.00
$z_1 = 1$	0.00	0.00
$z_2 = 2$	0.00	0.00
$z_3 = 3$	0.15	0.19
$z_4 = 4$	0.20	0.25
$z_5 = 5$	0.30	0.21
$z_6 = 6$	0.20	0.24
$z_7 = 7$	0.15	0.11

As in Example 3.5, these fractional values are converted to integers in our valid range, [0, 7]. The results are:

$$G(z_0) = 0.00 \rightarrow 0 \qquad G(z_4) = 2.45 \rightarrow 2$$
$$G(z_1) = 0.00 \rightarrow 0 \qquad G(z_5) = 4.55 \rightarrow 5$$
$$G(z_2) = 0.00 \rightarrow 0 \qquad G(z_6) = 5.95 \rightarrow 6$$
$$G(z_3) = 1.05 \rightarrow 1 \qquad G(z_7) = 7.00 \rightarrow 7$$

These results are summarized in Table 3.3, and the transformation function is sketched in Fig. 3.22(c). Observe that G is not strictly monotonic, so condition (a′) is violated. Therefore, we make use of the approach outlined in step 3 of the algorithm to handle this situation.

In the third step of the procedure, we find the smallest value of z_q so that the value $G(z_q)$ is the closest to s_k. We do this for every value of s_k to create the required mappings from s to z. For example, $s_0 = 1$, and we see that $G(z_3) = 1$, which is a perfect match in this case, so we have the correspondence $s_0 \rightarrow z_3$. That is, every pixel whose value is 1 in the histogram equalized image would map to a pixel valued 3 (in the corresponding location) in the histogram-specified image. Continuing in this manner, we arrive at the mappings in Table 3.4.

In the final step of the procedure, we use the mappings in Table 3.4 to map every pixel in the histogram equalized image into a corresponding pixel in the newly created histogram-specified image. The values of the resulting histogram are listed in the third column of Table 3.2, and the histogram is sketched in Fig. 3.22(d). The values of $p_z(z_q)$ were obtained using the same procedure as in Example 3.5. For instance, we see in Table 3.4 that $s = 1$ maps to $z = 3$, and there are 790 pixels in the histogram-equalized image with a value of 1. Therefore, $p_z(z_3) = 790/4096 = 0.19$.

Although the final result shown in Fig. 3.22(d) does not match the specified histogram exactly, the general trend of moving the intensities toward the high end of the intensity scale definitely was achieved. As mentioned earlier, obtaining the histogram-equalized image as an intermediate step is useful for explaining the procedure, but this is not necessary. Instead, we could list the mappings from the rs to the ss and from the ss to the zs in a three-column

z_q	$G(z_q)$
$z_0 = 0$	0
$z_1 = 1$	0
$z_2 = 2$	0
$z_3 = 3$	1
$z_4 = 4$	2
$z_5 = 5$	5
$z_6 = 6$	6
$z_7 = 7$	7

TABLE 3.3
All possible values of the transformation function G scaled, rounded, and ordered with respect to z.

TABLE 3.4
Mappings of all the values of s_k into corresponding values of z_q.

s_k	→	z_q
1	→	3
3	→	4
5	→	5
6	→	6
7	→	7

table. Then, we would use those mappings to map the original pixels directly into the pixels of the histogram-specified image. ■

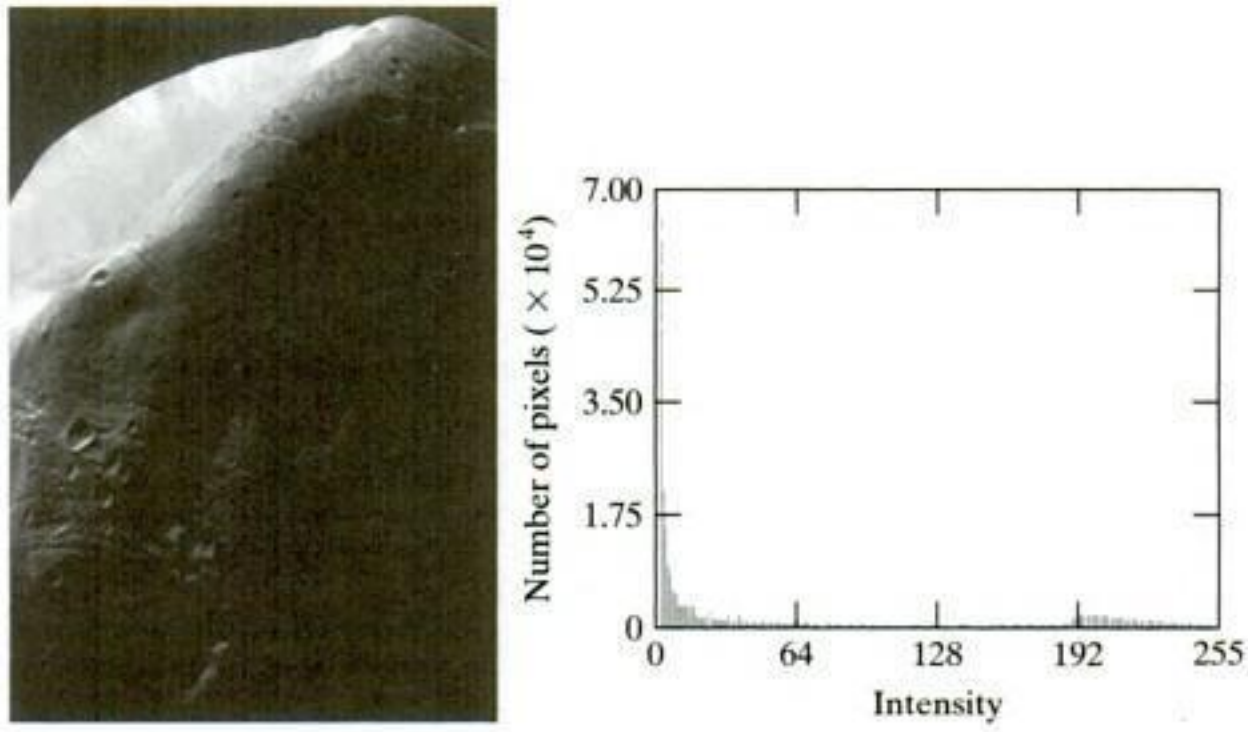
EXAMPLE 3.9: Comparison between histogram equalization and histogram matching.

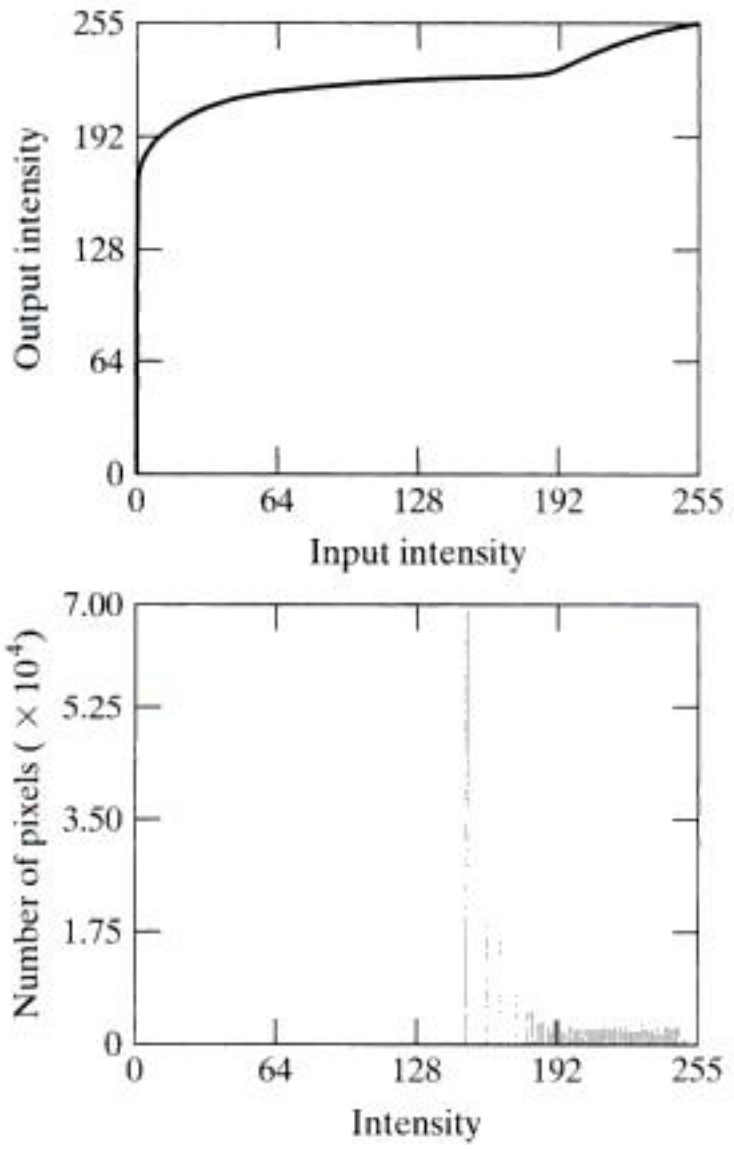
■ Figure 3.23(a) shows an image of the Mars moon, Phobos, taken by NASA's *Mars Global Surveyor*. Figure 3.23(b) shows the histogram of Fig. 3.23(a). The image is dominated by large, dark areas, resulting in a histogram characterized by a large concentration of pixels in the dark end of the gray scale. At first glance, one might conclude that histogram equalization would be a good approach to enhance this image, so that details in the dark areas become more visible. It is demonstrated in the following discussion that this is not so.

Figure 3.24(a) shows the histogram equalization transformation [Eq. (3.3-8) or (3.3-13)] obtained from the histogram in Fig. 3.23(b). The most relevant characteristic of this transformation function is how fast it rises from intensity level 0 to a level near 190. This is caused by the large concentration of pixels in the input histogram having levels near 0. When this transformation is applied to the levels of the input image to obtain a histogram-equalized result, the net effect is to map a very narrow interval of dark pixels into the upper end of the gray scale of the output image. Because numerous pixels in the input image have levels precisely in this interval, we would expect the result to be an image with a light, washed-out appearance. As Fig. 3.24(b) shows, this is indeed the

a b

FIGURE 3.23
(a) Image of the Mars moon Phobos taken by NASA's *Mars Global Surveyor*. (b) Histogram. (Original image courtesy of NASA.)





a b
c

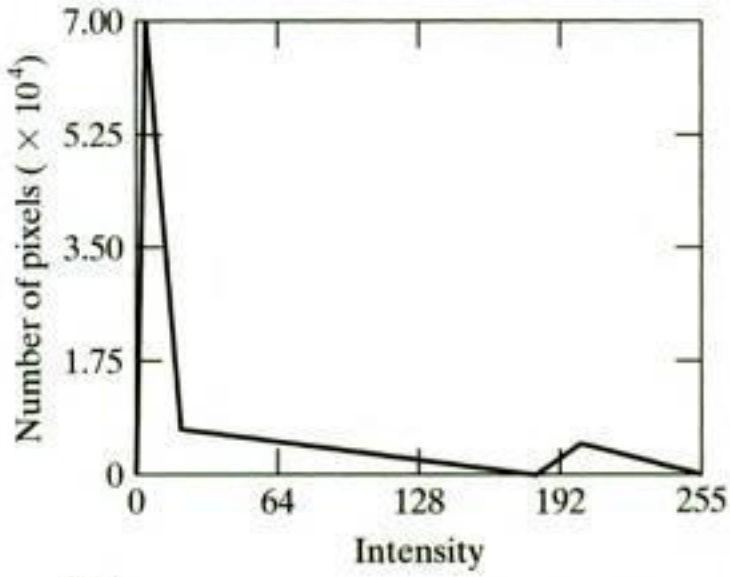
FIGURE 3.24
(a) Transformation function for histogram equalization. (b) Histogram-equalized image (note the washed-out appearance). (c) Histogram of (b).

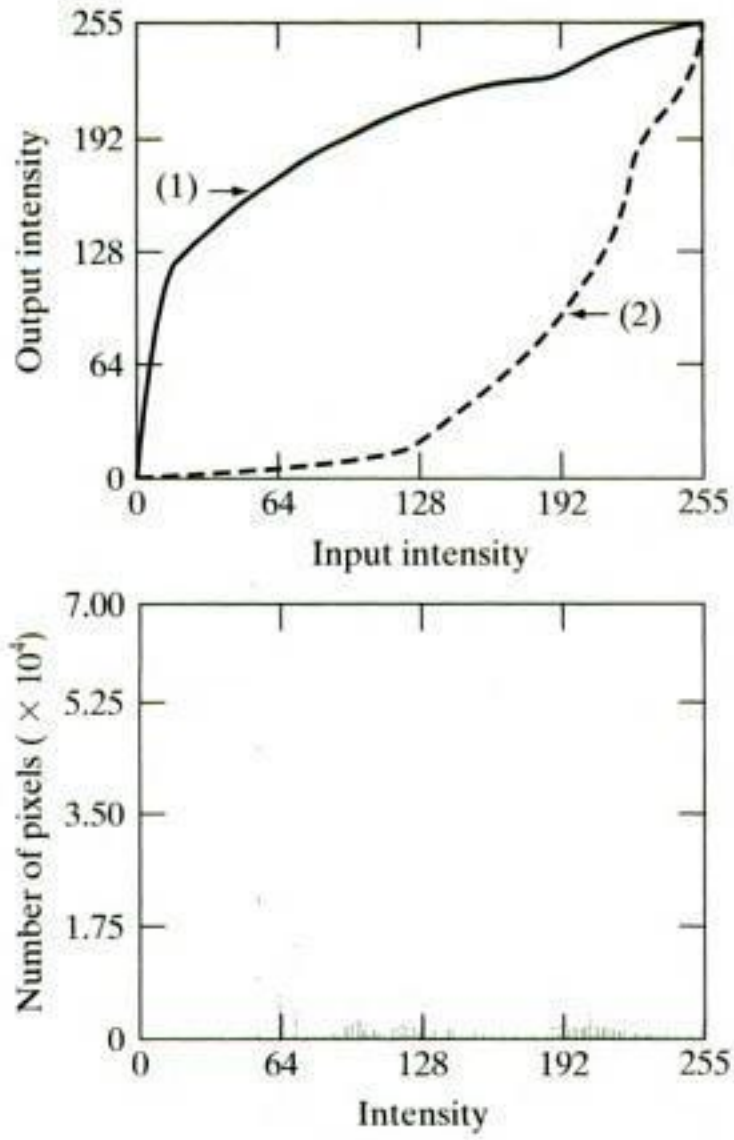
case. The histogram of this image is shown in Fig. 3.24(c). Note how all the intensity levels are biased toward the upper one-half of the gray scale.

Because the problem with the transformation function in Fig. 3.24(a) was caused by a large concentration of pixels in the original image with levels near 0, a reasonable approach is to modify the histogram of that image so that it does not have this property. Figure 3.25(a) shows a *manually specified* function that preserves the general shape of the original histogram, but has a smoother transition of levels in the dark region of the gray scale. Sampling this function into 256 equally spaced discrete values produced the desired specified histogram. The transformation function $G(z)$ obtained from this histogram using Eq. (3.3-14) is labeled transformation (1) in Fig. 3.25(b). Similarly, the inverse transformation $G^{-1}(s)$ from Eq. (3.3-16) (obtained using the step-by-step procedure discussed earlier) is labeled transformation (2) in Fig. 3.25(b). The enhanced image in Fig. 3.25(c) was obtained by applying transformation (2) to the pixels of the histogram-equalized image in Fig. 3.24(b). The improvement of the histogram-specified image over the result obtained by histogram equalization is evident by comparing these two images. It is of interest to note that a rather modest change in the original histogram was all that was required to obtain a significant improvement in appearance. Figure 3.25(d) shows the histogram of Fig. 3.25(c). The most distinguishing feature of this histogram is how its low end has shifted right toward the lighter region of the gray scale (but not excessively so), as desired. ■

a c
b
d

FIGURE 3.25
(a) Specified histogram.
(b) Transformations.
(c) Enhanced image using mappings from curve (2).
(d) Histogram of (c).





Although it probably is obvious by now, we emphasize before leaving this section that histogram specification is, for the most part, a trial-and-error process. One can use guidelines learned from the problem at hand, just as we did in the preceding example. At times, there may be cases in which it is possible to formulate what an "average" histogram should look like and use that as the specified histogram. In cases such as these, histogram specification becomes a straightforward process. In general, however, there are no rules for specifying histograms, and one must resort to analysis on a case-by-case basis for any given enhancement task.

3.3.3 Local Histogram Processing

The histogram processing methods discussed in the previous two sections are *global*, in the sense that pixels are modified by a transformation function based on the intensity distribution of an entire image. Although this global approach is suitable for overall enhancement, there are cases in which it is necessary to enhance details over small areas in an image. The number of pixels in these areas may have negligible influence on the computation of a global transformation whose shape does not necessarily guarantee the desired local enhancement. The solution is to devise transformation functions based on the intensity distribution in a neighborhood of every pixel in the image.

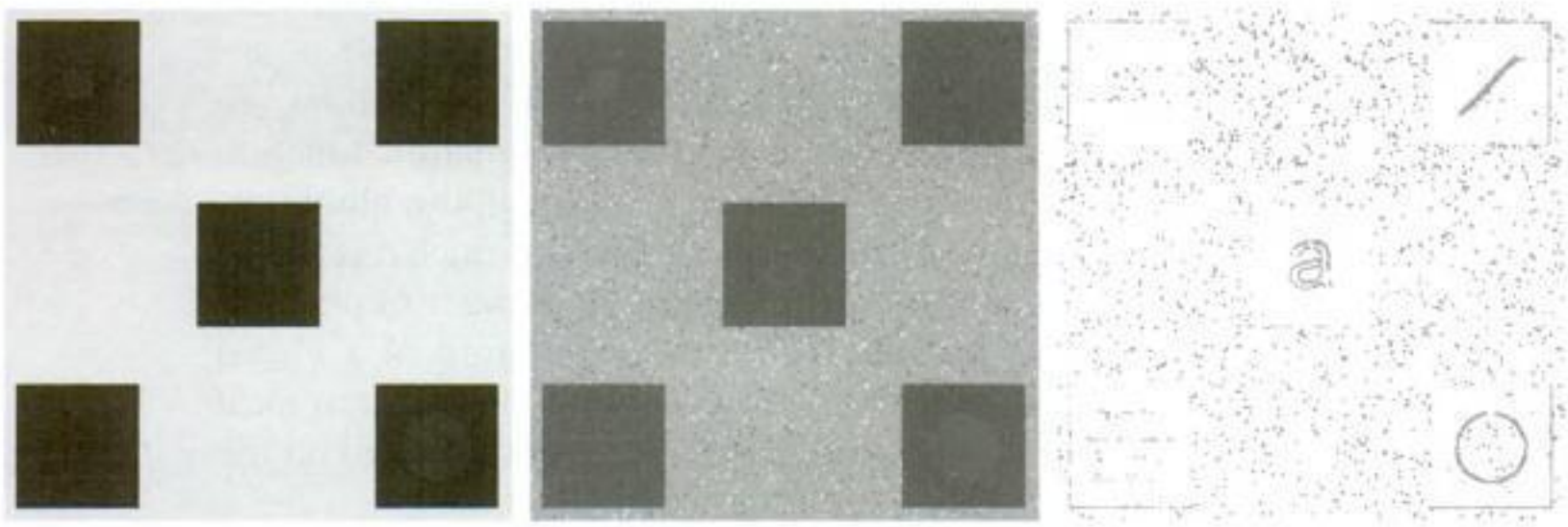
The histogram processing techniques previously described are easily adapted to local enhancement. The procedure is to define a neighborhood and move its center from pixel to pixel. At each location, the histogram of the points in the neighborhood is computed and either a histogram equalization or histogram specification transformation function is obtained. This function is then used to map the intensity of the pixel centered in the neighborhood. The center of the neighborhood region is then moved to an adjacent pixel location and the procedure is repeated. Because only one row or column of the neighborhood changes during a pixel-to-pixel translation of the neighborhood, updating the histogram obtained in the previous location with the new data introduced at each motion step is possible (Problem 3.12). This approach has obvious advantages over repeatedly computing the histogram of all pixels in the neighborhood region each time the region is moved one pixel location. Another approach used sometimes to reduce computation is to utilize nonoverlapping regions, but this method usually produces an undesirable "blocky" effect.

EXAMPLE 3.10: Local histogram equalization.

■ Figure 3.26(a) shows an 8-bit, 512 × 512 image that at first glance appears to contain five black squares on a gray background. The image is slightly noisy, but the noise is imperceptible. Figure 3.26(b) shows the result of global histogram equalization. As often is the case with histogram equalization of smooth, noisy regions, this image shows significant enhancement of the noise. Aside from the noise, however, Fig. 3.26(b) does not reveal any new significant details from the original, other than a very faint hint that the top left and bottom right squares contain an object. Figure 3.26(c) was obtained using local histogram equalization with a neighborhood of size 3 × 3. Here, we see significant detail contained within the dark squares. The intensity values of these objects were too close to the intensity of the large squares, and their sizes were too small, to influence global histogram equalization significantly enough to show this detail. ■

3.3.4 Using Histogram Statistics for Image Enhancement

Statistics obtained directly from an image histogram can be used for image enhancement. Let r denote a discrete random variable representing intensity values in the range $[0, L-1]$, and let $p(r_i)$ denote the normalized histogram



a b c

FIGURE 3.26 (a) Original image. (b) Result of global histogram equalization. (c) Result of local histogram equalization applied to (a), using a neighborhood of size 3 × 3.

component corresponding to value r_i. As indicated previously, we may view $p(r_i)$ as an estimate of the probability that intensity r_i occurs in the image from which the histogram was obtained.

As we discussed in Section 2.6.8, the nth moment of r about its mean is defined as

$$\mu_n(r) = \sum_{i=0}^{L-1} (r_i - m)^n p(r_i) \tag{3.3-17}$$

We follow convention in using m for the mean value. Do not confuse it with the same symbol used to denote the number of rows in an $m \times n$ neighborhood, in which we also follow notational convention.

where m is the mean (average intensity) value of r (i.e., the average intensity of the pixels in the image):

$$m = \sum_{i=0}^{L-1} r_i p(r_i) \tag{3.3-18}$$

The second moment is particularly important:

$$\mu_2(r) = \sum_{i=0}^{L-1} (r_i - m)^2 p(r_i) \tag{3.3-19}$$

We recognize this expression as the intensity variance, normally denoted by σ^2 (recall that the standard deviation is the square root of the variance). Whereas the mean is a measure of average intensity, the variance (or standard deviation) is a measure of contrast in an image. Observe that all moments are computed easily using the preceding expressions once the histogram has been obtained from a given image.

When working with only the mean and variance, it is common practice to estimate them directly from the sample values, without computing the histogram. Appropriately, these estimates are called the *sample mean* and *sample variance*. They are given by the following familiar expressions from basic statistics:

$$m = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \tag{3.3-20}$$

and

$$\sigma^2 = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[f(x, y) - m\right]^2 \tag{3.3-21}$$

for $x = 0, 1, 2, \ldots, M - 1$ and $y = 0, 1, 2, \ldots, N - 1$. In other words, as we know, the mean intensity of an image can be obtained simply by summing the values of all its pixels and dividing the sum by the total number of pixels in the image. A similar interpretation applies to Eq. (3.3-21). As we illustrate in the following example, the results obtained using these two equations are identical to the results obtained using Eqs. (3.3-18) and (3.3-19), provided that the histogram used in these equations is computed from the same image used in Eqs. (3.3-20) and (3.3-21).

The denominator in Eq. (3.3-21) is written sometimes as $MN - 1$ instead of MN. This is done to obtain a so-called *unbiased* estimate of the variance. However, we are more interested in Eqs. (3.3-21) and (3.3-19) agreeing when the histogram in the latter equation is computed from the same image used in Eq. (3.3-21). For this we require the MN term. The difference is negligible for any image of practical size.

■ Before proceeding, it will be useful to work through a simple numerical example to fix ideas. Consider the following 2-bit image of size 5×5:

EXAMPLE 3.11: Computing histogram statistics.

$$\begin{array}{ccccc} 0 & 0 & 1 & 1 & 2 \\ 1 & 2 & 3 & 0 & 1 \\ 3 & 3 & 2 & 2 & 0 \\ 2 & 3 & 1 & 0 & 0 \\ 1 & 1 & 3 & 2 & 2 \end{array}$$

The pixels are represented by 2 bits; therefore, $L = 4$ and the intensity levels are in the range $[0, 3]$. The total number of pixels is 25, so the histogram has the components

$$p(r_0) = \frac{6}{25} = 0.24;\ p(r_1) = \frac{7}{25} = 0.28;$$

$$p(r_2) = \frac{7}{25} = 0.28;\ p(r_3) = \frac{5}{25} = 0.20$$

where the numerator in $p(r_i)$ is the number of pixels in the image with intensity level r_i. We can compute the average value of the intensities in the image using Eq. (3.3-18):

$$\begin{aligned} m &= \sum_{i=0}^{3} r_i\, p(r_i) \\ &= (0)(0.24) + (1)(0.28) + (2)(0.28) + (3)(0.20) \\ &= 1.44 \end{aligned}$$

Letting $f(x, y)$ denote the preceding 5×5 array and using Eq. (3.3-20), we obtain

$$\begin{aligned} m &= \frac{1}{25}\sum_{x=0}^{4}\sum_{y=0}^{4} f(x, y) \\ &= 1.44 \end{aligned}$$

As expected, the results agree. Similarly, the result for the variance is the same (1.1264) using either Eq. (3.3-19) or (3.3-21). ■

We consider two uses of the mean and variance for enhancement purposes. The *global* mean and variance are computed over an entire image and are useful for gross adjustments in overall intensity and contrast. A more powerful use of these parameters is in local enhancement, where the *local* mean and variance are used as the basis for making changes that depend on image characteristics in a neighborhood about each pixel in an image.

Let (x, y) denote the coordinates of any pixel in a given image, and let S_{xy} denote a neighborhood (subimage) of specified size, centered on (x, y). The mean value of the pixels in this neighborhood is given by the expression

$$m_{S_{xy}} = \sum_{i=0}^{L-1} r_i p_{S_{xy}}(r_i) \tag{3.3-22}$$

where $p_{S_{xy}}$ is the histogram of the pixels in region S_{xy}. This histogram has L components, corresponding to the L possible intensity values in the input image. However, many of the components are 0, depending on the size of S_{xy}. For example, if the neighborhood is of size 3×3 and $L = 256$, only between 1 and 9 of the 256 components of the histogram of the neighborhood will be nonzero. These non-zero values will correspond to the number of different intensities in S_{xy} (the maximum number of possible different intensities in a 3×3 region is 9, and the minimum is 1).

The variance of the pixels in the neighborhood similarly is given by

$$\sigma^2_{S_{xy}} = \sum_{i=0}^{L-1} (r_i - m_{S_{xy}})^2 p_{S_{xy}}(r_i) \tag{3.3-23}$$

As before, the local mean is a measure of average intensity in neighborhood S_{xy}, and the local variance (or standard deviation) is a measure of intensity contrast in that neighborhood. Expressions analogous to (3.3-20) and (3.3-21) can be written for neighborhoods. We simply use the pixel values in the neighborhoods in the summations and the number of pixels in the neighborhood in the denominator.

As the following example illustrates, an important aspect of image processing using the local mean and variance is the flexibility they afford in developing simple, yet powerful enhancement techniques based on statistical measures that have a close, predictable correspondence with image appearance.

EXAMPLE 3.12: Local enhancement using histogram statistics.

Figure 3.27(a) shows an SEM (scanning electron microscope) image of a tungsten filament wrapped around a support. The filament in the center of the image and its support are quite clear and easy to study. There is another filament structure on the right, dark side of the image, but it is almost imperceptible, and its size and other characteristics certainly are not easily discernable. Local enhancement by contrast manipulation is an ideal approach to problems such as this, in which parts of an image may contain hidden features.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

The name *filter* is borrowed from frequency domain processing, which is the topic of the next chapter, where "filtering" refers to accepting (passing) or rejecting certain frequency components. For example, a filter that passes low frequencies is called a *lowpass* filter. The net effect produced by a lowpass filter is to blur (smooth) an image. We can accomplish a similar smoothing directly on the image itself by using spatial filters (also called spatial *masks*, *kernels*, *templates*, and *windows*). In fact, as we show in Chapter 4, there is a one-to-one correspondence between linear spatial filters and filters in the frequency domain. However, spatial filters offer considerably more versatility because, as you will see later, they can be used also for nonlinear filtering, something we cannot do in the frequency domain.

See Section 2.6.2 regarding linearity.

3.4.1 The Mechanics of Spatial Filtering

In Fig. 3.1, we explained briefly that a spatial filter consists of (1) a *neighborhood*, (typically a small rectangle), and (2) a *predefined operation* that is performed on the image pixels encompassed by the neighborhood. Filtering creates a new pixel with coordinates equal to the coordinates of the center of the neighborhood, and whose value is the result of the filtering operation.[†] A processed (filtered) image is generated as the center of the filter visits each pixel in the input image. If the operation performed on the image pixels is linear, then the filter is called a *linear spatial filter*. Otherwise, the filter is *nonlinear*. We focus attention first on linear filters and then illustrate some simple nonlinear filters. Section 5.3 contains a more comprehensive list of nonlinear filters and their application.

Figure 3.28 illustrates the mechanics of linear spatial filtering using a 3×3 neighborhood. At any point (x, y) in the image, the response, $g(x, y)$, of the filter is the sum of products of the filter coefficients and the image pixels encompassed by the filter:

$$g(x, y) = w(-1, -1)f(x - 1, y - 1) + w(-1, 0)f(x - 1, y) + \ldots + w(0, 0)f(x, y) + \ldots + w(1, 1)f(x + 1, y + 1)$$

Observe that the center coefficient of the filter, $w(0, 0)$, aligns with the pixel at location (x, y). For a mask of size $m \times n$, we assume that $m = 2a + 1$ and $n = 2b + 1$, where a and b are positive integers. This means that our focus in the following discussion is on filters of odd size, with the smallest being of size 3×3. In general, linear spatial filtering of an image of size $M \times N$ with a filter of size $m \times n$ is given by the expression:

It certainly is possible to work with filters of even size or mixed even and odd sizes. However, working with odd sizes simplifies indexing and also is more intuitive because the filters have centers falling on integer values.

$$g(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t)f(x + s, y + t)$$

where x and y are varied so that each pixel in w visits every pixel in f.

[†] The filtered pixel value typically is assigned to a corresponding location in a new image created to hold the results of filtering. It is seldom the case that filtered pixels replace the values of the corresponding location in the original image, as this would change the content of the image while filtering still is being performed.

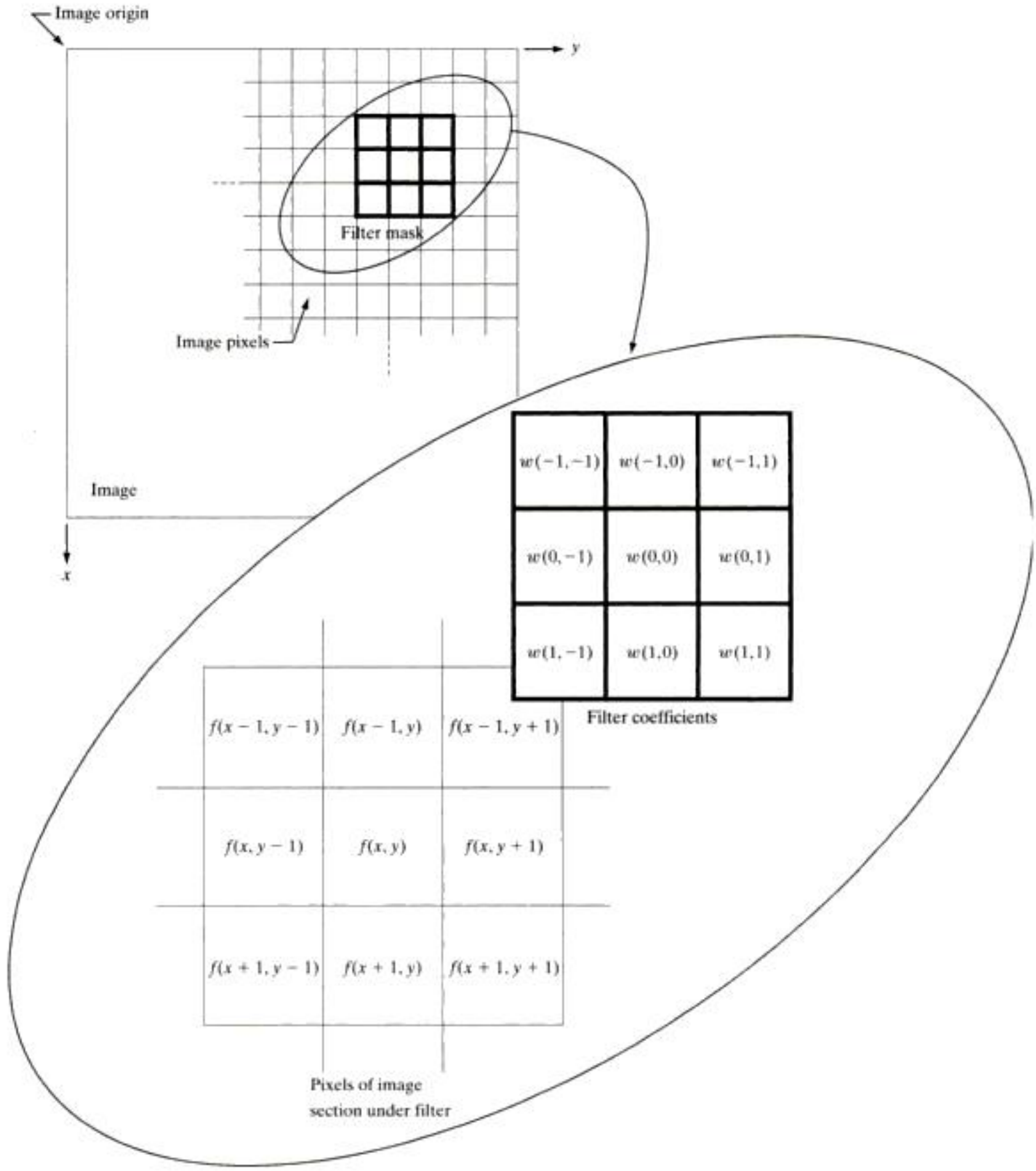


FIGURE 3.28 The mechanics of linear spatial filtering using a 3×3 filter mask. The form chosen to denote the coordinates of the filter mask coefficients simplifies writing expressions for linear filtering.

3.4.2 Spatial Correlation and Convolution

There are two closely related concepts that must be understood clearly when performing linear spatial filtering. One is *correlation* and the other is *convolution*. Correlation is the process of moving a filter mask over the image and computing the sum of products at each location, exactly as explained in the previous section. The mechanics of convolution are the same, except that the filter is first rotated by 180°. The best way to explain the differences between the two concepts is by example. We begin with a 1-D illustration.

Figure 3.29(a) shows a 1-D function, f, and a filter, w, and Fig. 3.29(b) shows the starting position to perform correlation. The first thing we note is that there

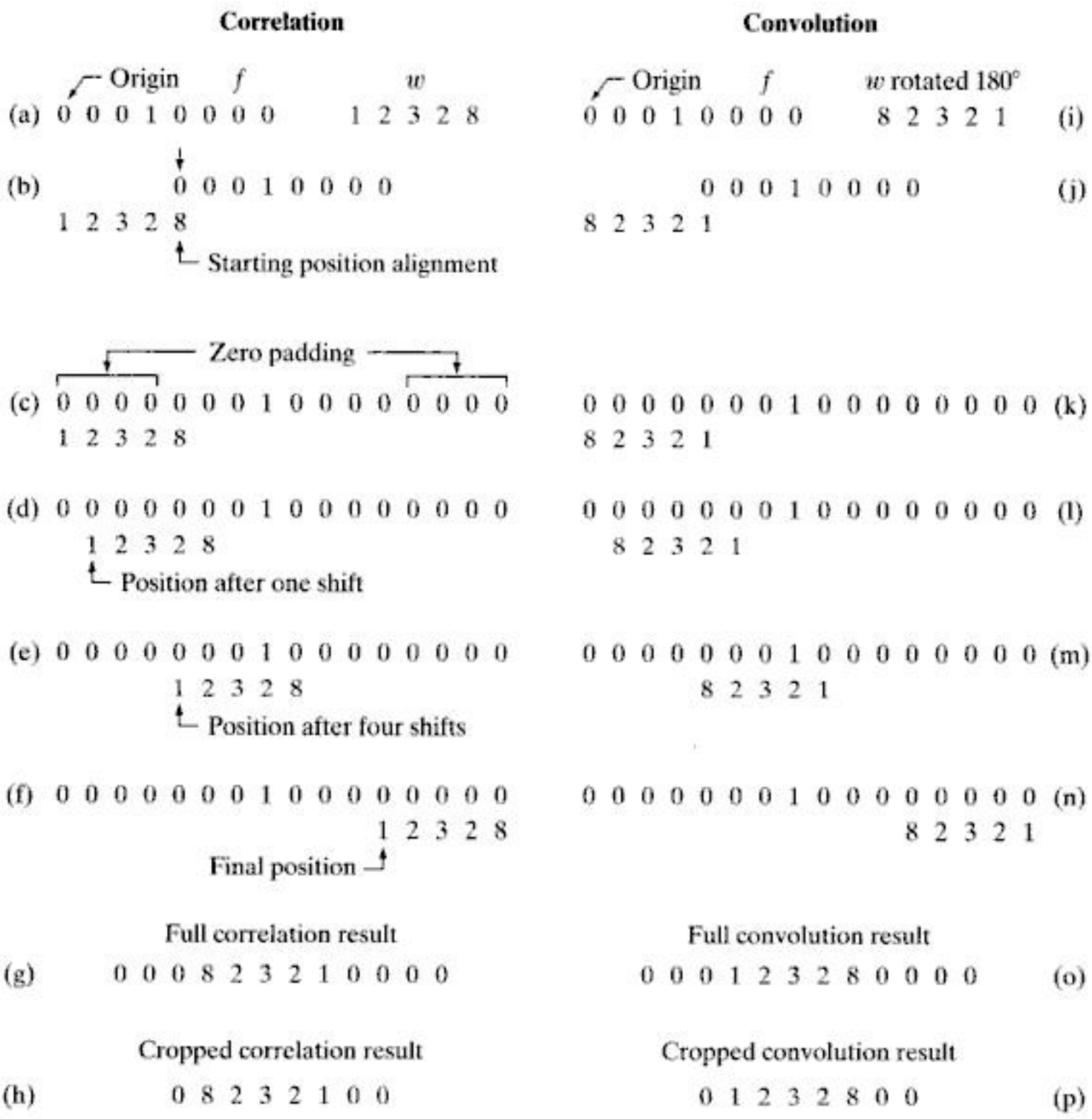


FIGURE 3.29 Illustration of 1-D correlation and convolution of a filter with a discrete unit impulse. Note that correlation and convolution are functions of *displacement*.

are parts of the functions that do not overlap. The solution to this problem is to pad f with enough 0s on either side to allow each pixel in w to visit every pixel in f. If the filter is of size m, we need $m - 1$ 0s on either side of f. Figure 3.29(c) shows a properly padded function. The first value of correlation is the sum of products of f and w for the initial position shown in Fig. 3.29(c) (the sum of products is 0). This corresponds to a displacement $x = 0$. To obtain the second value of correlation, we shift w one pixel location to the right (a displacement of $x = 1$) and compute the sum of products. The result again is 0. In fact, the first nonzero result is when $x = 3$, in which case the 8 in w overlaps the 1 in f and the result of correlation is 8. Proceeding in this manner, we obtain the full correlation result in Fig. 3.29(g). Note that it took 12 values of x (i.e., $x = 0, 1, 2, \ldots, 11$) to fully slide w past f so that each pixel in w visited every pixel in f. Often, we like to work with correlation arrays that are the same size as f, in which case we crop the full correlation to the size of the original function, as Fig. 3.29(h) shows.

Zero padding is not the only option. For example, we could duplicate the value of the first and last element $m - 1$ times on each side of f, or mirror the first and last $m - 1$ elements and use the mirrored values for padding.

There are two important points to note from the discussion in the preceding paragraph. First, correlation is a function of *displacement* of the filter. In other words, the first value of correlation corresponds to zero displacement of the filter, the second corresponds to one unit displacement, and so on. The second thing to notice is that correlating a filter w with a function that contains all 0s and a single 1 yields a result that is a *copy* of w, but *rotated* by 180°. We call a function that contains a single 1 with the rest being 0s a *discrete unit impulse*. So we conclude that correlation of a function with a discrete unit impulse yields a rotated version of the function at the location of the impulse.

The concept of convolution is a cornerstone of linear system theory. As you will learn in Chapter 4, a fundamental property of convolution is that convolving a function with a unit impulse yields a copy of the function at the location of the impulse. We saw in the previous paragraph that correlation yields a copy of the function also, but rotated by 180°. Therefore, if we *pre-rotate* the filter and perform the same sliding sum of products operation, we should be able to obtain the desired result. As the right column in Fig. 3.29 shows, this indeed is the case. Thus, we see that to perform convolution all we do is rotate one function by 180° and perform the same operations as in correlation. As it turns out, it makes no difference which of the two functions we rotate.

Note that rotation by 180° is equivalent to flipping the function horizontally.

The preceding concepts extend easily to images, as Fig. 3.30 shows. For a filter of size $m \times n$, we pad the image with a minimum of $m - 1$ rows of 0s at the top and bottom and $n - 1$ columns of 0s on the left and right. In this case, m and n are equal to 3, so we pad f with two rows of 0s above and below and two columns of 0s to the left and right, as Fig. 3.30(b) shows. Figure 3.30(c) shows the initial position of the filter mask for performing correlation, and Fig. 3.30(d) shows the full correlation result. Figure 3.30(e) shows the corresponding cropped result. Note again that the result is rotated by 180°. For convolution, we pre-rotate the mask as before and repeat the sliding sum of products just explained. Figures 3.30(f) through (h) show the result. You see again that convolution of a function with an impulse copies the function at the location of the impulse. It should be clear that, if the filter mask is symmetric, correlation and convolution yield the same result.

In 2-D, rotation by 180° is equivalent to flipping the mask along one axis and then the other.

If, instead of containing a single 1, image f in Fig. 3.30 had contained a region identically equal to w, the value of the correlation function (after normalization) would have been maximum when w was centered on that region of f. Thus, as you will see in Chapter 12, correlation can be used also to find *matches* between images.

Summarizing the preceding discussion in equation form, we have that the correlation of a filter $w(x, y)$ of size $m \times n$ with an image $f(x, y)$, denoted as $w(x, y) \star f(x, y)$, is given by the equation listed at the end of the last section, which we repeat here for convenience:

$$w(x, y) \star f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t) \qquad (3.4\text{-}1)$$

This equation is evaluated for all values of the displacement variables x and y so that all elements of w visit every pixel in f, where we assume that f has been padded appropriately. As explained earlier, $a = (m - 1)/2$, $b = (n - 1)/2$, and we assume for notational convenience that m and n are odd integers.

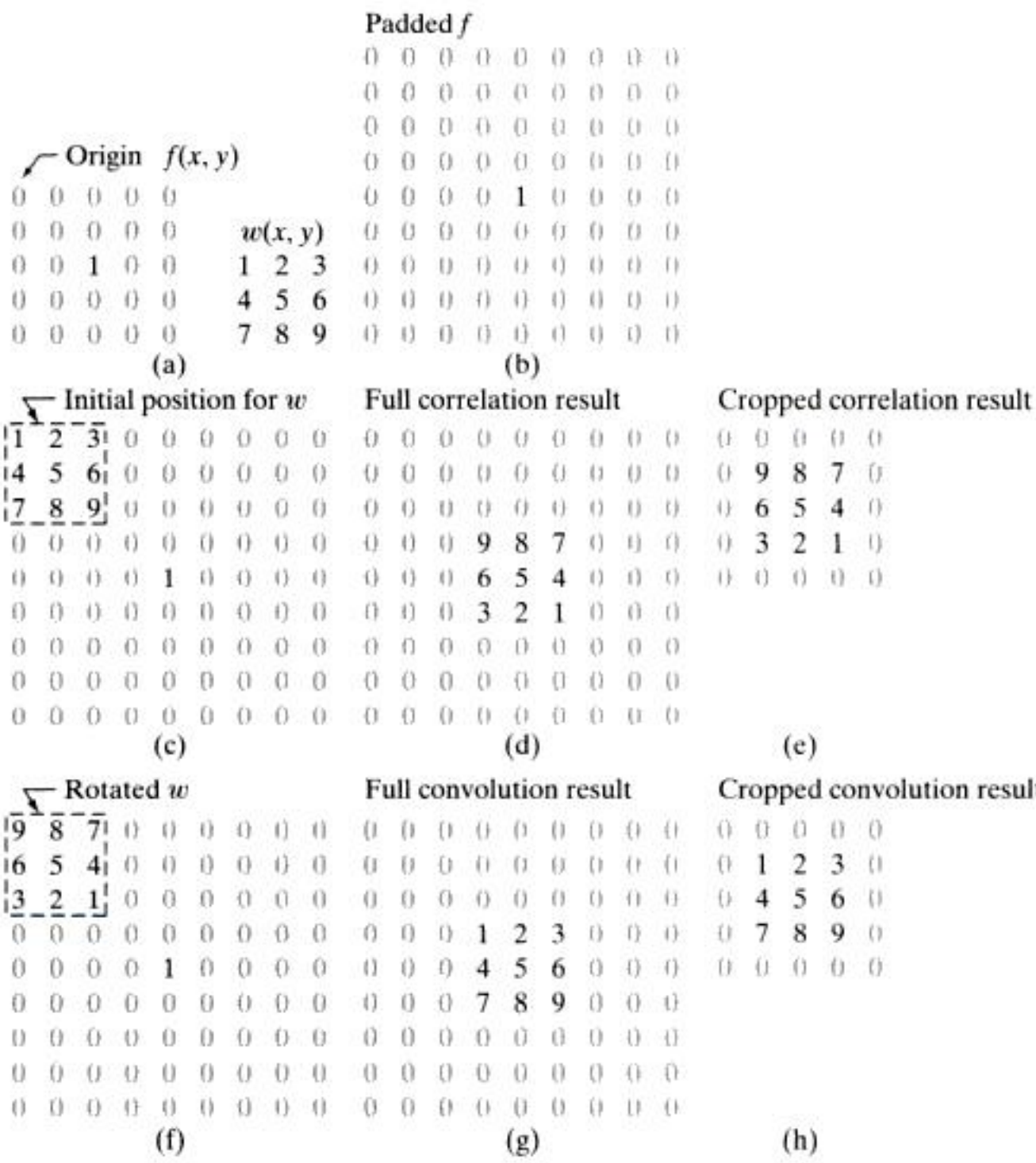


FIGURE 3.30 Correlation (middle row) and convolution (last row) of a 2-D filter with a 2-D discrete, unit impulse. The 0s are shown in gray to simplify visual analysis.

In a similar manner, the convolution of $w(x, y)$ and $f(x, y)$, denoted by $w(x, y) \star f(x, y)$,[†] is given by the expression

$$w(x, y) \star f(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x - s, y - t) \qquad (3.4\text{-}2)$$

where the minus signs on the right flip f (i.e., rotate it by 180°). Flipping and shifting f instead of w is done for notational simplicity and also to follow convention. The result is the same. As with correlation, this equation is evaluated for all values of the displacement variables x and y so that every element of w visits every pixel in f, which we assume has been padded appropriately. You should expand Eq. (3.4-2) for a 3×3 mask and convince yourself that the result using this equation is identical to the example in Fig. 3.30. In practice, we frequently work with an algorithm that implements

Often, when the meaning is clear, we denote the result of correlation or convolution by a function $g(x, y)$, instead of writing $w(x, y) \star f(x, y)$ or $w(x, y) \star f(x, y)$. For example, see the equation at the end of the previous section, and Eq. (3.5-1).

[†] Because correlation and convolution are commutative, we have that $w(x, y) \star f(x, y) = f(x, y) \star w(x, y)$ and $w(x, y) \star f(x, y) = f(x, y) \star w(x, y)$.

Eq. (3.4-1). If we want to perform correlation, we input w into the algorithm; for convolution, we input w rotated by 180°. The reverse is true if an algorithm that implements Eq. (3.4-2) is available instead.

As mentioned earlier, convolution is a cornerstone of linear system theory. As you will learn in Chapter 4, the property that the convolution of a function with a unit impulse copies the function at the location of the impulse plays a central role in a number of important derivations. We will revisit convolution in Chapter 4 in the context of the Fourier transform and the convolution theorem. Unlike Eq. (3.4-2), however, we will be dealing with convolution of functions that are of the same size. The form of the equation is the same, but the limits of summation are different.

Using correlation or convolution to perform spatial filtering is a matter of preference. In fact, because either Eq. (3.4-1) or (3.4-2) can be made to perform the function of the other by a simple rotation of the filter, what is important is that the filter mask used in a given filtering task be specified in a way that corresponds to the intended operation. All the linear spatial filtering results in this chapter are based on Eq. (3.4-1).

Finally, we point out that you are likely to encounter the terms, *convolution filter*, *convolution mask* or *convolution kernel* in the image processing literature. As a rule, these terms are used to denote a spatial filter, and not necessarily that the filter will be used for true convolution. Similarly, "convolving a mask with an image" often is used to denote the sliding, sum-of-products process we just explained, and does not necessarily differentiate between correlation and convolution. Rather, it is used generically to denote either of the two operations. This imprecise terminology is a frequent source of confusion.

3.4.3 Vector Representation of Linear Filtering

When interest lies in the characteristic response, R, of a mask either for correlation or convolution, it is convenient sometimes to write the sum of products as

$$\begin{aligned} R &= w_1 z_1 + w_2 z_2 + \ldots + w_{mn} z_{mn} \\ &= \sum_{k=1}^{mn} w_k z_k \\ &= \mathbf{w}^T \mathbf{z} \end{aligned} \tag{3.4-3}$$

Consult the Tutorials section of the book Web site for a brief review of vectors and matrices.

where the ws are the coefficients of an $m \times n$ filter and the zs are the corresponding image intensities encompassed by the filter. If we are interested in using Eq. (3.4-3) for correlation, we use the mask as given. To use the same equation for convolution, we simply rotate the mask by 180°, as explained in the last section. It is implied that Eq. (3.4-3) holds for a particular pair of coordinates (x, y). You will see in the next section why this notation is convenient for explaining the characteristics of a given linear filter.

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

FIGURE 3.31 Another representation of a general 3×3 filter mask.

As an example, Fig. 3.31 shows a general 3×3 mask with coefficients labeled as above. In this case, Eq. (3.4-3) becomes

$$\begin{aligned} R &= w_1 z_1 + w_2 z_2 + \ldots + w_9 z_9 \\ &= \sum_{k=1}^{9} w_k z_k \\ &= \mathbf{w}^T \mathbf{z} \end{aligned} \tag{3.4-4}$$

where $\mathbf{w}$ and $\mathbf{z}$ are 9-dimensional vectors formed from the coefficients of the mask and the image intensities encompassed by the mask, respectively.

3.4.4 Generating Spatial Filter Masks

Generating an $m \times n$ *linear* spatial filter requires that we specify mn mask coefficients. In turn, these coefficients are selected based on what the filter is supposed to do, keeping in mind that all we can do with linear filtering is to implement a sum of products. For example, suppose that we want to replace the pixels in an image by the average intensity of a 3×3 neighborhood centered on those pixels. The average value at any location (x, y) in the image is the sum of the nine intensity values in the 3×3 neighborhood centered on (x, y) divided by 9. Letting z_i, $i = 1, 2, \ldots, 9$, denote these intensities, the average is

$$R = \frac{1}{9} \sum_{i=1}^{9} z_i$$

But this is the same as Eq. (3.4-4) with coefficient values $w_i = 1/9$. In other words, a linear filtering operation with a 3×3 mask whose coefficients are $1/9$ implements the desired averaging. As we discuss in the next section, this operation results in image smoothing. We discuss in the following sections a number of other filter masks based on this basic approach.

In some applications, we have a continuous function of two variables, and the objective is to obtain a spatial filter mask based on that function. For example, a Gaussian function of two variables has the basic form

$$h(x, y) = e^{-\frac{x^2 + y^2}{2\sigma^2}}$$

where σ is the standard deviation and, as usual, we assume that coordinates x and y are integers. To generate, say, a 3×3 filter mask from this function, we

sample it about its center. Thus, $w_1 = h(-1, -1), w_2 = h(-1, 0), \ldots, w_9 = h(1, 1)$. An $m \times n$ filter mask is generated in a similar manner. Recall that a 2-D Gaussian function has a bell shape, and that the standard deviation controls the "tightness" of the bell.

Generating a *nonlinear* filter requires that we specify the size of a neighborhood and the operation(s) to be performed on the image pixels contained in the neighborhood. For example, recalling that the max operation is nonlinear (see Section 2.6.2), a 5×5 max filter centered at an arbitrary point (x, y) of an image obtains the maximum intensity value of the 25 pixels and assigns that value to location (x, y) in the processed image. Nonlinear filters are quite powerful, and in some applications can perform functions that are beyond the capabilities of linear filters, as we show later in this chapter and in Chapter 5.

3.5 Smoothing Spatial Filters

Smoothing filters are used for blurring and for noise reduction. Blurring is used in preprocessing tasks, such as removal of small details from an image prior to (large) object extraction, and bridging of small gaps in lines or curves. Noise reduction can be accomplished by blurring with a linear filter and also by nonlinear filtering.

3.5.1 Smoothing Linear Filters

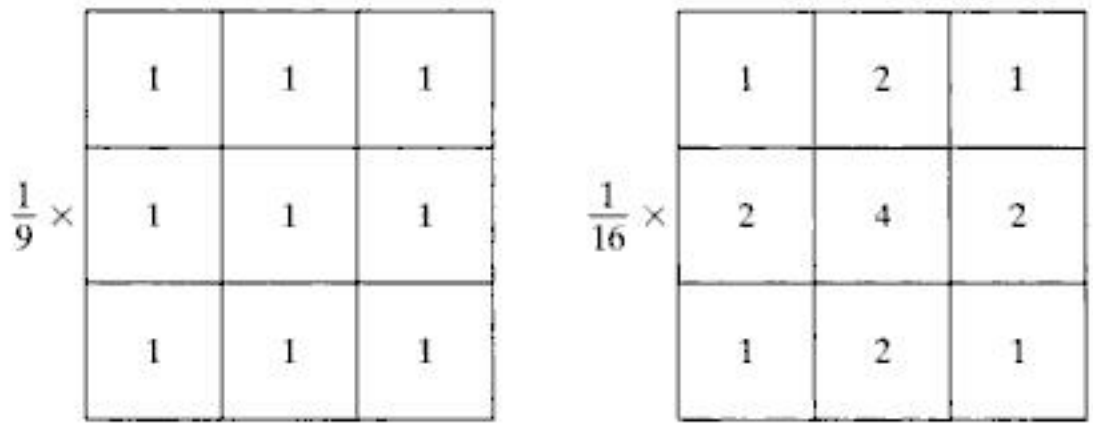
The output (response) of a smoothing, linear spatial filter is simply the average of the pixels contained in the neighborhood of the filter mask. These filters sometimes are called *averaging filters*. As mentioned in the previous section, they also are referred to a *lowpass filters*.

The idea behind smoothing filters is straightforward. By replacing the value of every pixel in an image by the average of the intensity levels in the neighborhood defined by the filter mask, this process results in an image with reduced "sharp" transitions in intensities. Because random noise typically consists of sharp transitions in intensity levels, the most obvious application of smoothing is noise reduction. However, edges (which almost always are desirable features of an image) also are characterized by sharp intensity transitions, so averaging filters have the undesirable side effect that they blur edges. Another application of this type of process includes the smoothing of false contours that result from using an insufficient number of intensity levels, as discussed in Section 2.4.3. A major use of averaging filters is in the reduction of "irrelevant" detail in an image. By "irrelevant" we mean pixel regions that are small with respect to the size of the filter mask. This latter application is illustrated later in this section.

Figure 3.32 shows two 3×3 smoothing filters. Use of the first filter yields the standard average of the pixels under the mask. This can best be seen by substituting the coefficients of the mask into Eq. (3.4-4):

$$R = \frac{1}{9}\sum_{i=1}^{9} z_i$$

which is the average of the intensity levels of the pixels in the 3×3 neighborhood defined by the mask, as discussed earlier. Note that, instead of being 1/9,



a b

FIGURE 3.32 Two 3×3 smoothing (averaging) filter masks. The constant multiplier in front of each mask is equal to 1 divided by the sum of the values of its coefficients, as is required to compute an average.

the coefficients of the filter are all 1s. The idea here is that it is computationally more efficient to have coefficients valued 1. At the end of the filtering process the entire image is divided by 9. An $m \times n$ mask would have a normalizing constant equal to $1/mn$. A spatial averaging filter in which all coefficients are equal sometimes is called a *box filter*.

The second mask in Fig. 3.32 is a little more interesting. This mask yields a so-called *weighted average*, terminology used to indicate that pixels are multiplied by different coefficients, thus giving more importance (weight) to some pixels at the expense of others. In the mask shown in Fig. 3.32(b) the pixel at the center of the mask is multiplied by a higher value than any other, thus giving this pixel more importance in the calculation of the average. The other pixels are inversely weighted as a function of their distance from the center of the mask. The diagonal terms are further away from the center than the orthogonal neighbors (by a factor of $\sqrt{2}$) and, thus, are weighed less than the immediate neighbors of the center pixel. The basic strategy behind weighing the center point the highest and then reducing the value of the coefficients as a function of increasing distance from the origin is simply an attempt to reduce blurring in the smoothing process. We could have chosen other weights to accomplish the same general objective. However, the sum of all the coefficients in the mask of Fig. 3.32(b) is equal to 16, an attractive feature for computer implementation because it is an integer power of 2. In practice, it is difficult in general to see differences between images smoothed by using either of the masks in Fig. 3.32, or similar arrangements, because the area spanned by these masks at any one location in an image is so small.

With reference to Eq. (3.4-1), the general implementation for filtering an $M \times N$ image with a weighted averaging filter of size $m \times n$ (m and n odd) is given by the expression

$$g(x, y) = \frac{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t)}{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t)} \tag{3.5-1}$$

The parameters in this equation are as defined in Eq. (3.4-1). As before, it is understood that the complete filtered image is obtained by applying Eq. (3.5-1) for $x = 0, 1, 2, \ldots, M - 1$ and $y = 0, 1, 2, \ldots, N - 1$. The denominator in

Eq. (3.5-1) is simply the sum of the mask coefficients and, therefore, it is a constant that needs to be computed only once.

EXAMPLE 3.13: Image smoothing with masks of various sizes.

■ The effects of smoothing as a function of filter size are illustrated in Fig. 3.33, which shows an original image and the corresponding smoothed results obtained using square averaging filters of sizes $m = 3, 5, 9, 15$, and 35 pixels, respectively. The principal features of these results are as follows: For $m = 3$, we note a general slight blurring throughout the entire image but, as expected, details that are of approximately the same size as the filter mask are affected considerably more. For example, the 3×3 and 5×5 black squares in the image, the small letter "a," and the fine grain noise show significant blurring when compared to the rest of the image. Note that the noise is less pronounced, and the jagged borders of the characters were pleasingly smoothed.

The result for $m = 5$ is somewhat similar, with a slight further increase in blurring. For $m = 9$ we see considerably more blurring, and the 20% black circle is not nearly as distinct from the background as in the previous three images, illustrating the blending effect that blurring has on objects whose intensities are close to that of its neighboring pixels. Note the significant further smoothing of the noisy rectangles. The results for $m = 15$ and 35 are extreme with respect to the sizes of the objects in the image. This type of aggresive blurring generally is used to eliminate small objects from an image. For instance, the three small squares, two of the circles, and most of the noisy rectangle areas have been blended into the background of the image in Fig. 3.33(f). Note also in this figure the pronounced black border. This is a result of padding the border of the original image with 0s (black) and then trimming off the padded area after filtering. Some of the black was blended into all filtered images, but became truly objectionable for the images smoothed with the larger filters. ■

As mentioned earlier, an important application of spatial averaging is to blur an image for the purpose of getting a gross representation of objects of interest, such that the intensity of smaller objects blends with the background and larger objects become "bloblike" and easy to detect. The size of the mask establishes the relative size of the objects that will be blended with the background. As an illustration, consider Fig. 3.34(a), which is an image from the Hubble telescope in orbit around the Earth. Figure 3.34(b) shows the result of applying a 15×15 averaging mask to this image. We see that a number of objects have either blended with the background or their intensity has diminished considerably. It is typical to follow an operation like this with thresholding to eliminate objects based on their intensity. The result of using the thresholding function of Fig. 3.2(b) with a threshold value equal to 25% of the highest intensity in the blurred image is shown in Fig. 3.34(c). Comparing this result with the original image, we see that it is a reasonable representation of what we would consider to be the largest, brightest objects in that image.

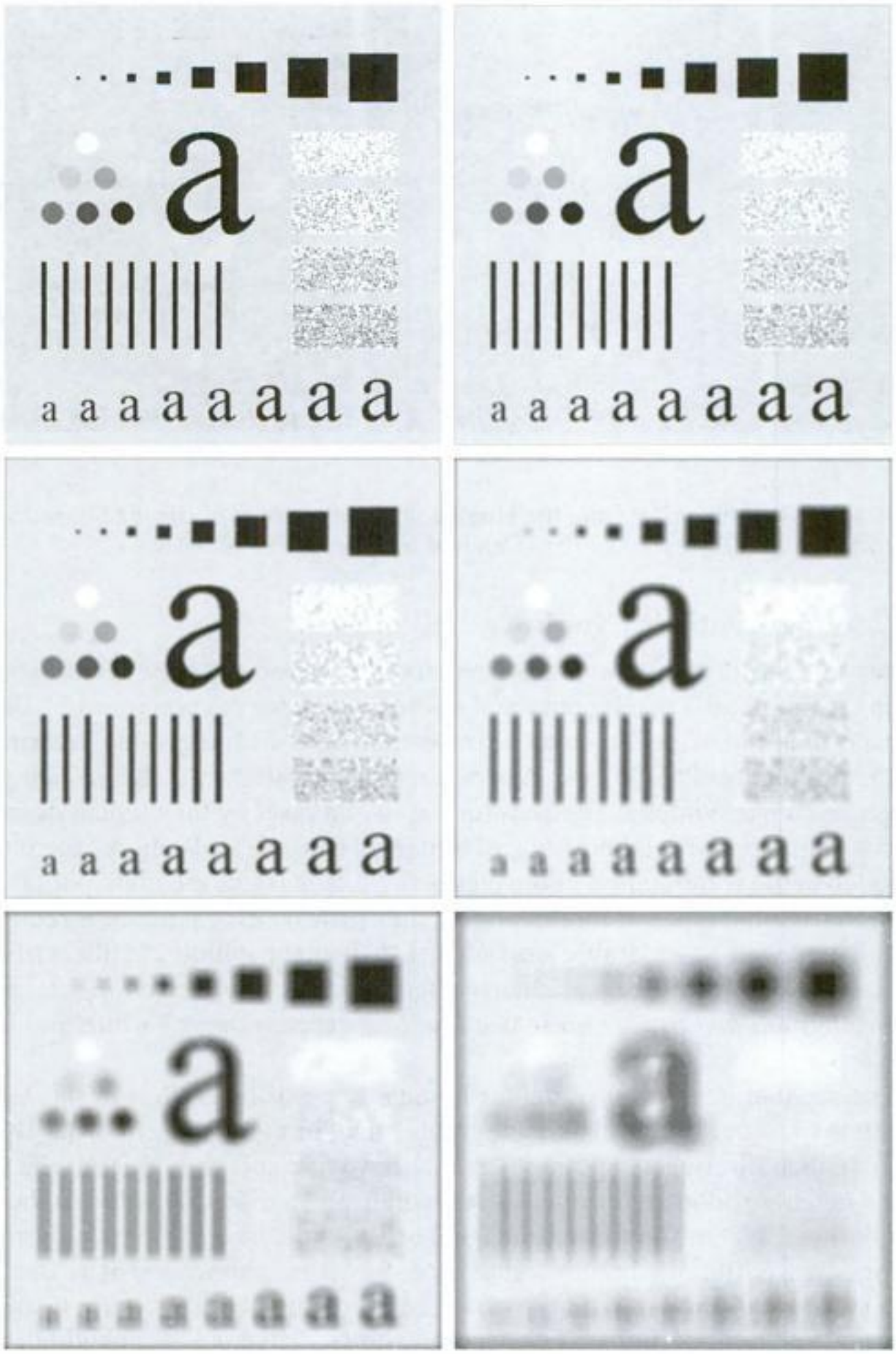
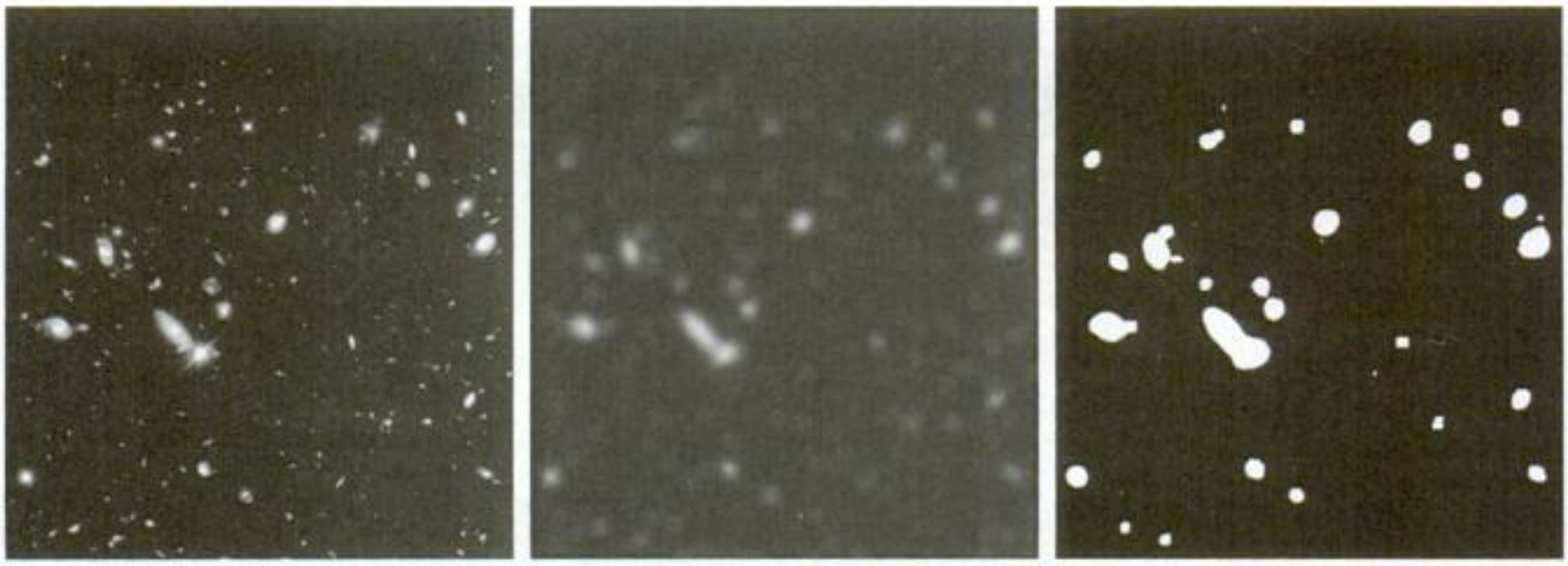


FIGURE 3.33 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $m = 3, 5, 9, 15$, and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their intensity levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

a b
c d
e f



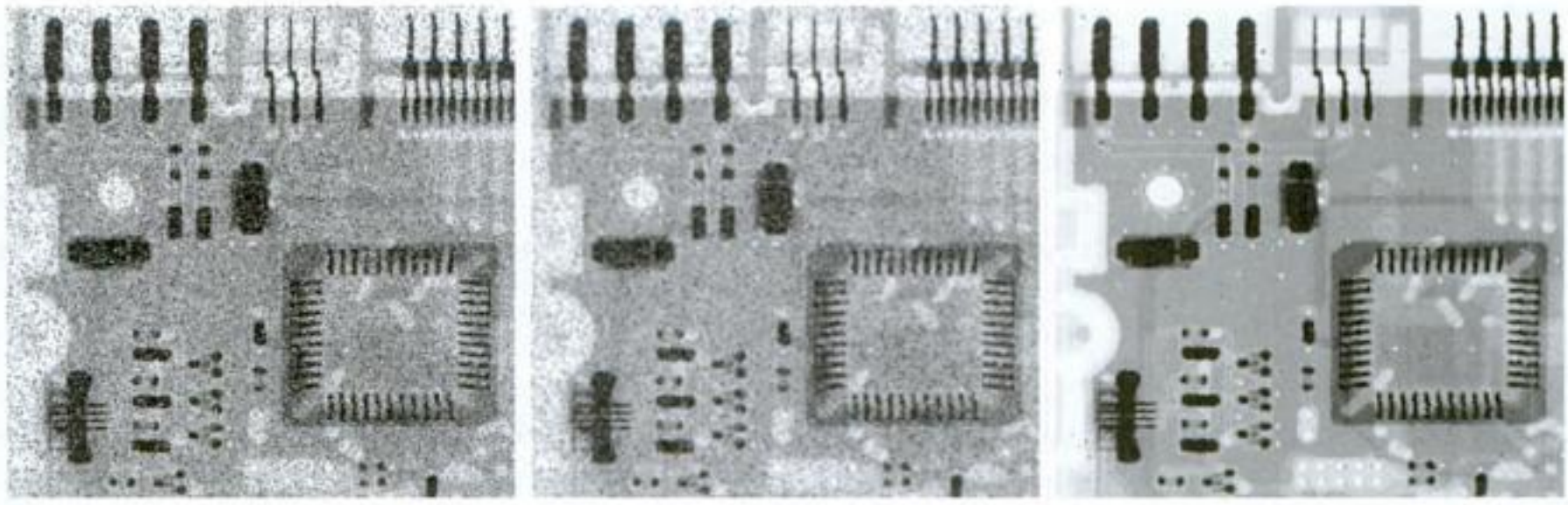
a b c

FIGURE 3.34 (a) Image of size 528 × 485 pixels from the Hubble Space Telescope. (b) Image filtered with a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

3.5.2 Order-Statistic (Nonlinear) Filters

Order-statistic filters are nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the center pixel with the value determined by the ranking result. The best-known filter in this category is the *median filter*, which, as its name implies, replaces the value of a pixel by the median of the intensity values in the neighborhood of that pixel (the original value of the pixel is included in the computation of the median). Median filters are quite popular because, for certain types of random noise, they provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filters of similar size. Median filters are particularly effective in the presence of *impulse noise*, also called *salt-and-pepper noise* because of its appearance as white and black dots superimposed on an image.

The median, ξ, of a set of values is such that half the values in the set are less than or equal to ξ, and half are greater than or equal to ξ. In order to perform median filtering at a point in an image, we first sort the values of the pixel in the neighborhood, determine their median, and assign that value to the corresponding pixel in the filtered image. For example, in a 3 × 3 neighborhood the median is the 5th largest value, in a 5 × 5 neighborhood it is the 13th largest value, and so on. When several values in a neighborhood are the same, all equal values are grouped. For example, suppose that a 3 × 3 neighborhood has values (10, 20, 20, 20, 15, 20, 20, 25, 100). These values are sorted as (10, 15, 20, 20, 20, 20, 20, 25, 100), which results in a median of 20. Thus, the principal function of median filters is to force points with distinct intensity levels to be more like their neighbors. In fact, isolated clusters of pixels that are light or dark with respect to their neighbors, and whose area is less than $m^2/2$ (one-half the filter area), are eliminated by an $m \times m$ median filter. In this case "eliminated" means forced to the median intensity of the neighbors. Larger clusters are affected considerably less.



a b c

FIGURE 3.35 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

Although the median filter is by far the most useful order-statistic filter in image processing, it is by no means the only one. The median represents the 50th percentile of a ranked set of numbers, but recall from basic statistics that ranking lends itself to many other possibilities. For example, using the 100th percentile results in the so-called *max filter*, which is useful for finding the brightest points in an image. The response of a 3×3 max filter is given by $R = \max\{z_k | k = 1, 2, \ldots, 9\}$. The 0th percentile filter is the *min filter*, used for the opposite purpose. Median, max, min, and several other nonlinear filters are considered in more detail in Section 5.3.

See Section 10.3.5 regarding percentiles.

■ Figure 3.35(a) shows an X-ray image of a circuit board heavily corrupted by salt-and-pepper noise. To illustrate the point about the superiority of median filtering over average filtering in situations such as this, we show in Fig. 3.35(b) the result of processing the noisy image with a 3×3 neighborhood averaging mask, and in Fig. 3.35(c) the result of using a 3×3 median filter. The averaging filter blurred the image and its noise reduction performance was poor. The superiority in all respects of median over average filtering in this case is quite evident. In general, median filtering is much better suited than averaging for the removal of salt-and-pepper noise. ■

EXAMPLE 3.14: Use of median filtering for noise reduction.

3.6 Sharpening Spatial Filters

The principal objective of sharpening is to highlight transitions in intensity. Uses of image sharpening vary and include applications ranging from electronic printing and medical imaging to industrial inspection and autonomous guidance in military systems. In the last section, we saw that image blurring could be accomplished in the spatial domain by pixel averaging in a neighborhood. Because averaging is analogous to integration, it is logical to conclude that sharpening can be accomplished by spatial differentiation. This, in fact, is the case,

and the discussion in this section deals with various ways of defining and implementing operators for sharpening by digital differentiation. Fundamentally, the strength of the response of a derivative operator is proportional to the degree of intensity discontinuity of the image at the point at which the operator is applied. Thus, image differentiation enhances edges and other discontinuities (such as noise) and deemphasizes areas with slowly varying intensities.

3.6.1 Foundation

In the two sections that follow, we consider in some detail sharpening filters that are based on first- and second-order derivatives, respectively. Before proceeding with that discussion, however, we stop to look at some of the fundamental properties of these derivatives in a digital context. To simplify the explanation, we focus attention initially on one-dimensional derivatives. In particular, we are interested in the behavior of these derivatives in areas of constant intensity, at the onset and end of discontinuities (step and ramp discontinuities), and along intensity ramps. As you will see in Chapter 10, these types of discontinuities can be used to model noise points, lines, and edges in an image. The behavior of derivatives during transitions into and out of these image features also is of interest.

The derivatives of a digital function are defined in terms of differences. There are various ways to define these differences. However, we require that any definition we use for a *first derivative* (1) must be zero in areas of constant intensity; (2) must be nonzero at the onset of an intensity step or ramp; and (3) must be nonzero along ramps. Similarly, any definition of a *second derivative* (1) must be zero in constant areas; (2) must be nonzero at the onset *and* end of an intensity step or ramp; and (3) must be zero along ramps of constant slope. Because we are dealing with digital quantities whose values are finite, the maximum possible intensity change also is finite, and the shortest distance over which that change can occur is between adjacent pixels.

A basic definition of the first-order derivative of a one-dimensional function $f(x)$ is the difference

$$\frac{\partial f}{\partial x} = f(x+1) - f(x) \tag{3.6-1}$$

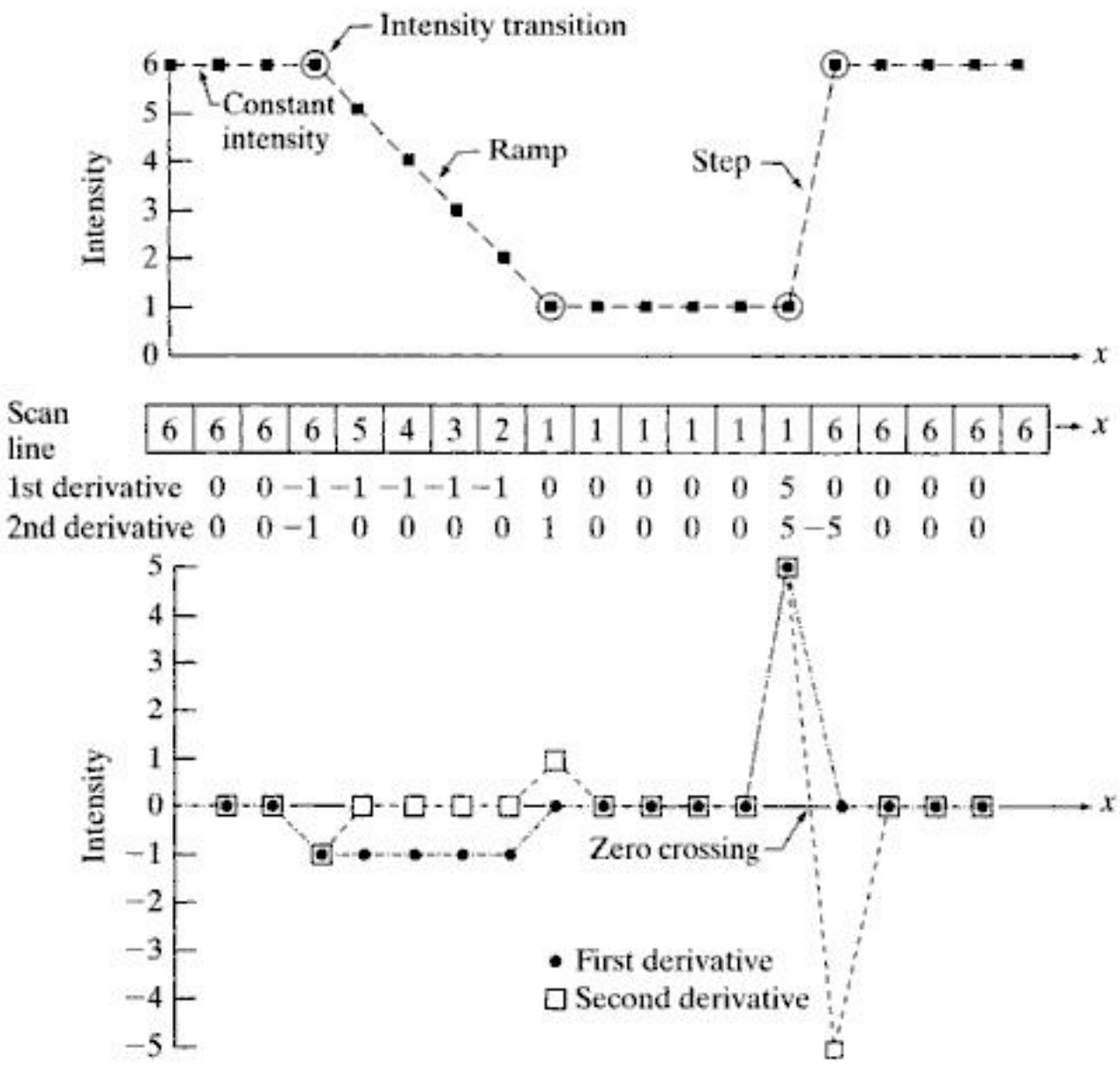
We return to Eq. (3.6-1) in Section 10.2.1 and show how it follows from a Taylor series expansion. For now, we accept it as a definition.

We used a partial derivative here in order to keep the notation the same as when we consider an image function of two variables, $f(x, y)$, at which time we will be dealing with partial derivatives along the two spatial axes. Use of a partial derivative in the present discussion does not affect in any way the nature of what we are trying to accomplish. Clearly, $\partial f/\partial x = df/dx$ when there is only one variable in the function; the same is true for the second derivative.

We define the second-order derivative of $f(x)$ as the difference

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x) \tag{3.6-2}$$

It is easily verified that these two definitions satisfy the conditions stated above. To illustrate this, and to examine the similarities and differences between



a
b
c

FIGURE 3.36
Illustration of the first and second derivatives of a 1-D digital function representing a section of a horizontal intensity profile from an image. In (a) and (c) data points are joined by dashed lines as a visualization aid.

first- and second-order derivatives of a digital function, consider the example in Fig. 3.36.

Figure 3.36(b) (center of the figure) shows a section of a scan line (intensity profile). The values inside the small squares are the intensity values in the scan line, which are plotted as black dots above it in Fig. 3.36(a). The dashed line connecting the dots is included to aid visualization. As the figure shows, the scan line contains an intensity ramp, three sections of constant intensity, and an intensity step. The circles indicate the onset or end of intensity transitions. The first- and second-order derivatives computed using the two preceding definitions are included below the scan line in Fig. 3.36(b), and are plotted in Fig. 3.36(c). When computing the first derivative at a location x, we subtract the value of the function at that location from the next point. So this is a "look-ahead" operation. Similarly, to compute the second derivative at x, we use the previous and the next points in the computation. To avoid a situation in which the previous or next points are outside the range of the scan line, we show derivative computations in Fig. 3.36 from the second through the penultimate points in the sequence.

Let us consider the properties of the first and second derivatives as we traverse the profile from left to right. First, we encounter an area of constant intensity and, as Figs. 3.36(b) and (c) show, both derivatives are zero there, so condition (1) is satisfied for both. Next, we encounter an intensity ramp followed by a step, and we note that the first-order derivative is nonzero at the onset of the ramp and

the step; similarly, the second derivative is nonzero at the onset *and* end of both the ramp and the step; therefore, property (2) is satisfied for both derivatives. Finally, we see that property (3) is satisfied also for both derivatives because the first derivative is nonzero and the second is zero along the ramp. Note that the sign of the second derivative changes at the onset and end of a step or ramp. In fact, we see in Fig. 3.36(c) that in a step transition a line joining these two values crosses the horizontal axis midway between the two extremes. This *zero crossing* property is quite useful for locating edges, as you will see in Chapter 10.

Edges in digital images often are ramp-like transitions in intensity, in which case the first derivative of the image would result in thick edges because the derivative is nonzero along a ramp. On the other hand, the second derivative would produce a double edge one pixel thick, separated by zeros. From this, we conclude that the second derivative enhances fine detail much better than the first derivative, a property that is ideally suited for sharpening images. Also, as you will learn later in this section, second derivatives are much easier to implement than first derivates, so we focus our attention initially on second derivatives.

3.6.2 Using the Second Derivative for Image Sharpening—The Laplacian

In this section we consider the implementation of 2-D, second-order derivatives and their use for image sharpening. We return to this derivative in Chapter 10, where we use it extensively for image segmentation. The approach basically consists of defining a discrete formulation of the second-order derivative and then constructing a filter mask based on that formulation. We are interested in *isotropic* filters, whose response is independent of the direction of the discontinuities in the image to which the filter is applied. In other words, isotropic filters are *rotation invariant*, in the sense that rotating the image and then applying the filter gives the same result as applying the filter to the image first and then rotating the result.

It can be shown (Rosenfeld and Kak [1982]) that the simplest isotropic derivative operator is the Laplacian, which, for a function (image) $f(x, y)$ of two variables, is defined as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \tag{3.6-3}$$

Because derivatives of any order are linear operations, the Laplacian is a linear operator. To express this equation in discrete form, we use the definition in Eq. (3.6-2), keeping in mind that we have to carry a second variable. In the x-direction, we have

$$\frac{\partial^2 f}{\partial x^2} = f(x + 1, y) + f(x - 1, y) - 2f(x, y) \tag{3.6-4}$$

and, similarly, in the y-direction we have

$$\frac{\partial^2 f}{\partial y^2} = f(x, y + 1) + f(x, y - 1) - 2f(x, y) \tag{3.6-5}$$

Therefore, it follows from the preceding three equations that the discrete Laplacian of two variables is

$$\nabla^2 f(x, y) = f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1) - 4f(x, y) \tag{3.6-6}$$

This equation can be implemented using the filter mask in Fig. 3.37(a), which gives an isotropic result for rotations in increments of 90°. The mechanics of implementation are as in Section 3.5.1 for linear smoothing filters. We simply are using different coefficients here.

The diagonal directions can be incorporated in the definition of the digital Laplacian by adding two more terms to Eq. (3.6-6), one for each of the two diagonal directions. The form of each new term is the same as either Eq. (3.6-4) or (3.6-5), but the coordinates are along the diagonals. Because each diagonal term also contains a $-2f(x, y)$ term, the total subtracted from the difference terms now would be $-8f(x, y)$. Figure 3.37(b) shows the filter mask used to implement this new definition. This mask yields isotropic results in increments of 45°. You are likely to see in practice the Laplacian masks in Figs. 3.37(c) and (d). They are obtained from definitions of the second derivatives that are the negatives of the ones we used in Eqs. (3.6-4) and (3.6-5). As such, they yield equivalent results, but the difference in sign must be kept in mind when combining (by addition or subtraction) a Laplacian-filtered image with another image.

Because the Laplacian is a derivative operator, its use highlights intensity discontinuities in an image and deemphasizes regions with slowly varying intensity levels. This will tend to produce images that have grayish edge lines and other discontinuities, all superimposed on a dark, featureless background. Background features can be "recovered" while still preserving the sharpening

0	1	0	1	1	1
1	−4	1	1	−8	1
0	1	0	1	1	1
0	−1	0	−1	−1	−1
−1	4	−1	−1	8	−1
0	−1	0	−1	−1	−1

a b
c d

FIGURE 3.37
(a) Filter mask used to implement Eq. (3.6-6).
(b) Mask used to implement an extension of this equation that includes the diagonal terms.
(c) and (d) Two other implementations of the Laplacian found frequently in practice.

effect of the Laplacian simply by adding the Laplacian image to the original. As noted in the previous paragraph, it is important to keep in mind which definition of the Laplacian is used. If the definition used has a negative center coefficient, then we *subtract*, rather than add, the Laplacian image to obtain a sharpened result. Thus, the basic way in which we use the Laplacian for image sharpening is

$$g(x, y) = f(x, y) + c\left[\nabla^2 f(x, y)\right] \tag{3.6-7}$$

where $f(x, y)$ and $g(x, y)$ are the input and sharpened images, respectively. The constant is $c = -1$ if the Laplacian filters in Fig. 3.37(a) or (b) are used, and $c = 1$ if either of the other two filters is used.

EXAMPLE 3.15: Image sharpening using the Laplacian.

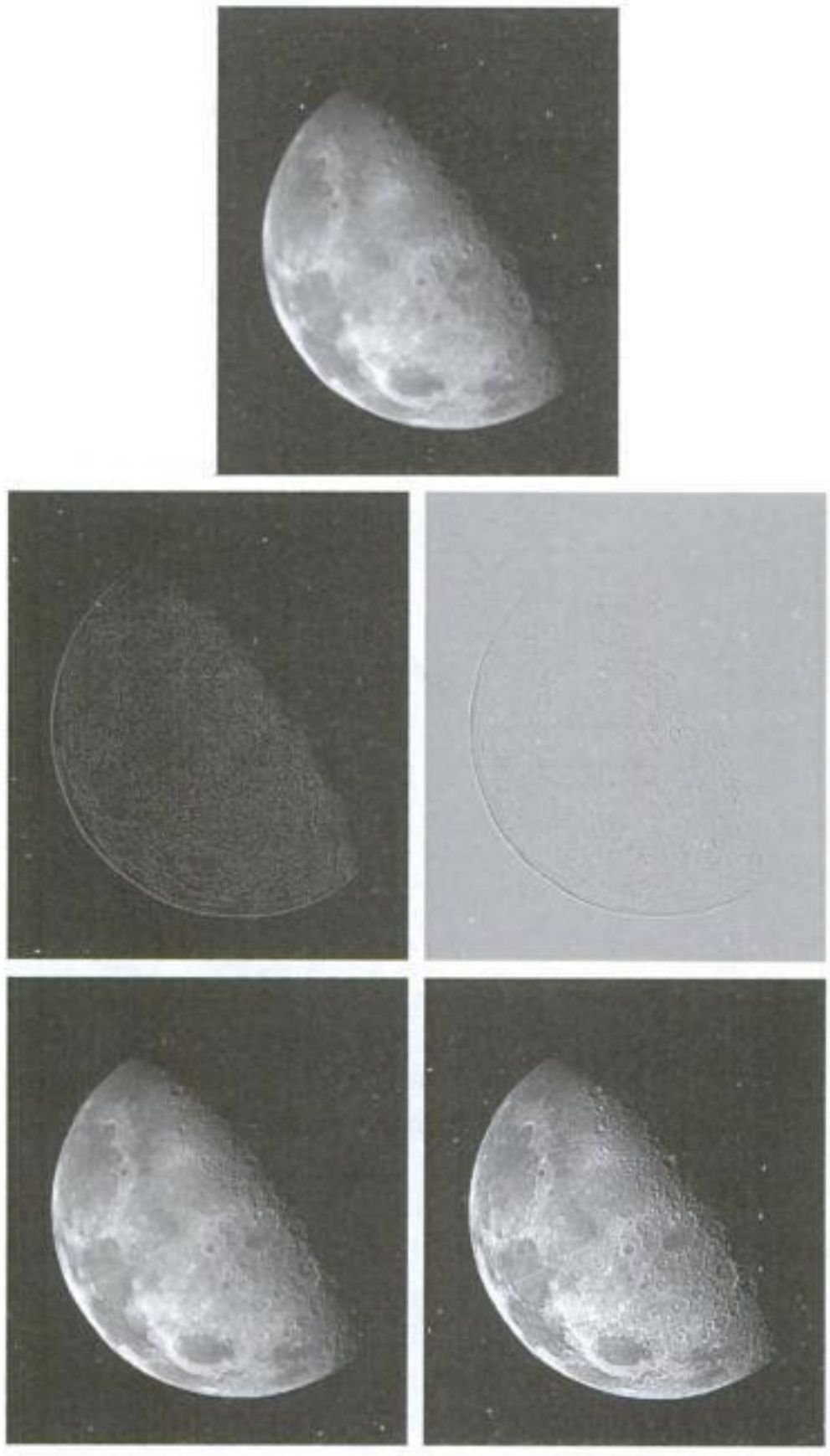
■ Figure 3.38(a) shows a slightly blurred image of the North Pole of the moon. Figure 3.38(b) shows the result of filtering this image with the Laplacian mask in Fig. 3.37(a). Large sections of this image are black because the Laplacian contains both positive and negative values, and all negative values are clipped at 0 by the display.

A typical way to scale a Laplacian image is to add to it its minimum value to bring the new minimum to zero and then scale the result to the full $[0, L - 1]$ intensity range, as explained in Eqs. (2.6-10) and (2.6-11). The image in Fig. 3.38(c) was scaled in this manner. Note that the dominant features of the image are edges and sharp intensity discontinuities. The background, previously black, is now gray due to scaling. This grayish appearance is typical of Laplacian images that have been scaled properly. Figure 3.38(d) shows the result obtained using Eq. (3.6-7) with $c = -1$. The detail in this image is unmistakably clearer and sharper than in the original image. Adding the original image to the Laplacian restored the overall intensity variations in the image, with the Laplacian increasing the contrast at the locations of intensity discontinuities. The net result is an image in which small details were enhanced and the background tonality was reasonably preserved. Finally, Fig. 3.38(e) shows the result of repeating the preceding procedure with the filter in Fig. 3.37(b). Here, we note a significant improvement in sharpness over Fig. 3.38(d). This is not unexpected because using the filter in Fig. 3.37(b) provides additional differentiation (sharpening) in the diagonal directions. Results such as those in Figs. 3.38(d) and (e) have made the Laplacian a tool of choice for sharpening digital images. ■

3.6.3 Unsharp Masking and Highboost Filtering

A process that has been used for many years by the printing and publishing industry to sharpen images consists of subtracting an unsharp (smoothed) version of an image from the original image. This process, called *unsharp masking*, consists of the following steps:

1. Blur the original image.
2. Subtract the blurred image from the original (the resulting difference is called the *mask*.)
3. Add the mask to the original.



a
b c
d e

FIGURE 3.38
(a) Blurred image of the North Pole of the moon. (b) Laplacian without scaling. (c) Laplacian with scaling. (d) Image sharpened using the mask in Fig. 3.37(a). (e) Result of using the mask in Fig. 3.37(b). (Original image courtesy of NASA.)

Letting $\bar{f}(x, y)$ denote the blurred image, unsharp masking is expressed in equation form as follows. First we obtain the mask:

$$g_{\text{mask}}(x, y) = f(x, y) - \bar{f}(x, y) \tag{3.6-8}$$

Then we add a weighted portion of the mask back to the original image:

$$g(x, y) = f(x, y) + k * g_{\text{mask}}(x, y) \tag{3.6-9}$$

where we included a weight, k ($k \geq 0$), for generality. When $k = 1$, we have unsharp masking, as defined above. When $k > 1$, the process is referred to as

a
b
c
d

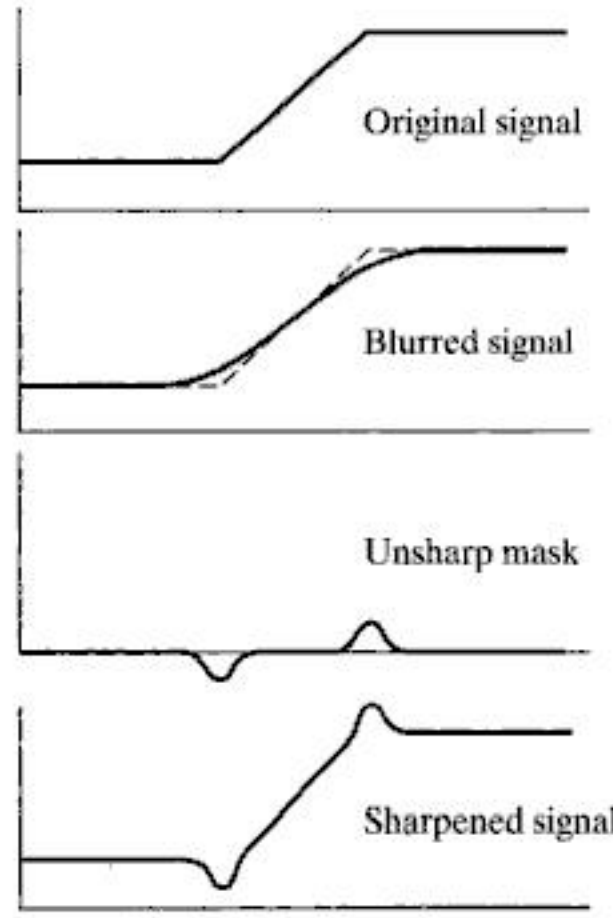


FIGURE 3.39 1-D illustration of the mechanics of unsharp masking. (a) Original signal. (b) Blurred signal with original shown dashed for reference. (c) Unsharp mask. (d) Sharpened signal, obtained by adding (c) to (a).

highboost filtering. Choosing $k < 1$ de-emphasizes the contribution of the unsharp mask.

Figure 3.39 explains how unsharp masking works. The intensity profile in Fig. 3.39(a) can be interpreted as a horizontal scan line through a vertical edge that transitions from a dark to a light region in an image. Figure 3.39(b) shows the result of smoothing, superimposed on the original signal (shown dashed) for reference. Figure 3.39(c) is the unsharp mask, obtained by subtracting the blurred signal from the original. By comparing this result with the section of Fig. 3.36(c) corresponding to the ramp in Fig. 3.36(a), we note that the unsharp mask in Fig. 3.39(c) is very similar to what we would obtain using a second-order derivative. Figure 3.39(d) is the final sharpened result, obtained by adding the mask to the original signal. The points at which a change of slope in the intensity occurs in the signal are now emphasized (sharpened). Observe that negative values were added to the original. Thus, it is possible for the final result to have negative intensities if the original image has any zero values or if the value of k is chosen large enough to emphasize the peaks of the mask to a level larger than the minimum value in the original. Negative values would cause a dark halo around edges, which, if k is large enough, can produce objectionable results.

EXAMPLE 3.16: Image sharpening using unsharp masking.

■ Figure 3.40(a) shows a slightly blurred image of white text on a dark gray background. Figure 3.40(b) was obtained using a Gaussian smoothing filter (see Section 3.4.4) of size 5×5 with $\sigma = 3$. Figure 3.40(c) is the unsharp mask, obtained using Eq. (3.6-8). Figure 3.40(d) was obtained using unsharp

a
b
c
d
e

FIGURE 3.40
(a) Original image. (b) Result of blurring with a Gaussian filter. (c) Unsharp mask. (d) Result of using unsharp masking. (e) Result of using highboost filtering.

masking [Eq. (3.6-9) with $k = 1$]. This image is a slight improvement over the original, but we can do better. Figure 3.40(e) shows the result of using Eq. (3.6-9) with $k = 4.5$, the largest possible value we could use and still keep positive all the values in the final result. The improvement in this image over the original is significant. ■

3.6.4 Using First-Order Derivatives for (Nonlinear) Image Sharpening—The Gradient

First derivatives in image processing are implemented using the magnitude of the gradient. For a function $f(x, y)$, the gradient of f at coordinates (x, y) is defined as the two-dimensional column *vector*

$$\nabla f \equiv \text{grad}(f) \equiv \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{bmatrix} \tag{3.6-10}$$

We discuss the gradient in detail in Section 10.2.5. Here, we are interested only in using the magnitude of the gradient for image sharpening.

This vector has the important geometrical property that it points in the direction of the greatest rate of change of f at location (x, y).

The *magnitude* (*length*) of vector ∇f, denoted as $M(x, y)$, where

$$M(x, y) = \text{mag}(\nabla f) = \sqrt{g_x^2 + g_y^2} \tag{3.6-11}$$

is the *value* at (x, y) of the rate of change in the direction of the gradient vector. Note that $M(x, y)$ is an image of the same size as the original, created when x and y are allowed to vary over all pixel locations in f. It is common practice to refer to this image as the *gradient image* (or simply as the *gradient* when the meaning is clear).

Because the components of the gradient vector are derivatives, they are linear operators. However, the magnitude of this vector is not because of the squaring and square root operations. On the other hand, the partial derivatives in Eq. (3.6-10) are not rotation invariant (isotropic), but the magnitude of the gradient vector is. In some implementations, it is more suitable computationally to approximate the squares and square root operations by absolute values:

$$M(x, y) \approx |g_x| + |g_y| \tag{3.6-12}$$

This expression still preserves the relative changes in intensity, but the isotropic property is lost in general. However, as in the case of the Laplacian, the isotropic properties of the discrete gradient defined in the following paragraph are preserved only for a limited number of rotational increments that depend on the filter masks used to approximate the derivatives. As it turns out, the most popular masks used to approximate the gradient are isotropic at multiples of 90°. These results are independent of whether we use Eq. (3.6-11) or (3.6-12), so nothing of significance is lost in using the latter equation if we choose to do so.

As in the case of the Laplacian, we now define discrete approximations to the preceding equations and from there formulate the appropriate filter masks. In order to simplify the discussion that follows, we will use the notation in Fig. 3.41(a) to denote the intensities of image points in a 3×3 region. For

a
b c
d e

FIGURE 3.41
A 3×3 region of an image (the zs are intensity values).
(b)–(c) Roberts cross gradient operators.
(d)–(e) Sobel operators. All the mask coefficients sum to zero, as expected of a derivative operator.

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

−1	0
0	1

0	−1
1	0

−1	−2	−1
0	0	0
1	2	1

−1	0	1
−2	0	2
−1	0	1

example, the center point, z_5, denotes $f(x, y)$ at an arbitrary location, (x, y); z_1 denotes $f(x - 1, y - 1)$; and so on, using the notation introduced in Fig. 3.28. As indicated in Section 3.6.1, the simplest approximations to a first-order derivative that satisfy the conditions stated in that section are $g_x = (z_8 - z_5)$ and $g_y = (z_6 - z_5)$. Two other definitions proposed by Roberts [1965] in the early development of digital image processing use cross differences:

$$g_x = (z_9 - z_5) \quad \text{and} \quad g_y = (z_8 - z_6) \tag{3.6-13}$$

If we use Eqs. (3.6-11) and (3.6-13), we compute the gradient image as

$$M(x, y) = \left[(z_9 - z_5)^2 + (z_8 - z_6)^2\right]^{1/2} \tag{3.6-14}$$

If we use Eqs. (3.6-12) and (3.6-13), then

$$M(x, y) \approx |z_9 - z_5| + |z_8 - z_6| \tag{3.6-15}$$

where it is understood that x and y vary over the dimensions of the image in the manner described earlier. The partial derivative terms needed in equation (3.6-13) can be implemented using the two linear filter masks in Figs. 3.41(b) and (c). These masks are referred to as the *Roberts cross-gradient operators*.

Masks of even sizes are awkward to implement because they do not have a center of symmetry. The smallest filter masks in which we are interested are of size 3×3. Approximations to g_x and g_y using a 3×3 neighborhood centered on z_5 are as follows:

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \tag{3.6-16}$$

and

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \tag{3.6-17}$$

These equations can be implemented using the masks in Figs. 3.41(d) and (e). The difference between the third and first rows of the 3×3 image region implemented by the mask in Fig. 3.41(d) approximates the partial derivative in the x-direction, and the difference between the third and first columns in the other mask approximates the derivative in the y-direction. After computing the partial derivatives with these masks, we obtain the magnitude of the gradient as before. For example, substituting g_x and g_y into Eq. (3.6-12) yields

$$\begin{aligned} M(x, y) \approx\ & |(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)| \\ & + |(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)| \end{aligned} \tag{3.6-18}$$

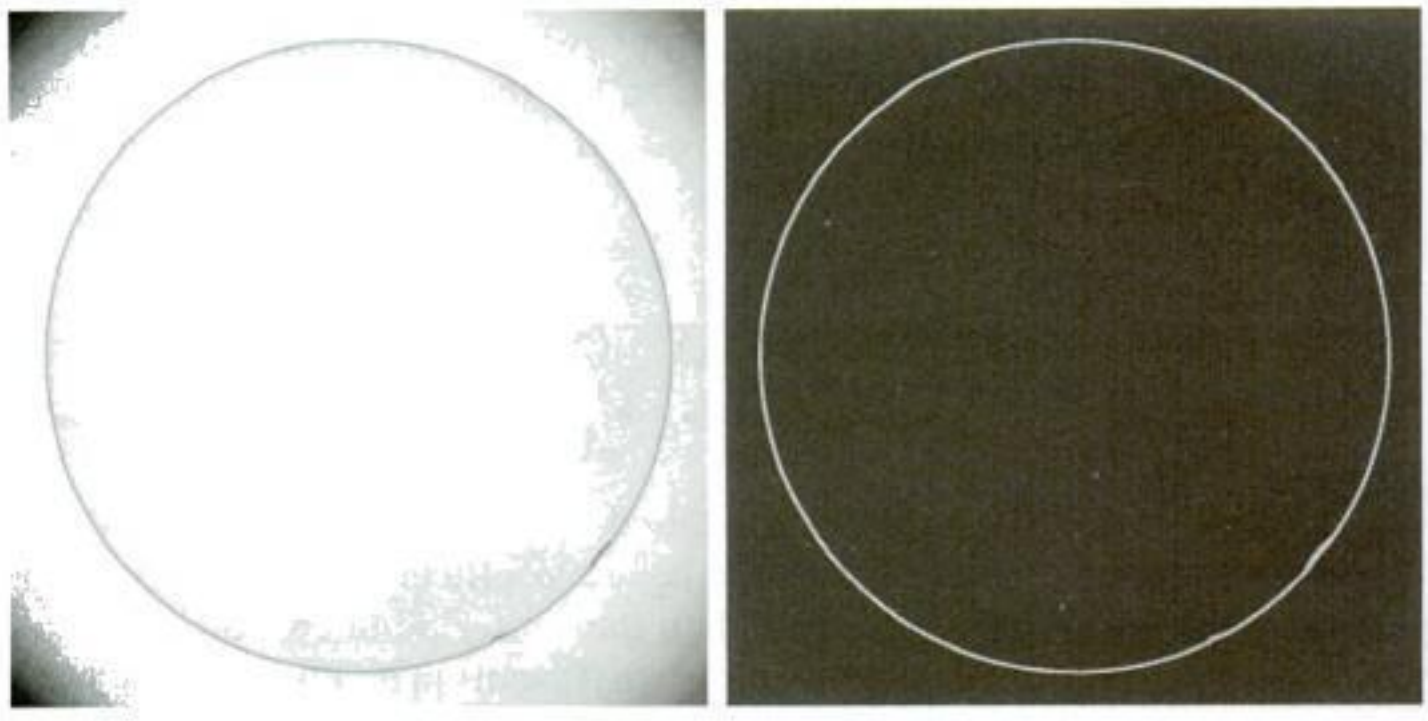
The masks in Figs. 3.41(d) and (e) are called the *Sobel operators*. The idea behind using a weight value of 2 in the center coefficient is to achieve some smoothing by giving more importance to the center point (we discuss this in more detail in Chapter 10). Note that the coefficients in all the masks shown in Fig. 3.41 sum to 0, indicating that they would give a response of 0 in an area of constant intensity, as is expected of a derivative operator.

As mentioned earlier, the computations of g_x and g_y are linear operations because they involve derivatives and, therefore, can be implemented as a sum of products using the spatial masks in Fig. 3.41. The nonlinear aspect of sharpening with the gradient is the computation of $M(x, y)$ involving squaring and square roots, or the use of absolute values, all of which are nonlinear operations. These operations are performed *after* the linear process that yields g_x and g_y.

EXAMPLE 3.17: Use of the gradient for edge enhancement.

■ The gradient is used frequently in industrial inspection, either to aid humans in the detection of defects or, what is more common, as a preprocessing step in automated inspection. We will have more to say about this in Chapters 10 and 11. However, it will be instructive at this point to consider a simple example to show how the gradient can be used to enhance defects and eliminate slowly changing background features. In this example, enhancement is used as a preprocessing step for automated inspection, rather than for human analysis.

Figure 3.42(a) shows an optical image of a contact lens, illuminated by a lighting arrangement designed to highlight imperfections, such as the two edge defects in the lens boundary seen at 4 and 5 o'clock. Figure 3.42(b) shows the gradient obtained using Eq. (3.6-12) with the two Sobel masks in Figs. 3.41(d) and (e). The edge defects also are quite visible in this image, but with the added advantage that constant or slowly varying shades of gray have been eliminated, thus simplifying considerably the computational task required for automated inspection. The gradient can be used also to highlight small specs that may not be readily visible in a gray-scale image (specs like these can be foreign matter, air pockets in a supporting solution, or miniscule imperfections in the lens). The ability to enhance small discontinuities in an otherwise flat gray field is another important feature of the gradient. ■



a b

FIGURE 3.42
(a) Optical image of contact lens (note defects on the boundary at 4 and 5 o'clock). (b) Sobel gradient. (Original image courtesy of Pete Sites, Perceptics Corporation.)

3.7 Combining Spatial Enhancement Methods

With a few exceptions, like combining blurring with thresholding (Fig. 3.34), we have focused attention thus far on individual approaches. Frequently, a given task will require application of several complementary techniques in order to achieve an acceptable result. In this section we illustrate by means of an example how to combine several of the approaches developed thus far in this chapter to address a difficult image enhancement task.

The image in Fig. 3.43(a) is a nuclear whole body bone scan, used to detect diseases such as bone infection and tumors. Our objective is to enhance this image by sharpening it and by bringing out more of the skeletal detail. The narrow dynamic range of the intensity levels and high noise content make this image difficult to enhance. The strategy we will follow is to utilize the Laplacian to highlight fine detail, and the gradient to enhance prominent edges. For reasons that will be explained shortly, a smoothed version of the gradient image will be used to mask the Laplacian image (see Fig. 2.30 regarding masking). Finally, we will attempt to increase the dynamic range of the intensity levels by using an intensity transformation.

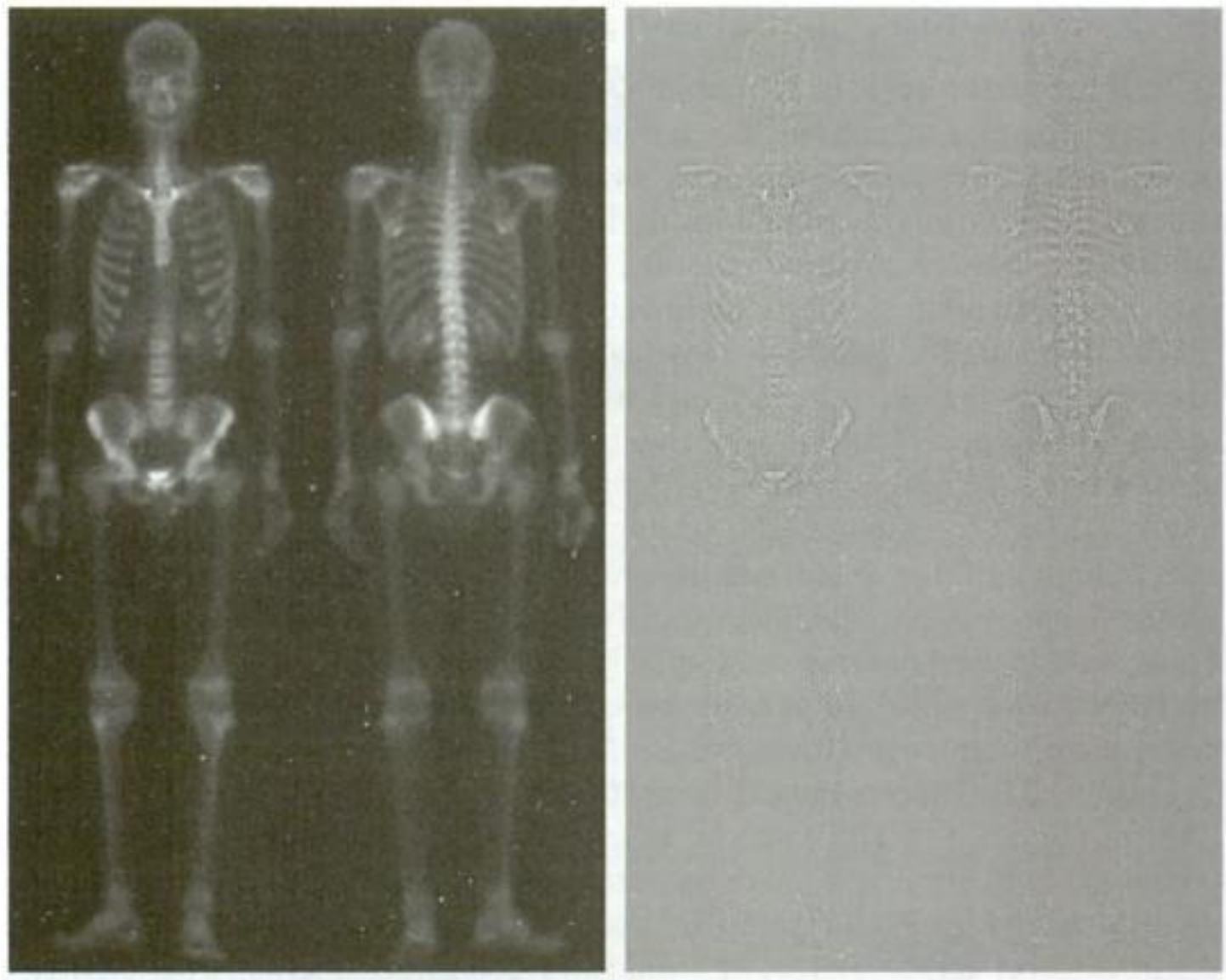
Figure 3.43(b) shows the Laplacian of the original image, obtained using the filter in Fig. 3.37(d). This image was scaled (for display only) using the same technique as in Fig. 3.38(c). We can obtain a sharpened image at this point simply by adding Figs. 3.43(a) and (b), according to Eq. (3.6-7). Just by looking at the noise level in Fig. 3.43(b), we would expect a rather noisy sharpened image if we added Figs. 3.43(a) and (b), a fact that is confirmed by the result in Fig. 3.43(c). One way that comes immediately to mind to reduce the noise is to use a median filter. However, median filtering is a nonlinear process capable of removing image features. This is unacceptable in medical image processing.

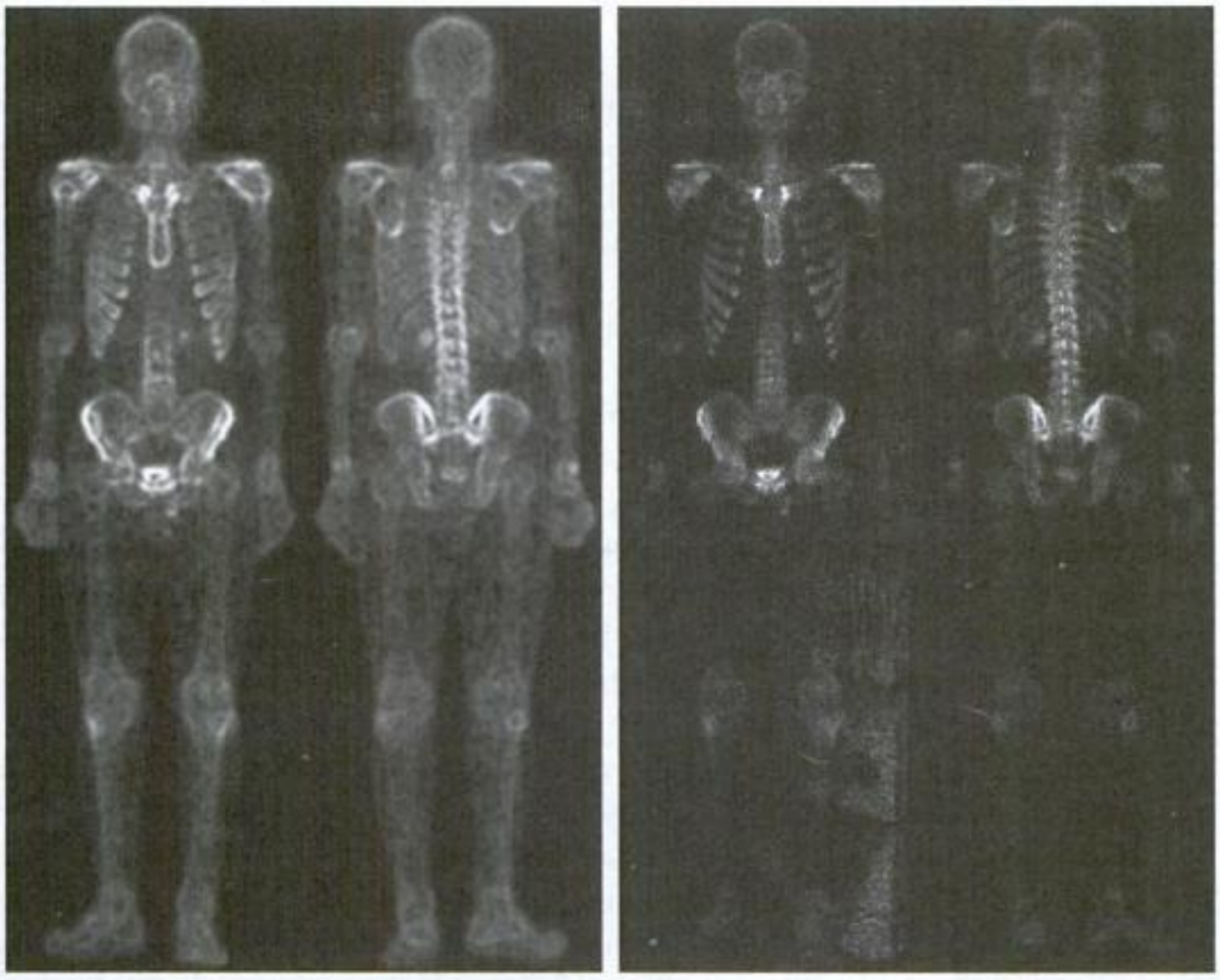
An alternate approach is to use a mask formed from a smoothed version of the gradient of the original image. The motivation behind this is straightforward and is based on the properties of first- and second-order derivatives explained in Section 3.6.1. The Laplacian, being a second-order derivative operator, has the definite advantage that it is superior in enhancing fine detail. However, this causes it to produce noisier results than the gradient. This noise is most objectionable in smooth areas, where it tends to be more visible. The gradient has a stronger average response in areas of significant intensity transitions (ramps and steps) than does the Laplacian. The response of the gradient to noise and fine detail is lower than the Laplacian's and can be lowered further by smoothing the gradient with an averaging filter. The idea, then, is to smooth the gradient and multiply it by the Laplacian image. In this context, we may view the smoothed gradient as a mask image. The product will preserve details in the strong areas while reducing noise in the relatively flat areas. This process can be interpreted roughly as combining the best features of the Laplacian and the gradient. The result is added to the original to obtain a final sharpened image.

Figure 3.43(d) shows the Sobel gradient of the original image, computed using Eq. (3.6-12). Components g_x and g_y were obtained using the masks in Figs. 3.41(d) and (e), respectively. As expected, edges are much more dominant

a b
c d

FIGURE 3.43
(a) Image of whole body bone scan.
(b) Laplacian of (a). (c) Sharpened image obtained by adding (a) and (b). (d) Sobel gradient of (a).





e f
g h

FIGURE 3.43
(Continued)
(e) Sobel image smoothed with a 5×5 averaging filter. (f) Mask image formed by the product of (c) and (e). (g) Sharpened image obtained by the sum of (a) and (f). (h) Final result obtained by applying a power-law transformation to (g). Compare (g) and (h) with (a). (Original image courtesy of G.E. Medical Systems.)

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

in surveillance, and in a host of other areas where the principal objective of enhancement is to obtain an image with a higher content of visual detail.

3.8 Using Fuzzy Techniques for Intensity Transformations and Spatial Filtering

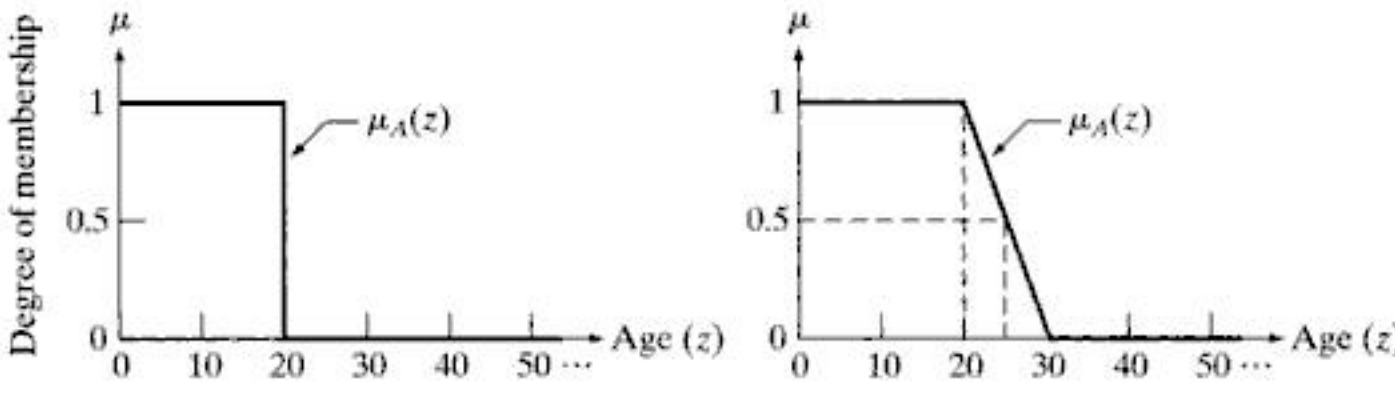
We conclude this chapter with an introduction to fuzzy sets and their application to intensity transformations and spatial filtering, which are the main topics of discussion in the preceding sections. As it turns out, these two applications are among the most frequent areas in which fuzzy techniques for image processing are applied. The references at the end of this chapter provide an entry point to the literature on fuzzy sets and to other applications of fuzzy techniques in image processing. As you will see in the following discussion, fuzzy sets provide a framework for incorporating human knowledge in the solution of problems whose formulation is based on imprecise concepts.

3.8.1 Introduction

As noted in Section 2.6.4, a *set* is a collection of objects (elements) and *set theory* is the set of tools that deals with operations on and among sets. Set theory, along with mathematical logic, is one of the axiomatic foundations of classical mathematics. Central to set theory is the notion of set membership. We are used to dealing with so-called "crisp" sets, whose membership only can be true or false in the traditional sense of bi-valued Boolean logic, with 1 typically indicating true and 0 indicating false. For example, let Z denote the set of all people, and suppose that we want to define a subset, A, of Z, called the "set of young people." In order to form this subset, we need to define a *membership function* that assigns a value of 1 or 0 to every element, z, of Z. Because we are dealing with a bi-valued logic, the membership function simply defines a threshold at or below which a person is considered young, and above which a person is considered not young. Figure 3.44(a) summarizes this concept using an age threshold of 20 years and letting $\mu_A(z)$ denote the membership function just discussed.

Membership functions also are called *characteristic functions*.

We see an immediate difficulty with this formulation: A person 20 years of age is considered young, but a person whose age is 20 years and 1 second is not a member of the set of young people. This is a fundamental problem with crisp sets that limits the use of classical set theory in many practical applications.



a b

FIGURE 3.44 Membership functions used to generate (a) a crisp set, and (b) a fuzzy set.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

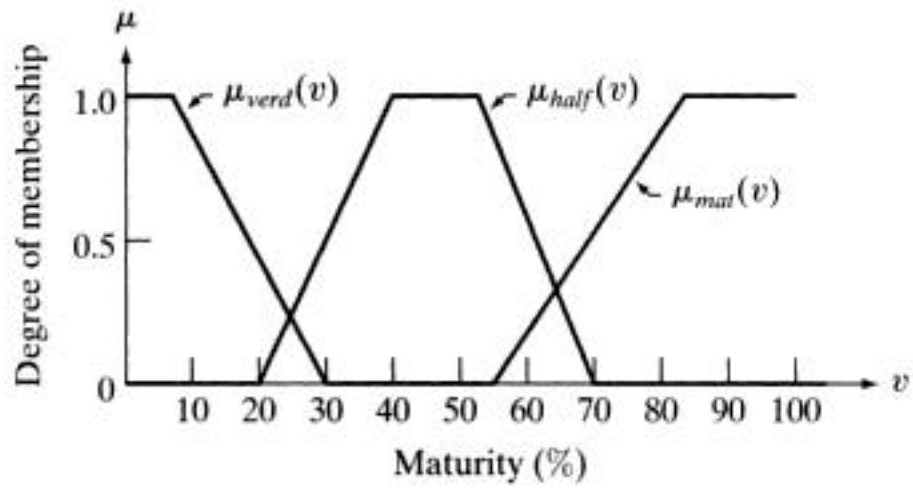


FIGURE 3.48 Membership functions characterizing the outputs *verdant*, *half-mature*, and *mature*.

shows the membership functions of the fuzzy outputs we are going to use in this example. Note that the independent variable of the outputs is *maturity*, which is different from the independent variable of the inputs.

Figures 3.47 and 3.48, together with the rule base, contain all the information required to relate inputs and outputs. For example, we note that the expression *red* AND *mature* is nothing more than the intersection (AND) operation defined earlier. In the present case, the independent variables of the membership functions of inputs and outputs are different, so the result will be two-dimensional. For instance, Figs. 3.49(a) and (b) show the membership functions of *red* and *mature*, and Fig. 3.49(c) shows how they relate in two dimensions. To find the result of the AND operation between these two functions, recall from Eq. (3.8-5) that AND is defined as the minimum of the two membership functions; that is,

$$\mu_3(z, v) = \min\{\mu_{red}(z), \mu_{mat}(v)\} \tag{3.8-12}$$

a b
c d

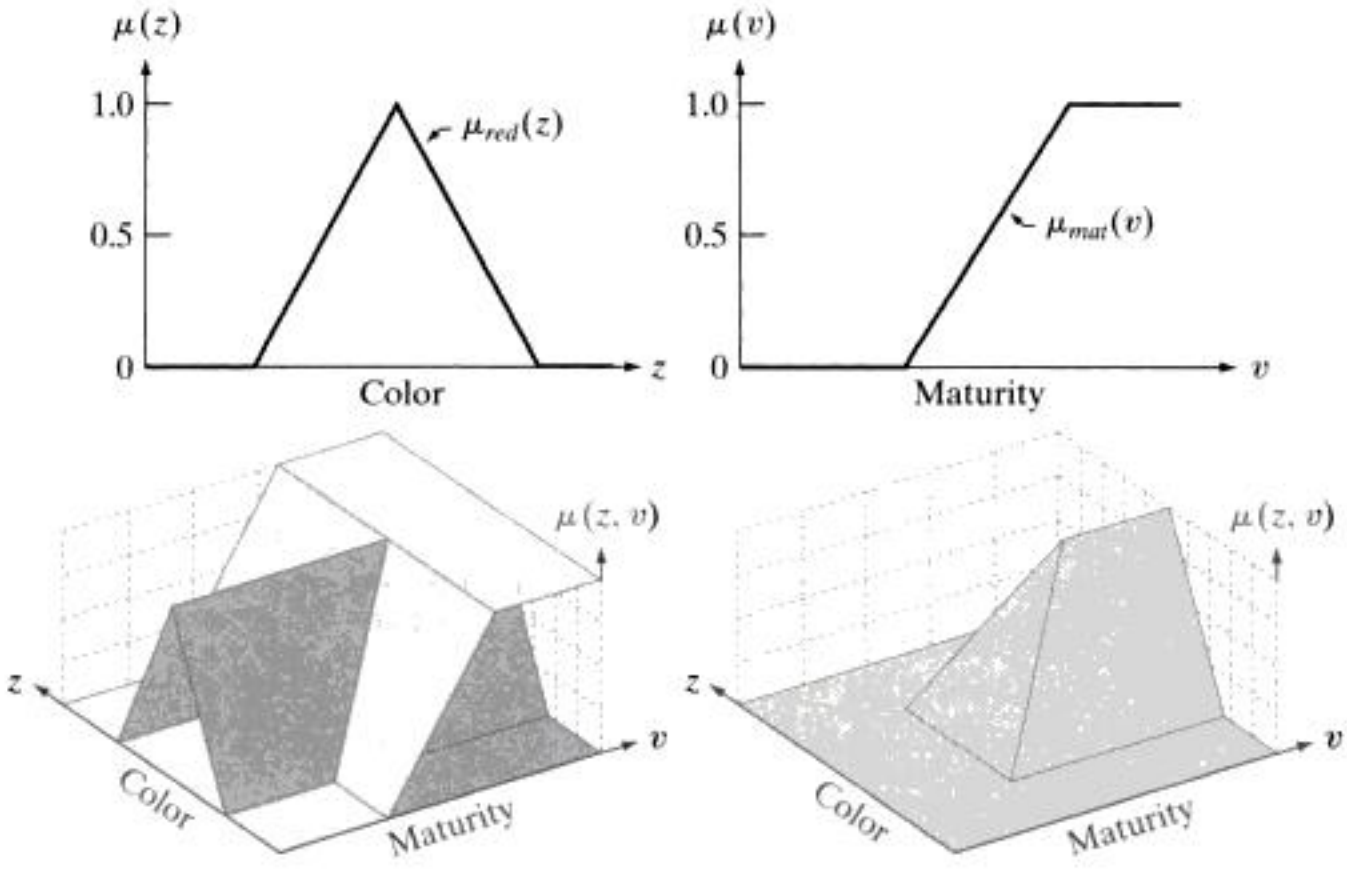


FIGURE 3.49 (a) Shape of the membership function associated with the color red, and (b) corresponding output membership function. These two functions are associated by rule R_3. (c) Combined representation of the two functions. The representation is 2-D because the independent variables in (a) and (b) are different. (d) The AND of (a) and (b), as defined in Eq. (3.8-5).

where 3 in the subscript denotes that this is the result of rule R_3 in the knowledge base. Figure 3.49(d) shows the result of the AND operation.†

Equation (3.8-12) is a general result involving two membership functions. In practice, we are interested in the output resulting from a *specific* input. Let z_0 denote a specific value of *red*. The degree of membership of the red color component in response to this input is simply a scalar value, $\mu_{red}(z_0)$. We find the output corresponding to rule R_3 and this specific input by performing the AND operation between $\mu_{red}(z_0)$ and the general result, $\mu_3(z, v)$, evaluated also at z_0. As noted before, the AND operation is implemented using the minimum operation:

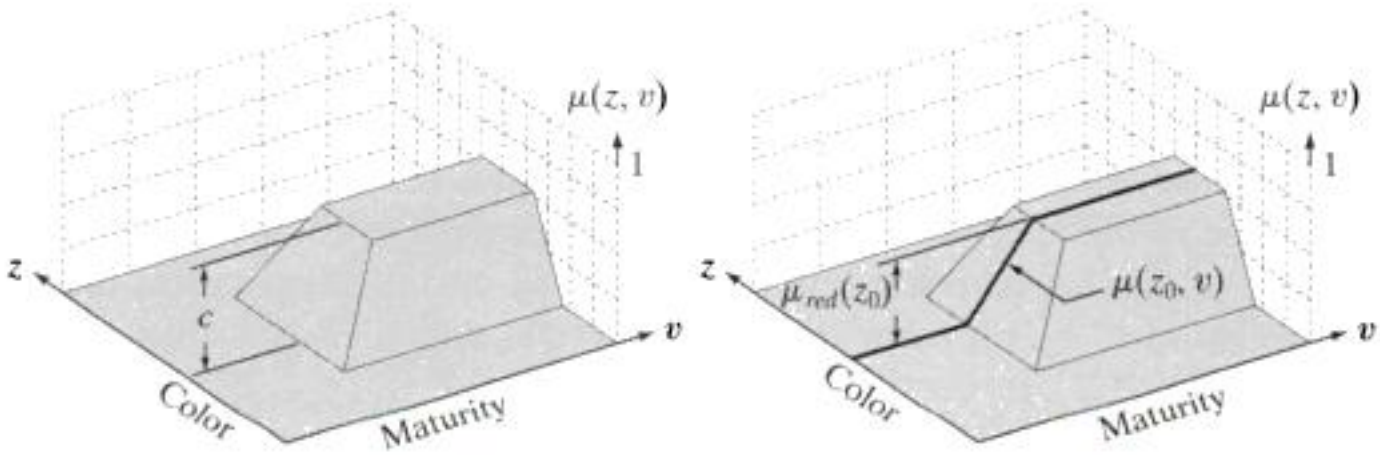
$$Q_3(v) = \min\{\mu_{red}(z_0), \mu_3(z_0, v)\} \tag{3.8-13}$$

where $Q_3(v)$ denotes the fuzzy output due to rule R_3 and a specific input. The only variable in Q_3 is the output variable, v, as expected.

To interpret Eq. (3.8-13) graphically, consider Fig. 3.49(d) again, which shows the general function $\mu_3(z, v)$. Performing the minimum operation of a positive constant, c, and this function would clip all values of $\mu_3(z, v)$ above that constant, as Fig. 3.50(a) shows. However, we are interested only in one value (z_0) along the color axis, so the relevant result is a cross section of the truncated function along the maturity axis, with the cross section placed at z_0, as Fig. 3.50(b) shows [because Fig. 3.50(a) corresponds to rule R_3, it follows that $c = \mu_{red}(z_0)$]. Equation (3.8-13) is the expression for this cross section.

Using the same line of reasoning, we obtain the fuzzy responses due to the other two rules and the specific input z_0, as follows:

$$Q_2(v) = \min\{\mu_{yellow}(z_0), \mu_2(z_0, v)\} \tag{3.8-14}$$



a b

FIGURE 3.50 (a) Result of computing the minimum of an arbitrary constant, c, and function $\mu_3(z, v)$ from Eq. (3.8-12). The minimum is equivalent to an AND operation. (b) Cross section (dark line) at a specific color, z_0.

† Note that Eq. (3.8-12) is formed from ordered pairs of values $\{\mu_{red}(z), \mu_{mat}(v)\}$, and recall that a set of ordered pairs is commonly called a *Cartesian product*, denoted by $X \times V$, where X is a set of values $\{\mu_{red}(z_1), \mu_{red}(z_2), \ldots, \mu_{red}(z_n)\}$ generated from $\mu_{red}(z)$ by varying z, and V is a similar set of n values generated from $\mu_{med}(v)$ by varying v. Thus, $X \times V = \{(\mu_{red}(z_1), \mu_{med}(v_1)), \ldots, (\mu_{red}(z_n), \mu_{med}(v_n))\}$, and we see from Fig. 3.49(d) that the AND operation involving two variables can be expresses as a mapping from $X \times V$ to the range $[0, 1]$, denoted as $X \times V \rightarrow [0, 1]$. Although we do not use this notation in the present discussion, we mention it here because you are likely to encounter it in the literature on fuzzy sets.

and

$$Q_1(v) = \min\{\mu_{green}(z_0), \mu_1(z_0, v)\} \tag{3.8-15}$$

Each of these equations is the output associated with a particular rule and a specific input. That is, they represent the result of the implication process mentioned a few paragraphs back. Keep in mind that each of these three responses is a fuzzy set, even though the input is a scalar value.

To obtain the overall response, we *aggregate* the individual responses. In the rule base given at the beginning of this section the three rules are associated by the OR operation. Thus, the complete (aggregated) fuzzy output is given by

$$Q = Q_1 \text{ OR } Q_2 \text{ OR } Q_3 \tag{3.8-16}$$

and we see that the overall response is the union of three individual fuzzy sets. Because OR is defined as a max operation, we can write this result as

$$Q(v) = \max_r \left\{ \min_s \{\mu_s(z_0), \mu_r(z_0, v)\} \right\} \tag{3.8-17}$$

for $r = \{1, 2, 3\}$ and $s = \{green, yellow, red\}$. Although it was developed in the context of an example, this expression is perfectly general; to extend it to n rules, we simply let $r = \{1, 2, \ldots, n\}$; similarly, we can expand s to include any finite number of membership functions. Equations (3.8-16) and (3.8-17) say the same thing: The response, Q, of our fuzzy system is the union of the individual fuzzy sets resulting from each rule by the implication process.

Figure 3.51 summarizes graphically the discussion up to this point. Figure 3.51(a) shows the three input membership functions evaluated at z_0, and Fig. 3.51(b) shows the outputs in response to input z_0. These fuzzy sets are the clipped cross sections discussed in connection with Fig. 3.50(b). Note that, numerically, Q_1 consists of all 0s because $\mu_{green}(z_0) = 0$; that is, Q_1 is *empty*, as defined in Section 3.8.2. Figure 3.51(c) shows the final result, Q, itself a fuzzy set formed from the union of Q_1, Q_2, and Q_3.

We have successfully obtained the complete output corresponding to a specific input, but we are still dealing with a fuzzy set. The last step is to obtain a crisp output, v_0, from fuzzy set Q using a process appropriately called *defuzzification*. There are a number of ways to defuzzify Q to obtain a crisp output. One of the approaches used most frequently is to compute the center of gravity of this set (the references cited at the end of this chapter discuss others). Thus, if $Q(v)$ from Eq. (3.8-17) can have K possible values, $Q(1), Q(2), \ldots Q(K)$, its center of gravity is given by

$$v_0 = \frac{\sum_{v=1}^{K} vQ(v)}{\sum_{v=1}^{K} Q(v)} \tag{3.8-18}$$

Evaluating this equation with the (discrete)[†] values of Q in Fig. 3.51(c) yields $v_0 = 72.3$, indicating that the given color z_0 implies a fruit maturity of approximately 72%.

[†] Fuzzy set Q in Fig. 3.51(c) is shown as a solid curve for clarity, but keep in mind that we are dealing with digital quantities in this book, so Q is a digital function.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

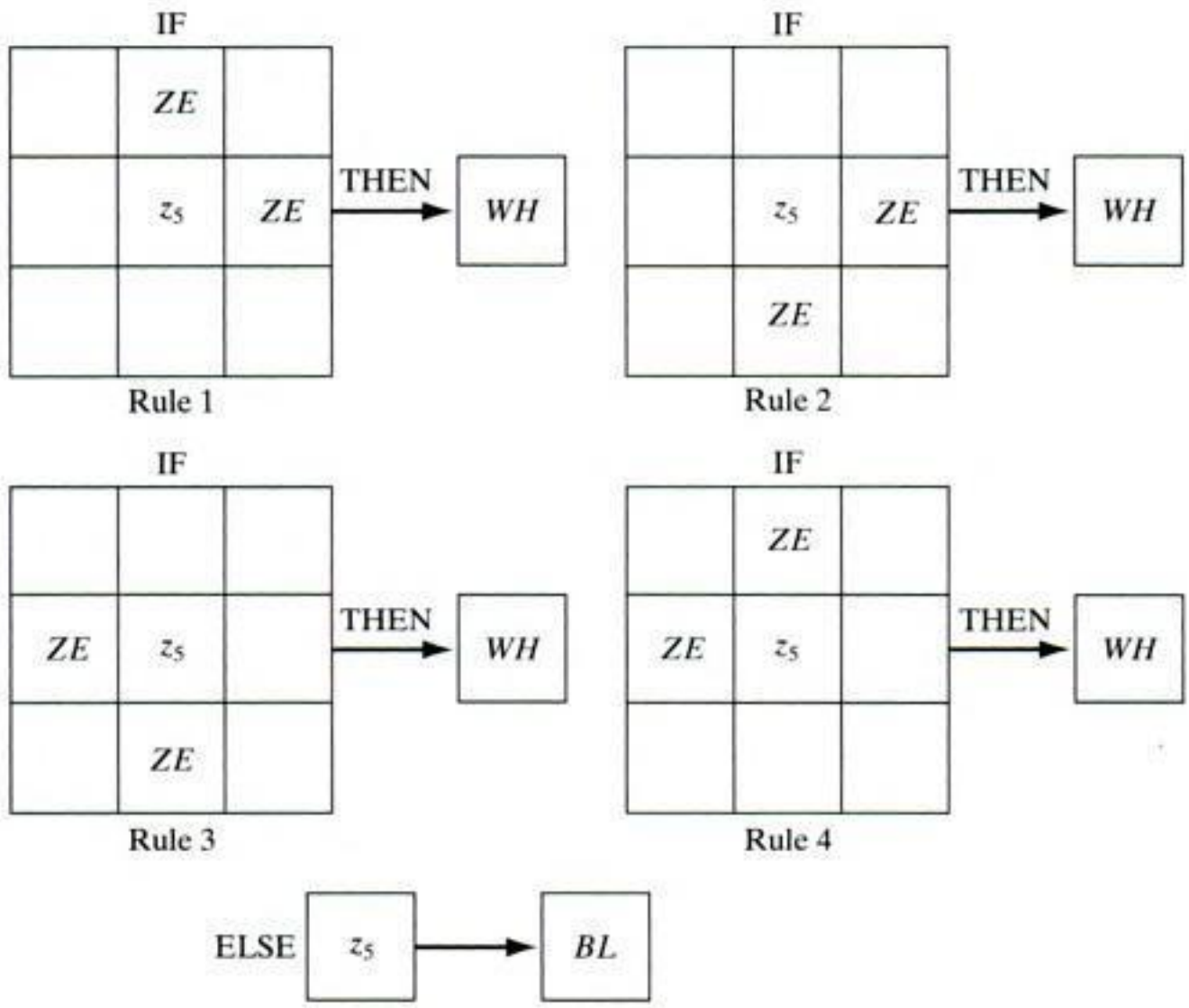


FIGURE 3.58 Fuzzy rules for boundary detection.

gray scale. For example, Fig. 3.59(c) was obtained by performing the intensity scaling defined in Eqs. (2.6-10) and (2.6-11), with $K = L - 1$. The net result is that intensity values in Fig. 3.59(c) span the full gray scale from 0 to $(L - 1)$. ■

a b c

FIGURE 3.59 (a) CT scan of a human head. (b) Result of fuzzy spatial filtering using the membership functions in Fig. 3.57 and the rules in Fig. 3.58. (c) Result after intensity scaling. The thin black picture borders in (b) and (c) were added for clarity; they are not part of the data. (Original image courtesy of Dr. David R. Pickens, Vanderbilt University.)

Summary

The material you have just learned is representative of current techniques used for intensity transformations and spatial filtering. The topics included in this chapter were selected for their value as fundamental material that would serve as a foundation in an evolving field. Although most of the examples used in this chapter were related to image enhancement, the techniques presented are perfectly general, and you will encounter them again throughout the remaining chapters in contexts totally unrelated to enhancement. In the following chapter, we look again at filtering, but using concepts from the frequency domain. As you will see, there is a one-to-one correspondence between the linear spatial filters studied here and frequency domain filters.

References and Further Reading

The material in Section 3.1 is from Gonzalez [1986]. Additional reading for the material in Section 3.2 may be found in Schowengerdt [1983], Poyton [1996], and Russ [1999]. See also the paper by Tsujii et al. [1998] regarding the optimization of image displays. Early references on histogram processing are Hummel [1974], Gonzalez and Fittes [1977], and Woods and Gonzalez [1981]. Stark [2000] gives some interesting generalizations of histogram equalization for adaptive contrast enhancement. Other approaches for contrast enhancement are exemplified by Centeno and Haertel [1997] and Cheng and Xu [2000]. For further reading on exact histogram specification see Coltuc, Bolon, and Chassery [2006]. For extensions of the local histogram equalization method, see Caselles et al. [1999], and Zhu et al. [1999]. See Narendra and Fitch [1981] on the use and implementation of local statistics for image processing. Kim et al. [1997] present an interesting approach combining the gradient with local statistics for image enhancement.

For additional reading on linear spatial filters and their implementation, see Umbaugh [2005], Jain [1989], and Rosenfeld and Kak [1982]. Rank-order filters are discussed in these references as well. Wilburn [1998] discusses generalizations of rank-order filters. The book by Pitas and Venetsanopoulos [1990] also deals with median and other nonlinear spatial filters. A special issue of the *IEEE Transactions in Image Processing* [1996] is dedicated to the topic of nonlinear image processing. The material on high boost filtering is from Schowengerdt [1983]. We will encounter again many of the spatial filters introduced in this chapter in discussions dealing with image restoration (Chapter 5) and edge detection (Chapter 10).

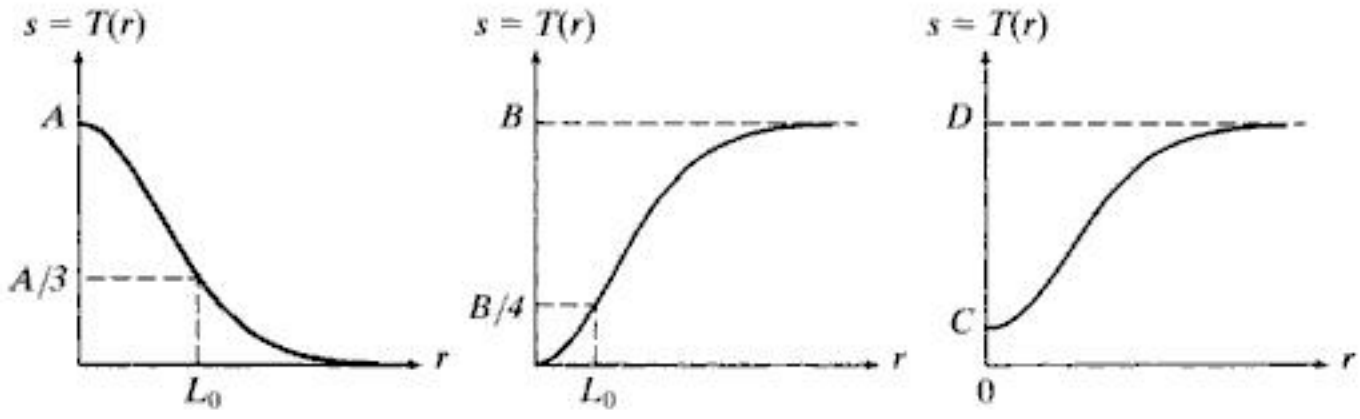
Fundamental references for Section 3.8 are three papers on fuzzy logic by L. A. Zadeh (Zadeh [1965, 1973, 1976]). These papers are well written and worth reading in detail, as they established the foundation for fuzzy logic and some of its applications. An overview of a broad range of applications of fuzzy logic in image processing can be found in the book by Kerre and Nachtegael [2000]. The example in Section 3.8.4 is based on a similar application described by Tizhoosh [2000]. The example in Section 3.8.5 is basically from Russo and Ramponi [1994]. For additional examples of applications of fuzzy sets to intensity transformations and image filtering, see Patrascu [2004] and Nie and Barner [2006], respectively. The preceding range of references from 1965 through 2006 is a good starting point for more detailed study of the many ways in which fuzzy sets can be used in image processing. Software implementation of most of the methods discussed in this chapter can be found in Gonzalez, Woods, and Eddins [2004].

Problems

★**3.1** Give a single intensity transformation function for spreading the intensities of an image so the lowest intensity is C and the highest is $L - 1$.

Detailed solutions to the problems marked with a star can be found in the book Web site. The site also contains suggested projects based on the material in this chapter.

3.2 Exponentials of the form $e^{-\alpha r^2}$, with α a positive constant, are useful for constructing smooth intensity transformation functions. Start with this basic function and construct transformation functions having the general shapes shown in the following figures. The constants shown are *input* parameters, and your proposed transformations must include them in their specification. (For simplicity in your answers, L_0 is not a required parameter in the third curve.)



3.3 ★**(a)** Give a continuous function for implementing the contrast stretching transformation shown in Fig. 3.2(a). In addition to m, your function must include a parameter, E, for controlling the slope of the function as it transitions from low to high intensity values. Your function should be normalized so that its minimum and maximum values are 0 and 1, respectively.

(b) Sketch a family of transformations as a function of parameter E, for a fixed value $m = L/4$, where L is the number of intensity levels in the image.

(c) What is the smallest value of E that will make your function *effectively* perform as the function in Fig. 3.2(b)? In other words, your function does not have to be identical to Fig. 3.2(b). It just has to yield the same result of producing a binary image. Assume that you are working with 8-bit images, and let $m = 128$. Let C denote the smallest positive number representable in the computer you are using.

3.4 Propose a set of intensity-slicing transformations capable of producing all the individual bit planes of an 4-bit monochrome image. (For example, a transformation function with the property $T(r) = 0$ for r in the range $[0, 7]$, and $T(r) = 15$ for r in the range $[8, 15]$ produces an image of the 4th bit plane in an 8-bit image.)

3.5 ★**(a)** What effect would setting to zero the half of lower-order bit planes have on the histogram of an image in general?

(b) What would be the effect on the histogram if we set to zero the half of higher-order bit planes instead?

★**3.6** Explain why the discrete histogram equalization technique does not, in general, yield a flat histogram.

3.7 Suppose that a digital image is subjected to histogram equalization. Show that a second pass of histogram equalization (on the histogram-equalized image) will produce exactly the same result as the first pass.

3.8 In some applications it is useful to model the histogram of input images as Gaussian probability density functions of the form

$$p_r(r) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(r-m)^2}{2\sigma^2}}$$

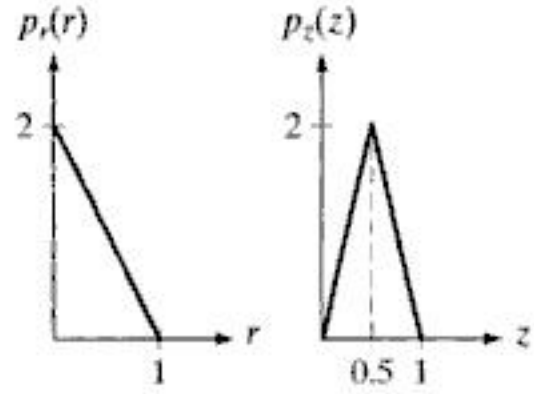
where m and σ are the mean and standard deviation of the Gaussian PDF. The approach is to let m and σ be measures of average intensity and contrast of a given image. What is the transformation function you would use for histogram equalization?

★**3.9** Assuming continuous values, show by example that it is possible to have a case in which the transformation function given in Eq. (3.3-4) satisfies conditions (a) and (b) in Section 3.3.1, but its inverse may fail to be single valued.

3.10 **(a)** Show that the discrete transformation function given in Eq. (3.3-8) for histogram equalization satisfies conditions (a) and (b) in Section 3.3.1.

★**(b)** Show that the inverse discrete transformation in Eq. (3.3-9) satisfies conditions (a′) and (b) in Section 3.3.1 only if none of the intensity levels $r_k, k = 0, 1, \ldots, L-1$, are missing.

3.11 An image with intensities in the range $[0, 1]$ has the PDF $p_r(r)$ shown in the following diagram. It is desired to transform the intensity levels of this image so that they will have the specified $p_z(z)$ shown. Assume continuous quantities and find the transformation (in terms of r and z) that will accomplish this.



★**3.12** Propose a method for updating the local histogram for use in the local enhancement technique discussed in Section 3.3.3.

3.13 Two images, $f(x, y)$ and $g(x, y)$, have histograms h_f and h_g. Give the conditions under which you can determine the histograms of

★**(a)** $f(x, y) + g(x, y)$

(b) $f(x, y) - g(x, y)$

(c) $f(x, y) \times g(x, y)$

(d) $f(x, y) \div g(x, y)$

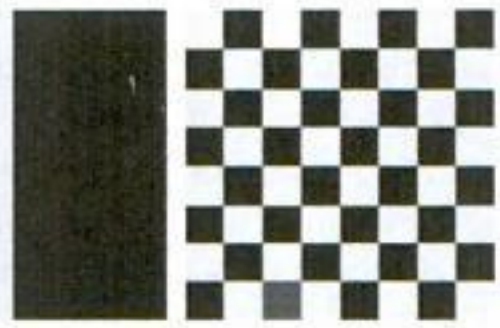
(e) $f(x, y) * g(x, y)$

in terms of h_f and h_g. Explain how to obtain the histogram in each case.

3.14 The images shown on the next page are quite different, but their histograms are the same. Suppose that each image is blurred with a 3×3 averaging mask.

(a) Would the histograms of the blurred images still be equal? Explain.

(b) If your answer is no, sketch the two histograms.



3.15 The implementation of linear spatial filters requires moving the center of a mask throughout an image and, at each location, computing the sum of products of the mask coefficients with the corresponding pixels at that location (see Section 3.4). A lowpass filter can be implemented by setting all coefficients to 1, allowing use of a so-called *box-filter* or *moving-average* algorithm, which consists of updating only the part of the computation that changes from one location to the next.

★ **(a)** Formulate such an algorithm for an $n \times n$ filter, showing the nature of the computations involved and the scanning sequence used for moving the mask around the image.

(b) The ratio of the number of computations performed by a brute-force implementation to the number of computations performed by the box-filter algorithm is called the *computational advantage*. Obtain the computational advantage in this case and plot it as a function of n for $n > 1$. The $1/n^2$ scaling factor is common to both approaches, so you need not consider it in obtaining the computational advantage. Assume that the image has an outer border of zeros that is wide enough to allow you to ignore border effects in your analysis.

3.16 ★ **(a)** Suppose that you filter an image, $f(x, y)$, with a spatial filter mask, $w(x, y)$, using convolution, as defined in Eq. (3.4-2), where the mask is smaller than the image in both spatial directions. Show the important property that, if the coefficients of the mask sum to zero, then the sum of all the elements in the resulting convolution array (filtered image) will be zero also (you may ignore computational inaccuracies). Also, you may assume that the border of the image has been padded with the appropriate number of zeros.

(b) Would the result to (a) be the same if the filtering is implemented using correlation, as defined in Eq. (3.4-1)?

3.17 Discuss the limiting effect of repeatedly applying a 3×3 lowpass spatial filter to a digital image. You may ignore border effects. Is this effect different from applying a 5×5 filter?

3.18 ★ **(a)** It was stated in Section 3.5.2 that isolated clusters of dark or light (with respect to the background) pixels whose area is less than one-half the area of a median filter are eliminated (forced to the median value of the neighbors) by the filter. Assume a filter of size $n \times n$, with n odd, and explain why this is so.

(b) Consider an image having various sets of pixel clusters. Assume that all points in a cluster are lighter or darker than the background (but not both simultaneously in the same cluster), and that the area of each cluster is less than or equal to $n^2/2$. In terms of n, under what condition would one or more of these clusters cease to be isolated in the sense described in part (a)?

★**3.19** **(a)** Develop a procedure for computing the median of an $n \times n$ neighborhood.

(b) Propose a technique for updating the median as the center of the neighborhood is moved from pixel to pixel.

3.20 **(a)** In a character recognition application, text pages are reduced to binary form using a thresholding transformation function of the form shown in Fig. 3.2(b). This is followed by a procedure that thins the characters until they become strings of binary 1s on a background of 0s. Due to noise, the binarization and thinning processes result in broken strings of characters with gaps ranging from 1 to 3 pixels. One way to "repair" the gaps is to run an averaging mask over the binary image to blur it, and thus create bridges of nonzero pixels between gaps. Give the (odd) size of the smallest averaging mask capable of performing this task.

(b) After bridging the gaps, it is desired to threshold the image in order to convert it back to binary form. For your answer in (a), what is the minimum value of the threshold required to accomplish this, without causing the segments to break up again?

★**3.21** The three images shown were blurred using square averaging masks of sizes $n = 23$, 25, and 45, respectively. The vertical bars on the left lower part of (a) and (c) are blurred, but a clear separation exists between them. However, the bars have merged in image (b), in spite of the fact that the mask that produced this image is significantly smaller than the mask that produced image (c). Explain the reason for this.

(a) **(b)** **(c)**

3.22 Consider an application such as the one shown in Fig. 3.34, in which it is desired to eliminate objects smaller than those enclosed by a square of size $q \times q$ pixels. Suppose that we want to reduce the average intensity of those objects to one-tenth of their original average value. In this way, those objects will be closer to the intensity of the background and they can then be eliminated by thresholding. Give the (odd) size of the smallest averaging mask that will accomplish the desired reduction in average intensity in only one pass of the mask over the image.

3.23 In a given application an averaging mask is applied to input images to reduce noise, and then a Laplacian mask is applied to enhance small details. Would the result be the same if the order of these operations were reversed?

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

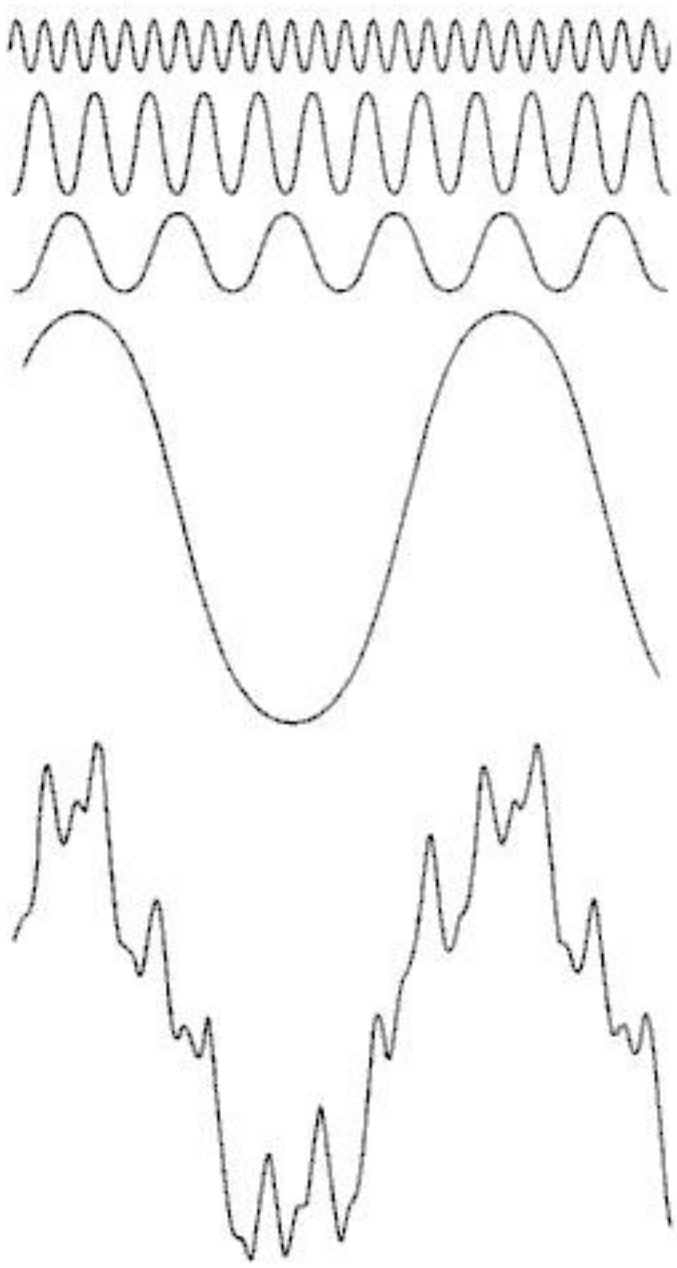


FIGURE 4.1 The function at the bottom is the sum of the four functions above it. Fourier's idea in 1807 that periodic functions could be represented as a weighted sum of sines and cosines was met with skepticism.

a host of signals of exceptional importance, ranging from medical monitors and scanners to modern electronic communications.

We will be dealing only with functions (images) of finite duration, so the Fourier transform is the tool in which we are interested. The material in the following section introduces the Fourier transform and the frequency domain. It is shown that Fourier techniques provide a meaningful and practical way to study and implement a host of image processing approaches. In some cases, these approaches are similar to the ones we developed in Chapter 3.

4.1.2 About the Examples in this Chapter

As in Chapter 3, most of the image filtering examples in this chapter deal with image enhancement. For example, smoothing and sharpening are traditionally associated with image enhancement, as are techniques for contrast manipulation. By its very nature, beginners in digital image processing find enhancement to be interesting and relatively simple to understand. Therefore, using

examples from image enhancement in this chapter not only saves having an extra chapter in the book but, more importantly, is an effective tool for introducing newcomers to filtering techniques in the frequency domain. We use frequency domain processing methods for other applications in Chapters 5, 8, 10, and 11.

4.2 Preliminary Concepts

In order to simplify the progression of ideas presented in this chapter, we pause briefly to introduce several of the basic concepts that underlie the material that follows in later sections.

4.2.1 Complex Numbers

A complex number, C, is defined as

$$C = R + jI \tag{4.2-1}$$

where R and I are real numbers, and j is an imaginary number equal to the square of -1; that is, $j = \sqrt{-1}$. Here, R denotes the *real part* of the complex number and I its *imaginary part*. Real numbers are a subset of complex numbers in which $I = 0$. The *conjugate* of a complex number C, denoted C^*, is defined as

$$C^* = R - jI \tag{4.2-2}$$

Complex numbers can be viewed geometrically as points in a plane (called the *complex plane*) whose abscissa is the *real axis* (values of R) and whose ordinate is the *imaginary axis* (values of I). That is, the complex number $R + jI$ is point (R, I) in the rectangular coordinate system of the complex plane.

Sometimes, it is useful to represent complex numbers in polar coordinates,

$$C = |C|(\cos\theta + j\sin\theta) \tag{4.2-3}$$

where $|C| = \sqrt{R^2 + I^2}$ is the length of the vector extending from the origin of the complex plane to point (R, I), and θ is the angle between the vector and the real axis. Drawing a simple diagram of the real and complex axes with the vector in the first quadrant will reveal that $\tan\theta = (I/R)$ or $\theta = \arctan(I/R)$. The arctan function returns angles in the range $[-\pi/2, \pi/2]$. However, because I and R can be positive and negative independently, we need to be able to obtain angles in the full range $[-\pi, \pi]$. This is accomplished simply by keeping track of the sign of I and R when computing θ. Many programming languages do this automatically via so called *four-quadrant arctangent* functions. For example, MATLAB provides the function `atan2(Imag, Real)` for this purpose.

Using Euler's formula,

$$e^{j\theta} = \cos\theta + j\sin\theta \tag{4.2-4}$$

where $e = 2.71828\ldots$, gives the following familiar representation of complex numbers in polar coordinates,

$$C = |C|e^{j\theta} \tag{4.2-5}$$

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

of the sifting property involves an impulse located at an arbitrary point t_0, denoted by $\delta(t - t_0)$. In this case, the sifting property becomes

$$\int_{-\infty}^{\infty} f(t)\,\delta(t - t_0)\,dt = f(t_0) \tag{4.2-10}$$

which yields the value of the function at the impulse location, t_0. For instance, if $f(t) = \cos(t)$, using the impulse $\delta(t - \pi)$ in Eq. (4.2-10) yields the result $f(\pi) = \cos(\pi) = -1$. The power of the sifting concept will become quite evident shortly.

Let x represent a *discrete* variable. The *unit discrete impulse*, $\delta(x)$, serves the same purposes in the context of discrete systems as the impulse $\delta(t)$ does when working with continuous variables. It is defined as

$$\delta(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases} \tag{4.2-11a}$$

Clearly, this definition also satisfies the discrete equivalent of Eq. (4.2-8b):

$$\sum_{x=-\infty}^{\infty} \delta(x) = 1 \tag{4.2-11b}$$

The sifting property for discrete variables has the form

$$\sum_{x=-\infty}^{\infty} f(x)\,\delta(x) = f(0) \tag{4.2-12}$$

or, more generally using a discrete impulse located at $x = x_0$,

$$\sum_{x=-\infty}^{\infty} f(x)\,\delta(x - x_0) = f(x_0) \tag{4.2-13}$$

As before, we see that the sifting property simply yields the value of the function at the location of the impulse. Figure 4.2 shows the unit discrete impulse diagrammatically. Unlike its continuous counterpart, the discrete impulse is an ordinary function.

Of particular interest later in this section is an *impulse train*, $s_{\Delta T}(t)$, defined as the sum of infinitely many *periodic* impulses ΔT units apart:

$$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} \delta(t - n\Delta T) \tag{4.2-14}$$

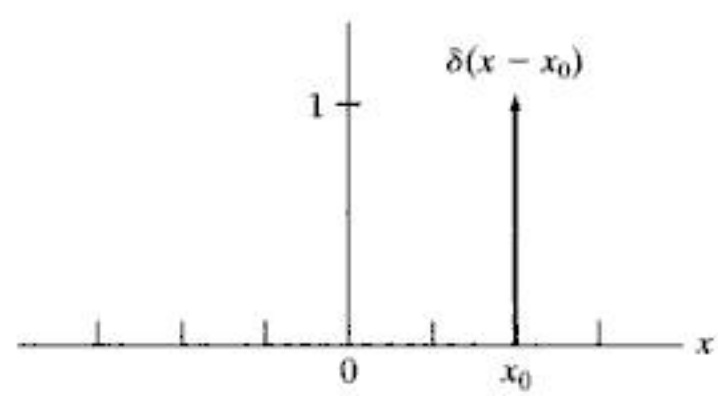


FIGURE 4.2
A unit discrete impulse located at $x = x_0$. Variable x is discrete, and δ is 0 everywhere except at $x = x_0$.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

where the third line follows from the sifting property in Eq. (4.2-10) and the last line follows from Euler's formula. These last two lines are equivalent representations of a unit circle centered on the origin of the complex plane.

In Section 4.3, we make use of the Fourier transform of a periodic impulse train. Obtaining this transform is not as straightforward as we just showed for individual impulses. However, understanding how to derive the transform of an impulse train is quite important, so we take the time to derive it in detail here. We start by noting that the only difference in the *form* of Eqs. (4.2-16) and (4.2-17) is the sign of the exponential. Thus, if a function $f(t)$ has the Fourier transform $F(\mu)$, then the latter function evaluated at t, that is, $F(t)$, must have the transform $f(-\mu)$. Using this *symmetry* property and given, as we showed above, that the Fourier transform of an impulse $\delta(t - t_0)$ is $e^{-j2\pi\mu t_0}$, it follows that the function $e^{-j2\pi t_0 t}$ has the transform $\delta(-\mu - t_0)$. By letting $-t_0 = a$, it follows that the transform of $e^{j2\pi at}$ is $\delta(-\mu + a) = \delta(\mu - a)$, where the last step is true because δ is not zero only when $\mu = a$, which is the same result for either $\delta(-\mu + a)$ or $\delta(\mu - a)$, so the two forms are equivalent.

The impulse train $s_{\Delta T}(t)$ in Eq. (4.2-14) is periodic with period ΔT, so we know from Section 4.2.2 that it can be expressed as a Fourier series:

$$s_{\Delta T}(t) = \sum_{n=-\infty}^{\infty} c_n e^{j\frac{2\pi n}{\Delta T}t}$$

where

$$c_n = \frac{1}{\Delta T}\int_{-\Delta T/2}^{\Delta T/2} s_{\Delta T}(t) e^{-j\frac{2\pi n}{\Delta T}t} dt$$

With reference to Fig. 4.3, we see that the integral in the interval $[-\Delta T/2, \Delta T/2]$ encompasses only the impulse of $s_{\Delta T}(t)$ that is located at the origin. Therefore, the preceding equation becomes

$$\begin{aligned} c_n &= \frac{1}{\Delta T}\int_{-\Delta T/2}^{\Delta T/2} \delta(t) e^{-j\frac{2\pi n}{\Delta T}t} dt \\ &= \frac{1}{\Delta T} e^0 \\ &= \frac{1}{\Delta T} \end{aligned}$$

The Fourier series expansion then becomes

$$s_{\Delta T}(t) = \frac{1}{\Delta T}\sum_{n=-\infty}^{\infty} e^{j\frac{2\pi n}{\Delta T}t}$$

Our objective is to obtain the Fourier transform of this expression. Because summation is a linear process, obtaining the Fourier transform of a sum is

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

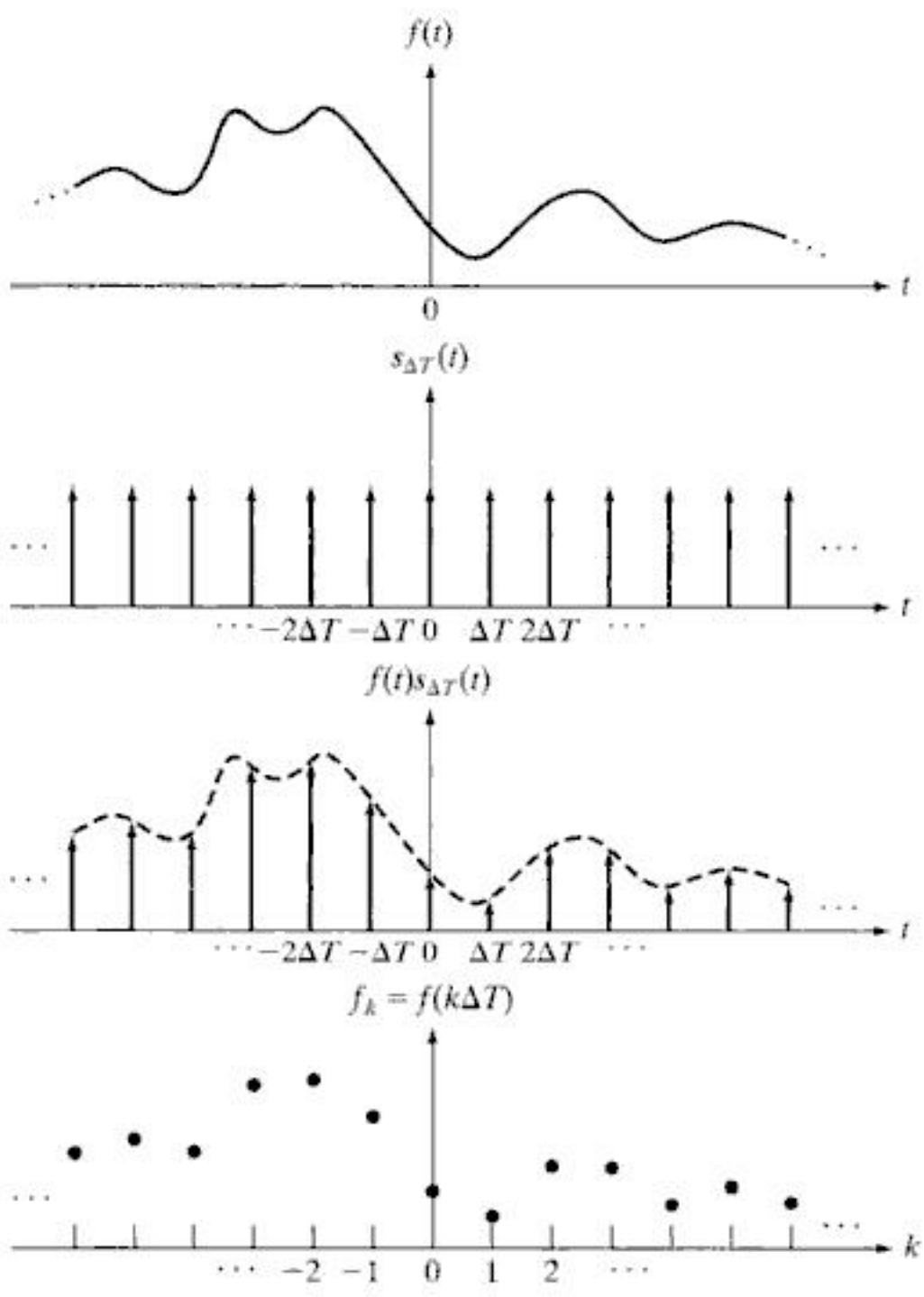
4.3 Sampling and the Fourier Transform of Sampled Functions

In this section, we use the concepts from Section 4.2 to formulate a basis for expressing sampling mathematically. This will lead us, starting from basic principles, to the Fourier transform of sampled functions.

4.3.1 Sampling

Continuous functions have to be converted into a sequence of discrete values before they can be processed in a computer. This is accomplished by using sampling and quantization, as introduced in Section 2.4. In the following discussion, we examine sampling in more detail.

With reference to Fig. 4.5, consider a continuous function, $f(t)$, that we wish to sample at uniform intervals (ΔT) of the independent variable t. We



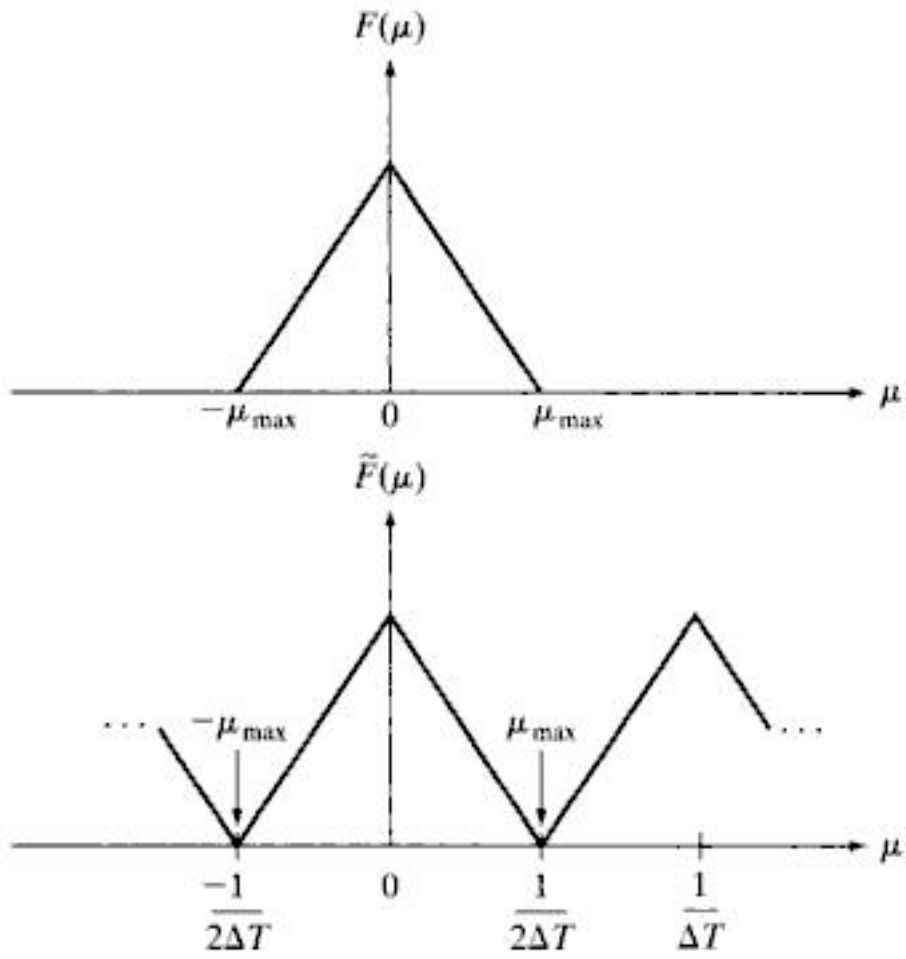
a
b
c
d

FIGURE 4.5
(a) A continuous function. (b) Train of impulses used to model the sampling process. (c) Sampled function formed as the product of (a) and (b). (d) Sample values obtained by integration and using the sifting property of the impulse. (The dashed line in (c) is shown for reference. It is not part of the data.)

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a
b

FIGURE 4.7
(a) Transform of a band-limited function.
(b) Transform resulting from critically sampling the same function.

Extracting from $\tilde{F}(\mu)$ a single period that is equal to $F(\mu)$ is possible if the separation between copies is sufficient (see Fig. 4.6). In terms of Fig. 4.7(b), sufficient separation is guaranteed if $1/2\Delta T > \mu_{max}$ or

$$\frac{1}{\Delta T} > 2\mu_{max} \tag{4.3-6}$$

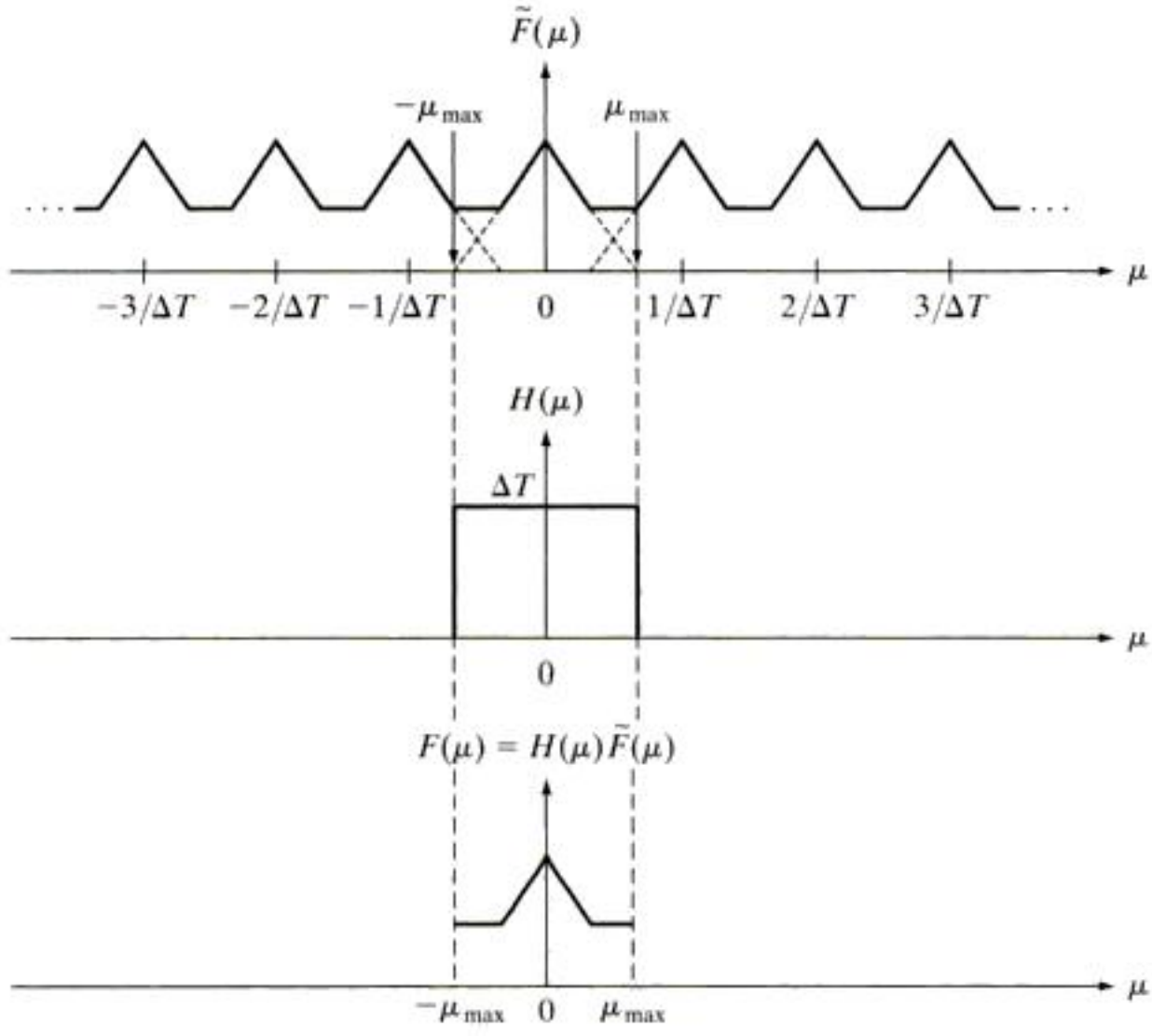
This equation indicates that a continuous, band-limited function can be recovered completely from a set of its samples if the samples are acquired at a rate exceeding twice the highest frequency content of the function. This result is known as the *sampling theorem*.† We can say based on this result that no information is lost if a continuous, band-limited function is represented by samples acquired at a rate greater than twice the highest frequency content of the function. Conversely, we can say that the *maximum* frequency that can be "captured" by sampling a signal at a rate $1/\Delta T$ is $\mu_{max} = 1/2\Delta T$. Sampling at the Nyquist rate sometimes is sufficient for perfect function recovery, but there are cases in which this leads to difficulties, as we illustrate later in Example 4.3. Thus, the sampling theorem specifies that sampling must exceed the Nyquist rate.

A sampling rate equal to *exactly* twice the highest frequency is called the *Nyquist rate*.

†The sampling theorem is a cornerstone of digital signal processing theory. It was first formulated in 1928 by Harry Nyquist, a Bell Laboratories scientist and engineer. Claude E. Shannon, also from Bell Labs, proved the theorem formally in 1949. The renewed interest in the sampling theorem in the late 1940s was motivated by the emergence of early digital computing systems and modern communications, which created a need for methods dealing with digital (sampled) data.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a
b
c

FIGURE 4.9 (a) Fourier transform of an under-sampled, band-limited function. (Interference from adjacent periods is shown dashed in this figure). (b) The same ideal lowpass filter used in Fig. 4.8(b). (c) The product of (a) and (b). The interference from adjacent periods results in aliasing that prevents perfect recovery of $F(\mu)$ and, therefore, of the original, band-limited continuous function. Compare with Fig. 4.8.

components extending to infinity. Therefore, no function of finite duration can be band-limited. Conversely, a function that is band-limited must extend from $-\infty$ to ∞.[†]

We conclude that aliasing is an inevitable fact of working with sampled records of finite length for the reasons stated in the previous paragraph. In practice, the effects of aliasing can be *reduced* by smoothing the input function to attenuate its higher frequencies (e.g., by defocusing in the case of an image). This process, called *anti-aliasing*, has to be done *before* the function is sampled because aliasing is a sampling issue that cannot be "undone after the fact" using computational techniques.

[†]An important special case is when a function that extends from $-\infty$ to ∞ is band-limited *and* periodic. In this case, the function can be truncated and still be band-limited, *provided* that the truncation encompasses *exactly* an integral number of periods. A single truncated period (and thus the function) can be represented by a set of discrete samples satisfying the sampling theorem, taken over the truncated interval.

EXAMPLE 4.3: Aliasing.

■ Figure 4.10 shows a classic illustration of aliasing. A pure sine wave extending infinitely in both directions has a single frequency so, obviously, it is band-limited. Suppose that the sine wave in the figure (ignore the large dots for now) has the equation $\sin(\pi t)$, and that the horizontal axis corresponds to time, t, in seconds. The function crosses the axis at $t = \ldots -1, 0, 1, 2, 3 \ldots$.

The period, P, of $\sin(\pi t)$ is 2 s, and its frequency is $1/P$, or $1/2$ cycles/s. According to the sampling theorem, we can recover this signal from a set of its samples if the sampling rate, $1/\Delta T$, exceeds twice the highest frequency of the signal. This means that a sampling rate greater than 1 sample/s $[2 \times (1/2) = 1]$, or $\Delta T < 1$ s, is required to recover the signal. Observe that sampling this signal at *exactly* twice the frequency (1 sample/s), with samples taken at $t = \ldots -1, 0, 1, 2, 3 \ldots$, results in $\ldots \sin(-\pi), \sin(0), \sin(\pi), \sin(2\pi), \ldots$, which are all 0. This illustrates the reason why the sampling theorem requires a sampling rate that exceeds twice the highest frequency, as mentioned earlier.

Recall that 1 cycle/s is defined as 1 Hz.

The large dots in Fig. 4.10 are samples taken uniformly at a rate of less than 1 sample/s (in fact, the separation between samples exceeds 2 s, which gives a sampling rate lower than 1/2 samples/s). The sampled signal *looks* like a sine wave, but its frequency is about *one-tenth* the frequency of the original. This sampled signal, having a frequency well below anything present in the original continuous function is an example of aliasing. Given just the samples in Fig. 4.10, the seriousness of aliasing in a case such as this is that we would have no way of knowing that these samples are not a true representation of the original function. As you will see in later in this chapter, aliasing in images can produce similarly misleading results. ■

4.3.5 Function Reconstruction (Recovery) from Sampled Data

In this section, we show that reconstruction of a function from a set of its samples reduces in practice to interpolating between the samples. Even the simple act of displaying an image requires reconstruction of the image from its samples

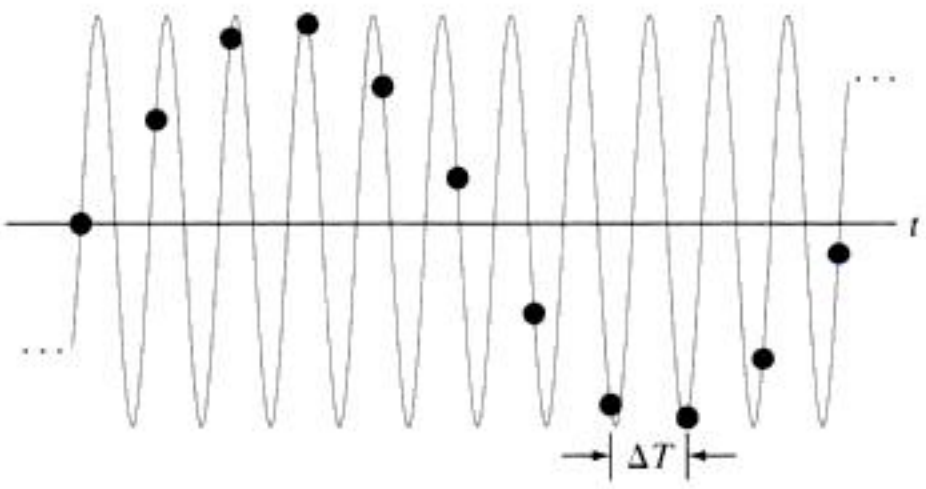


FIGURE 4.10 Illustration of aliasing. The under-sampled function (black dots) looks like a sine wave having a frequency much lower than the frequency of the continuous signal. The period of the sine wave is 2 s, so the zero crossings of the horizontal axis occur every second. ΔT is the separation between samples.

by the display medium. Therefore, it is important to understand the fundamentals of sampled data reconstruction. Convolution is central to developing this understanding, showing again the importance of this concept.

The discussion of Fig. 4.8 and Eq. (4.3-8) outlines the procedure for perfect recovery of a band-limited function from its samples using frequency domain methods. Using the convolution theorem, we can obtain the equivalent result in the spatial domain. From Eq. (4.3-8), $F(\mu) = H(\mu)\tilde{F}(\mu)$, so it follows that

$$\begin{aligned} f(t) &= \Im^{-1}\{F(\mu)\} \\ &= \Im^{-1}\{H(\mu)\tilde{F}(\mu)\} \\ &= h(t) \star \tilde{f}(t) \end{aligned} \tag{4.3-11}$$

where the last step follows from the convolution theorem, Eq. (4.2-21). It can be shown (Problem 4.6) that substituting Eq. (4.3-1) for $\tilde{f}(t)$ into Eq. (4.3-11) and then using Eq. (4.2-20) leads to the following *spatial domain* expression for $f(t)$:

$$f(t) = \sum_{n=-\infty}^{\infty} f(n\Delta T)\,\mathrm{sinc}\left[(t - n\Delta T)/n\Delta T\right] \tag{4.3-12}$$

where the sinc function is defined in Eq. (4.2-19). This result is not unexpected because the inverse Fourier transform of the box filter, $H(\mu)$, is a sinc function (see Example 4.1). Equation (4.3-12) shows that the perfectly reconstructed function is an infinite sum of sinc functions weighted by the sample values, and has the important property that the reconstructed function is identically equal to the sample values at multiple integer increments of ΔT. That is, for any $t = k\,\Delta T$, where k is an integer, $f(t)$ is equal to the kth sample $f(k\Delta T)$. This follows from Eq. (4.3-12) because $\mathrm{sinc}(0) = 1$ and $\mathrm{sinc}(m) = 0$ for any other integer value of m. Between sample points, values of $f(t)$ are *interpolations* formed by the sum of the sinc functions.

Equation (4.3-12) requires an infinite number of terms for the interpolations between samples. In practice, this implies that we have to look for approximations that are finite interpolations between samples. As we discussed in Section 2.4.4, the principal interpolation approaches used in image processing are nearest-neighbor, bilinear, and bicubic interpolation. We discuss the effects of interpolation on images in Section 4.5.4.

4.4 The Discrete Fourier Transform (DFT) of One Variable

One of the key goals of this chapter is the derivation of the *discrete Fourier transform* (DFT) starting from basic principles. The material up to this point may be viewed as the foundation of those basic principles, so now we have in place the necessary tools to derive the DFT.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

It can be shown (Problem 4.9) that both the forward and inverse discrete transforms are infinitely periodic, with period M. That is,

$$F(u) = F(u + kM) \tag{4.4-8}$$

and

$$f(x) = f(x + kM) \tag{4.4-9}$$

where k is an integer.

The discrete equivalent of the convolution in Eq. (4.2-20) is

$$f(x) \star h(x) = \sum_{m=0}^{M-1} f(m)h(x - m) \tag{4.4-10}$$

for $x = 0, 1, 2, \ldots, M - 1$. Because in the preceding formulations the functions are periodic, their convolution also is periodic. Equation (4.4-10) gives one period of the periodic convolution. For this reason, the process inherent in this equation often is referred to as *circular convolution*, and is a direct result of the periodicity of the DFT and its inverse. This is in contrast with the convolution you studied in Section 3.4.2, in which values of the displacement, x, were determined by the requirement of sliding one function completely past the other, and were not fixed to the range $[0, M - 1]$ as in circular convolution. We discuss this difference and its significance in Section 4.6.3 and in Fig. 4.28.

It is not obvious why the discrete function $f(x)$ should be periodic, considering that the continuous function from which it was sampled may not be. One informal way to reason this out is to keep in mind that sampling results in a periodic DFT. It is logical that $f(x)$, which is the inverse DFT, has to be periodic also for the DFT pair to exist.

Finally, we point out that the convolution theorem given in Eqs. (4.2-21) and (4.2-22) is applicable also to discrete variables (Problem 4.10).

4.4.2 Relationship Between the Sampling and Frequency Intervals

If $f(x)$ consists of M samples of a function $f(t)$ taken ΔT units apart, the duration of the record comprising the set $\{f(x)\}$, $x = 0, 1, 2, \ldots, M - 1$, is

$$T = M\Delta T \tag{4.4-11}$$

The corresponding spacing, Δu, in the discrete frequency domain follows from Eq. (4.4-3):

$$\Delta u = \frac{1}{M\Delta T} = \frac{1}{T} \tag{4.4-12}$$

The entire frequency range spanned by the M components of the DFT is

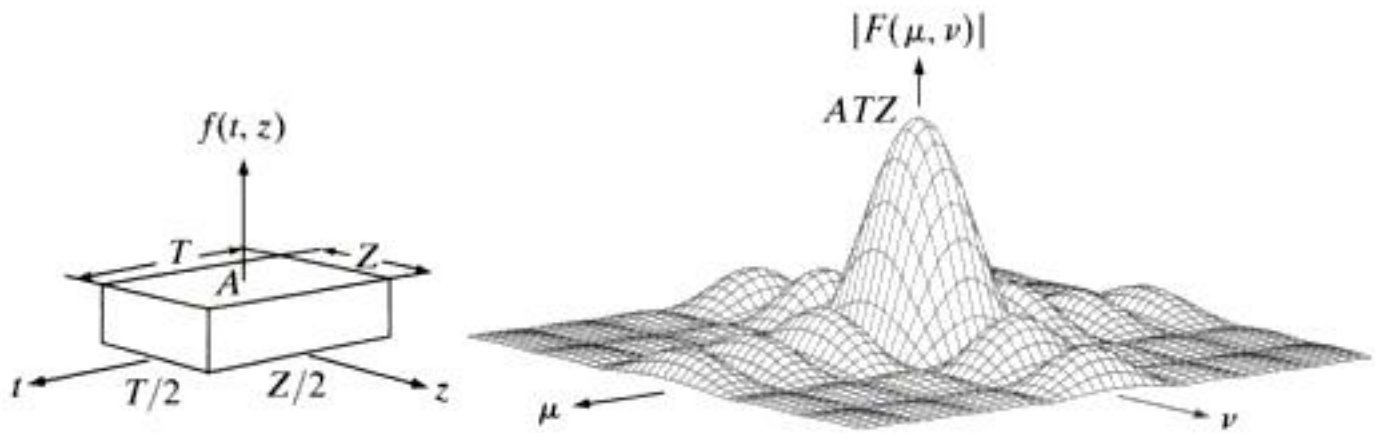
$$\Omega = M\Delta u = \frac{1}{\Delta T} \tag{4.4-13}$$

Thus, we see from Eqs. (4.4-12) and (4.4-13) that that the resolution in frequency, Δu, of the DFT depends on the duration T over which the continuous function, $f(t)$, is sampled, and the range of frequencies spanned by the DFT depends on the sampling interval ΔT. Observe that both expressions exhibit *inverse* relationships with respect to T and ΔT.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a b

FIGURE 4.13 (a) A 2-D function, and (b) a section of its spectrum (not to scale). The block is longer along the t-axis, so the spectrum is more "contracted" along the μ-axis. Compare with Fig. 4.4.

the values of T and Z. Thus, the larger T and Z are, the more "contracted" the spectrum will become, and vice versa. ■

4.5.3 Two-Dimensional Sampling and the 2-D Sampling Theorem

In a manner similar to the 1-D case, sampling in two dimensions can be modeled using the sampling function (2-D impulse train):

$$s_{\Delta T \Delta Z}(t, z) = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \delta(t - m\Delta T, z - n\Delta Z) \qquad (4.5\text{-}9)$$

where ΔT and ΔZ are the separations between samples along the t- and z-axis of the continuous function $f(t, z)$. Equation (4.5-9) describes a set of periodic impulses extending infinitely along the two axes (Fig. 4.14). As in the 1-D case illustrated in Fig. 4.5, multiplying $f(t, z)$ by $s_{\Delta T \Delta Z}(t, z)$ yields the sampled function.

Function $f(t, z)$ is said to be *band-limited* if its Fourier transform is 0 outside a rectangle established by the intervals $[-\mu_{\max}, \mu_{\max}]$ and $[-\nu_{\max}, \nu_{\max}]$; that is,

$$F(\mu, \nu) = 0 \quad \text{for } |\mu| \geq \mu_{\max} \text{ and } |\nu| \geq \nu_{\max} \qquad (4.5\text{-}10)$$

The *two-dimensional sampling theorem* states that a continuous, band-limited function $f(t, z)$ can be recovered with no error from a set of its samples if the sampling intervals are

$$\Delta T < \frac{1}{2\mu_{\max}} \qquad (4.5\text{-}11)$$

and

$$\Delta Z < \frac{1}{2\nu_{\max}} \qquad (4.5\text{-}12)$$

or, expressed in terms of the sampling rate, if

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

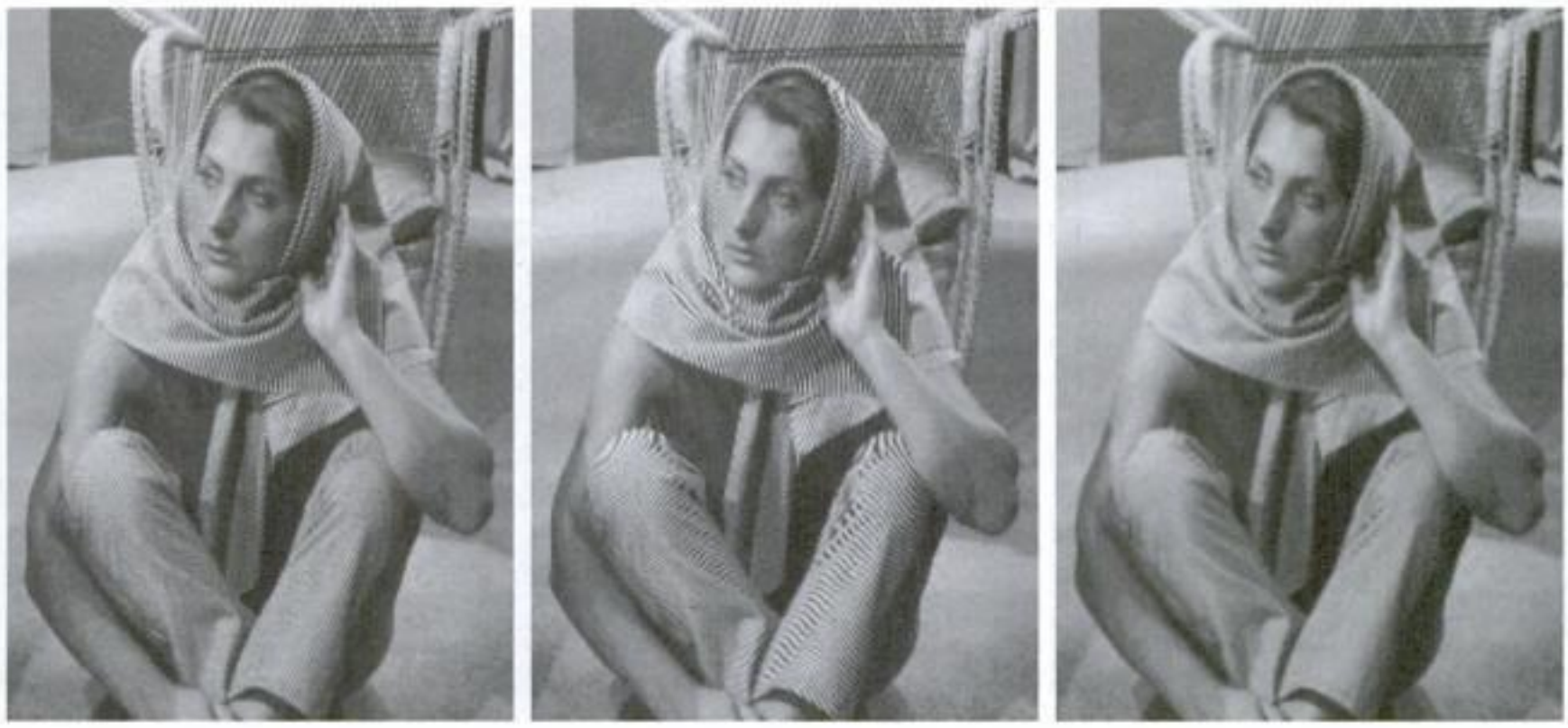
instance, to double the size of an image, we duplicate each column. This doubles the image size in the horizontal direction. Then, we duplicate each row of the enlarged image to double the size in the vertical direction. The same procedure is used to enlarge the image any integer number of times. The intensity-level assignment of each pixel is predetermined by the fact that new locations are exact duplicates of old locations.

Image shrinking is done in a manner similar to zooming. Under-sampling is achieved by row-column deletion (e.g., to shrink an image by one-half, we delete every other row and column). We can use the zooming grid analogy in Section 2.4.4 to visualize the concept of shrinking by a non-integer factor, except that we now expand the grid to fit over the original image, do intensity-level interpolation, and then shrink the grid back to its specified size. To reduce aliasing, it is a good idea to blur an image slightly before shrinking it (we discuss frequency domain blurring in Section 4.8). An alternate technique is to *super-sample* the original scene and then reduce (resample) its size by row and column deletion. This can yield sharper results than with smoothing, but it clearly requires access to the original scene. Clearly, if we have no access to the original scene (as typically is the case in practice) super-sampling is not an option.

The process of resampling an image without using band-limiting blurring is called *decimation*.

EXAMPLE 4.7: Illustration of aliasing in resampled images.

■ The effects of aliasing generally are worsened when the size of a digital image is reduced. Figure 4.17(a) is an image purposely created to illustrate the effects of aliasing (note the thinly-spaced parallel lines in all garments worn by the subject). There are no objectionable artifacts in Fig. 4.17(a), indicating that



a b c

FIGURE 4.17 Illustration of aliasing on resampled images. (a) A digital image with negligible visual aliasing. (b) Result of resizing the image to 50% of its original size by pixel deletion. Aliasing is clearly visible. (c) Result of blurring the image in (a) with a 3×3 averaging filter prior to resizing. The image is slightly more blurred than (b), but aliasing is not longer objectionable. (Original image courtesy of the Signal Compression Laboratory, University of California, Santa Barbara.)

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

what happens when a newspaper image is sampled at 75 dpi. The sampling lattice (which is oriented vertically and horizontally) and dot patterns on the newspaper image (oriented at ±45°) interact to create a uniform moiré pattern that makes the image look blotchy. (We discuss a technique in Section 4.10.2 for reducing moiré interference patterns.)

As a related point of interest, Fig. 4.22 shows a newspaper image sampled at 400 dpi to avoid moiré effects. The enlargement of the region surrounding the subject's left eye illustrates how halftone dots are used to create shades of gray. The dot size is inversely proportional to image intensity. In light areas, the dots are small or totally absent (see, for example, the white part of the eye). In light gray areas, the dots are larger, as shown below the eye. In darker areas, when dot size exceeds a specified value (typically 50%), dots are allowed to join along two specified directions to form an interconnected mesh (see, for example, the left part of the eye). In some cases the dots join along only one direction, as in the top right area below the eyebrow.

4.5.5 The 2-D Discrete Fourier Transform and Its Inverse

A development similar to the material in Sections 4.3 and 4.4 would yield the following 2-D *discrete Fourier transform* (DFT):

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)} \tag{4.5-15}$$

where $f(x, y)$ is a digital image of size $M \times N$. As in the 1-D case, Eq. (4.5-15) must be evaluated for values of the discrete variables u and v in the ranges $u = 0, 1, 2, \ldots, M-1$ and $v = 0, 1, 2, \ldots, N-1$.[†]

Sometimes you will find in the literature the $1/MN$ constant in front of DFT instead of the IDFT. At times, the constant is expressed as $1/\sqrt{MN}$ and is included in front of the forward and inverse transforms, thus creating a more symmetric pair. Any of these formulations is correct, provided that you are consistent.

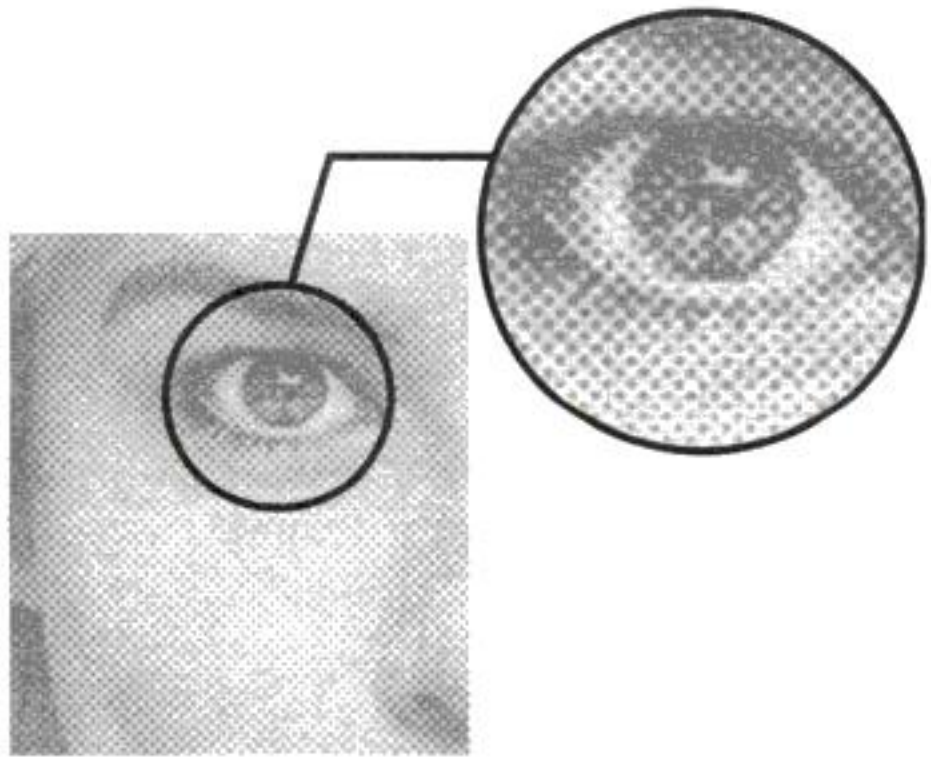


FIGURE 4.22 A newspaper image and an enlargement showing how halftone dots are arranged to render shades of gray.

[†] As mentioned in Section 4.4.1, keep in mind that in this chapter we use (t, z) and (μ, ν) to denote 2-D *continuous* spatial and frequency-domain variables. In the 2-D *discrete* case, we use (x, y) for spatial variables and (u, v) for frequency-domain variables.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

4.6.4 Symmetry Properties

An important result from functional analysis is that any real *or* complex function, $w(x, y)$, can be expressed as the sum of an even and an odd part (*each of which can be real or complex*):

$$w(x, y) = w_e(x, y) + w_o(x, y) \tag{4.6-9}$$

where the even and odd parts are defined as

$$w_e(x, y) \triangleq \frac{w(x, y) + w(-x, -y)}{2} \tag{4.6-10a}$$

and

$$w_o(x, y) \triangleq \frac{w(x, y) - w(-x, -y)}{2} \tag{4.6-10b}$$

Substituting Eqs. (4.6-10a) and (4.6-10b) into Eq. (4.6-9) gives the identity $w(x, y) \equiv w(x, y)$, thus proving the validity of the latter equation. It follows from the preceding definitions that

$$w_e(x, y) = w_e(-x, -y) \tag{4.6-11a}$$

and that

$$w_o(x, y) = -w_o(-x, -y) \tag{4.6-11b}$$

Even functions are said to be *symmetric* and odd functions are *antisymmetric*. Because all indices in the DFT and IDFT are positive, when we talk about symmetry (antisymmetry) we are referring to symmetry (antisymmetry) about the *center point* of a sequence. In terms of Eq. (4.6-11), indices to the right of the center point of a 1-D array are considered positive, and those to the left are considered negative (similarly in 2-D). In our work, it is more convenient to think only in terms of nonnegative indices, in which case the definitions of evenness and oddness become:

$$w_e(x, y) = w_e(M - x, N - y) \tag{4.6-12a}$$

and

$$w_o(x, y) = -w_o(M - x, N - y) \tag{4.6-12b}$$

where, as usual, M and N are the number of rows and columns of a 2-D array.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

EXAMPLE 4.11: 1-D illustrations of properties from Table 4.1.

■ With reference to the even and odd concepts discussed earlier and illustrated in Example 4.10, the following 1-D sequences and their transforms are short examples of the properties listed in Table 4.1. The numbers in parentheses on the right are the individual elements of $F(u)$, and similarly for $f(x)$ in the last two properties.

Property	$f(x)$		$F(u)$
3	$\{1\ \ 2\ \ 3\ \ 4\}$	$\Leftrightarrow$	$\{(10)\ (-2+2j)\ (-2)\ (-2-2j)\}$
4	$j\{1\ \ 2\ \ 3\ \ 4\}$	$\Leftrightarrow$	$\{(2.5j)\ (.5-.5j)\ (-.5j)\ (-.5-.5j)\}$
8	$\{2\ \ 1\ \ 1\ \ 1\}$	$\Leftrightarrow$	$\{(5)\ (1)\ (1)\ (1)\}$
9	$\{0\ \ -1\ \ 0\ \ 1\}$	$\Leftrightarrow$	$\{(0)\ (2j)\ (0)\ (-2j)\}$
10	$j\{2\ \ 1\ \ 1\ \ 1\}$	$\Leftrightarrow$	$\{(5j)\ (j)\ (j)\ (j)\}$
11	$j\{0\ \ -1\ \ 0\ \ 1\}$	$\Leftrightarrow$	$\{(0)\ (-2)\ (0)\ (2)\}$
12	$\{(4+4j)\ (3+2j)\ (0+2j)\ (3+2j)\}$	$\Leftrightarrow$	$\{(10+10j)\ (4+2j)\ (-2+2j)\ (4+2j)\}$
13	$\{(0+0j)\ (1+1j)\ (0+0j)\ (-1-j)\}$	$\Leftrightarrow$	$\{(0+0j)\ (2-2j)\ (0+0j)\ (-2+2j)\}$

For example, in property 3 we see that a real function with elements $\{1\ \ 2\ \ 3\ \ 4\}$ has Fourier transform whose real part, $\{10\ \ -2\ \ -2\ \ -2\}$, is even and whose imaginary part, $\{0\ \ 2\ \ 0\ \ -2\}$, is odd. Property 8 tells us that a real even function has a transform that is real and even also. Property 12 shows that an even complex function has a transform that is also complex and even. The other property examples are analyzed in a similar manner. ■

EXAMPLE 4.12: Proving several symmetry properties of the DFT from Table 4.1.

■ In this example, we prove several of the properties in Table 4.1 to develop familiarity with manipulating these important properties, and to establish a basis for solving some of the problems at the end of the chapter. We prove only the properties on the right given the properties on the left. The converse is proved in a manner similar to the proofs we give here.

Consider property 3, which reads: If $f(x, y)$ is a real function, the real part of its DFT is even and the odd part is odd; similarly, if a DFT has real and imaginary parts that are even and odd, respectively, then its IDFT is a real function. We prove this property formally as follows. $F(u, v)$ is complex in general, so it can be expressed as the sum of a real and an imaginary part: $F(u, v) = R(u, v) + jI(u, v)$. Then, $F^*(u, v) = R(u, v) - jI(u, v)$. Also, $F(-u, -v) = R(-u, -v) + jI(-u, -v)$. But, as proved earlier, if $f(x, y)$ is real then $F^*(u, v) = F(-u, -v)$, which, based on the preceding two equations, means that $R(u, v) = R(-u, -v)$ and $I(u, v) = -I(-u, -v)$. In view of Eqs. (4.6-11a) and (4.6-11b), this proves that R is an even function and I is an odd function.

Next, we prove property 8. If $f(x, y)$ is real we know from property 3 that the real part of $F(u, v)$ is even, so to prove property 8 all we have to do is show that if $f(x, y)$ is real *and* even then the imaginary part of $F(u, v)$ is 0 (i.e., F is real). The steps are as follows:

$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux/M + vy/N)}$$

which we can write as

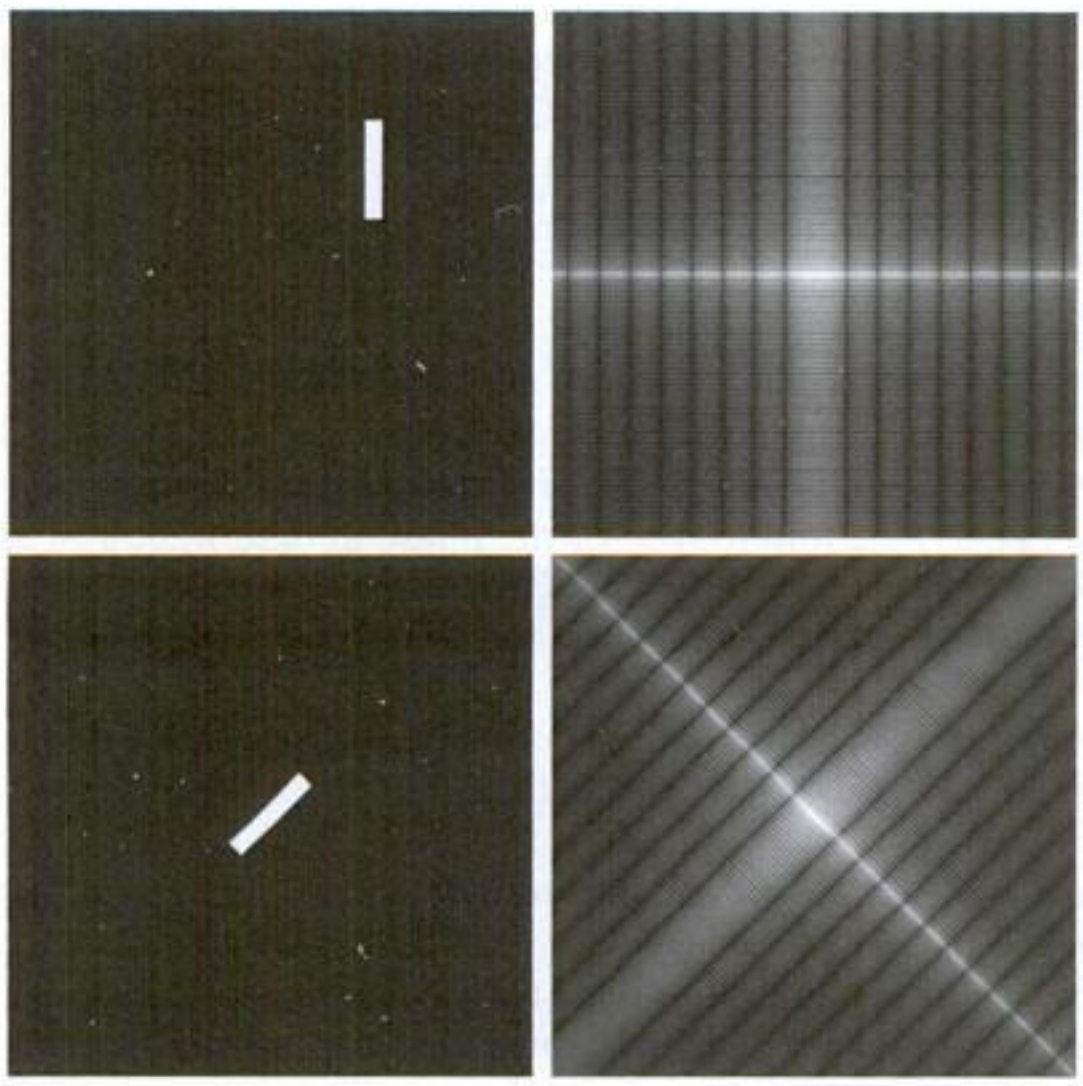
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

transform contains the highest values (and thus appears brighter in the image). However, note that the four corners of the spectrum contain similarly high values. The reason is the periodicity property discussed in the previous section. To center the spectrum, we simply multiply the image in (a) by $(-1)^{x+y}$ before computing the DFT, as indicated in Eq. (4.6-8). Figure 4.22(c) shows the result, which clearly is much easier to visualize (note the symmetry about the center point). Because the dc term dominates the values of the spectrum, the dynamic range of other intensities in the displayed image are compressed. To bring out those details, we perform a log transformation, as described in Section 3.2.2. Figure 4.24(d) shows the display of $(1 + \log|F(u, v)|)$. The increased rendition of detail is evident. Most spectra shown in this and subsequent chapters are scaled in this manner.

It follows from Eqs. (4.6-4) and (4.6-5) that the spectrum is insensitive to image translation (the absolute value of the exponential term is 1), but it rotates by the same angle of a rotated image. Figure 4.25 illustrates these properties. The spectrum in Fig. 4.25(b) is identical to the spectrum in Fig. 4.24(d). Clearly, the images in Figs. 4.24(a) and 4.25(a) are different, so if their Fourier spectra are the same then, based on Eq. (4.6-15), their phase angles must be different. Figure 4.26 confirms this. Figures 4.26(a) and (b) are the phase angle arrays (shown as images) of the DFTs of Figs. 4.24(a) and 4.25(a). Note the lack of similarity between the phase images, in spite of the fact that the only differences between their corresponding images is simple translation. In general, visual analysis of phase angle images yields little intuitive information. For instance, due to its 45° orientation, one would expect intuitively that the phase angle in



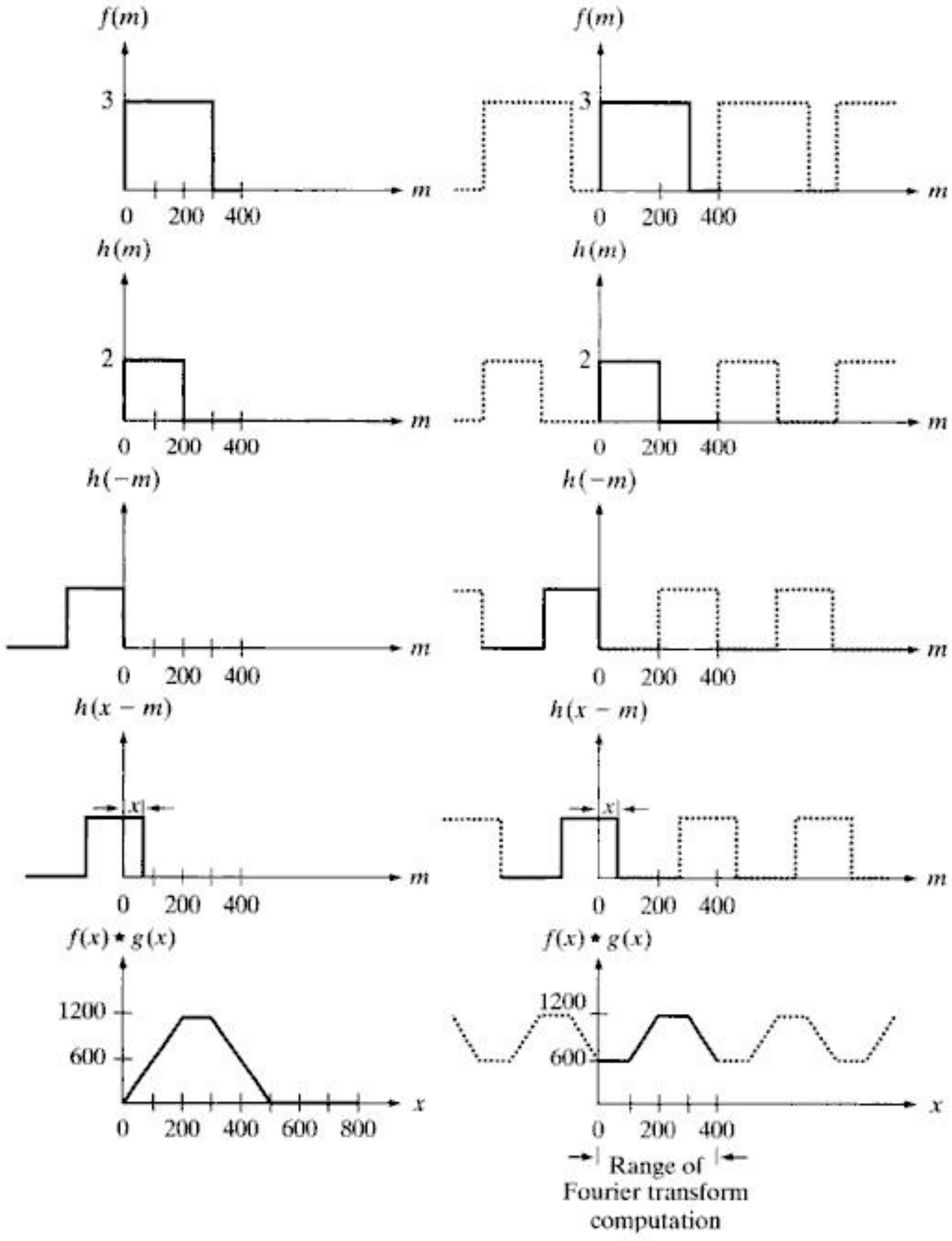
a b
c d

FIGURE 4.25
(a) The rectangle in Fig. 4.24(a) translated, and (b) the corresponding spectrum. (c) Rotated rectangle, and (d) the corresponding spectrum. The spectrum corresponding to the translated rectangle is identical to the spectrum corresponding to the original image in Fig. 4.24(a).

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a f
b g
c h
d i
e j

FIGURE 4.28 Left column: convolution of two discrete functions obtained using the approach discussed in Section 3.4.2. The result in (e) is correct. Right column: Convolution of the same functions, but taking into account the periodicity implied by the DFT. Note in (j) how data from adjacent periods produce wraparound error, yielding an incorrect convolution result. To obtain the correct result, function padding must be used.

two transforms, and then computed the inverse DFT, we would have obtained the erroneous 400-point segment of the convolution shown in Fig. 4.28(j).

Fortunately, the solution to the wraparound error problem is simple. Consider two functions, $f(x)$ and $h(x)$ composed of A and B samples, respectively. It can be shown (Brigham [1988]) that if we append zeros to both functions so that they have the same length, denoted by P, then wraparound is avoided by choosing

$$P \geq A + B - 1 \tag{4.6-26}$$

In our example, each function has 400 points, so the minimum value we could use is $P = 799$, which implies that we would append 399 zeros to the trailing edge of each function. This process is called *zero padding*. As an exercise, you

The zeros could be appended also to the beginning of the functions, or they could be divided between the beginning and end of the functions. It is simpler to append them at the end.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

TABLE 4.3
(*Continued*)

	Name	DFT Pairs
7)	Correlation theorem†	$f(x, y) \star h(x, y) \Leftrightarrow F^*(u, v)H(u, v)$ $f^*(x, y)h(x, y) \Leftrightarrow F(u, v) \star H(u, v)$
8)	Discrete unit impulse	$\delta(x, y) \Leftrightarrow 1$
9)	Rectangle	$\text{rect}[a, b] \Leftrightarrow ab\dfrac{\sin(\pi ua)}{(\pi ua)}\dfrac{\sin(\pi vb)}{(\pi vb)}e^{-j\pi(ua+vb)}$
10)	Sine	$\sin(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $j\dfrac{1}{2}\left[\delta(u + Mu_0, v + Nv_0) - \delta(u - Mu_0, v - Nv_0)\right]$
11)	Cosine	$\cos(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow$ $\dfrac{1}{2}\left[\delta(u + Mu_0, v + Nv_0) + \delta(u - Mu_0, v - Nv_0)\right]$
	The following Fourier transform pairs are derivable only for continuous variables, denoted as before by t and z for spatial variables and by μ and ν for frequency variables. These results can be used for DFT work by sampling the continuous forms.	
12)	*Differentiation* (The expressions on the right assume that $f(\pm\infty, \pm\infty) = 0$.)	$\left(\dfrac{\partial}{\partial t}\right)^m\left(\dfrac{\partial}{\partial z}\right)^n f(t, z) \Leftrightarrow (j2\pi\mu)^m(j2\pi\nu)^n F(\mu, \nu)$ $\dfrac{\partial^m f(t, z)}{\partial t^m} \Leftrightarrow (j2\pi\mu)^m F(\mu, \nu); \dfrac{\partial^n f(t, z)}{\partial z^n} \Leftrightarrow (j2\pi\nu)^n F(\mu, \nu)$
13)	*Gaussian*	$A2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2+z^2)} \Leftrightarrow Ae^{-(\mu^2+\nu^2)/2\sigma^2}$ (A is a constant)

†Assumes that the functions have been extended by zero padding. Convolution and correlation are associative, commutative, and distributive.

be used to derive the frequency-domain equivalent of the Laplacian defined in Eq. (3.6-3) (Problem 4.26). The Gaussian pair is discussed in Section 4.7.4.

Tables 4.1 through 4.3 provide a summary of properties useful when working with the DFT. Many of these properties are key elements in the development of the material in the rest of this chapter, and some are used in subsequent chapters.

4.7 The Basics of Filtering in the Frequency Domain

In this section, we lay the groundwork for all the filtering techniques discussed in the remainder of the chapter.

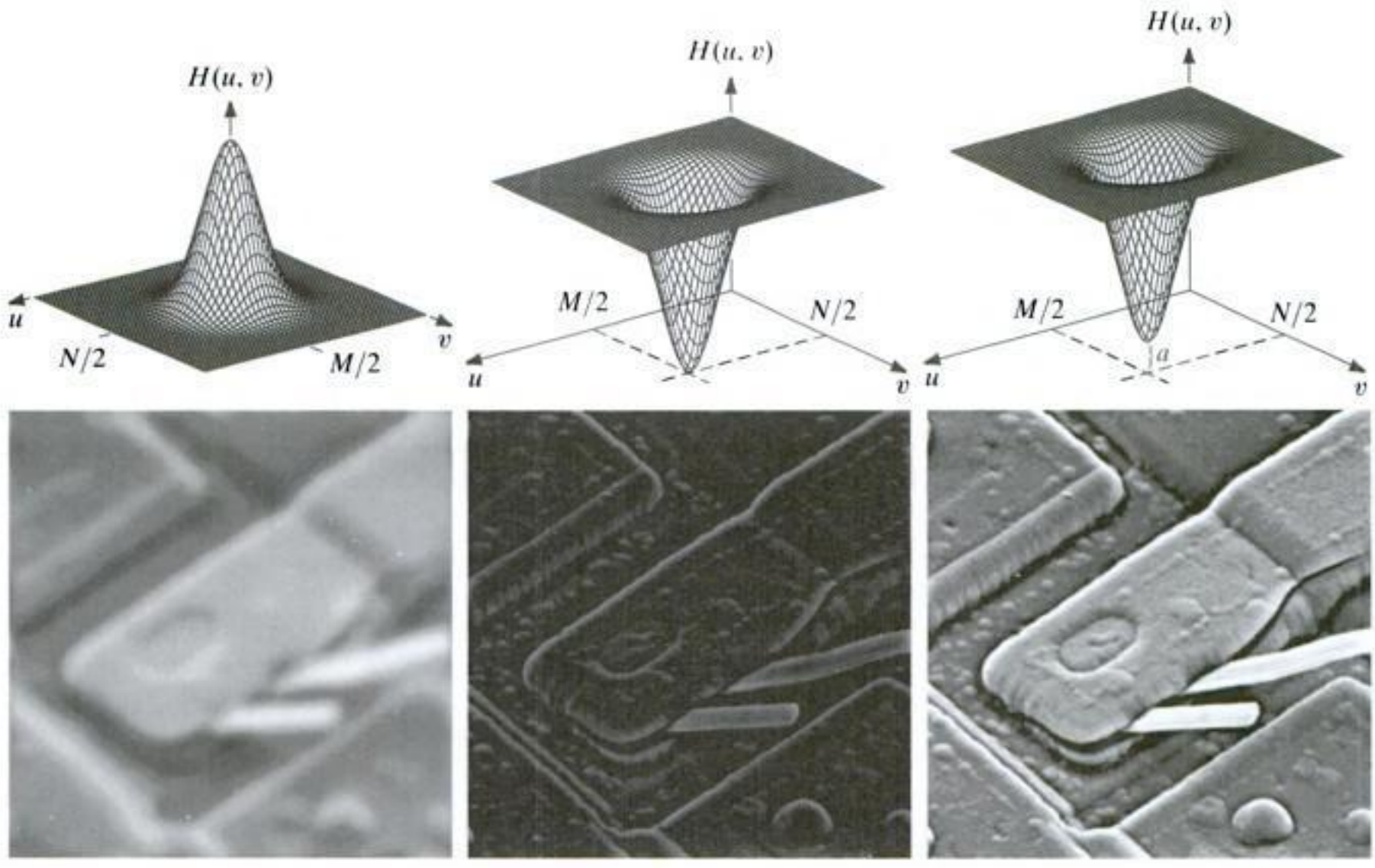
4.7.1 Additional Characteristics of the Frequency Domain

We begin by observing in Eq. (4.5-15) that *each* term of $F(u, v)$ contains *all* values of $f(x, y)$, modified by the values of the exponential terms. Thus, with the exception of trivial cases, it usually is impossible to make direct associations between specific components of an image and its transform. However, some general statements can be made about the relationship between the frequency

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a b c
d e f

FIGURE 4.31 Top row: frequency domain filters. Bottom row: corresponding filtered images obtained using Eq. (4.7-1). We used $a = 0.85$ in (c) to obtain (f) (the height of the filter itself is 1). Compare (f) with Fig. 4.29(a).

a b c

FIGURE 4.32 (a) A simple image. (b) Result of blurring with a Gaussian lowpass filter without padding. (c) Result of lowpass filtering with padding. Compare the light area of the vertical edges in (b) and (c).

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

$|F(u, v)|$, and then computing the IDFT. The basic shapes remain unchanged, but the intensity distribution is quite distorted. Figure 4.35(b) shows the result of multiplying the phase by 0.25. The image is almost unrecognizable.

4.7.3 Summary of Steps for Filtering in the Frequency Domain

The material in the previous two sections can be summarized as follows:

1. Given an input image $f(x, y)$ of size $M \times N$, obtain the padding parameters P and Q from Eqs. (4.6-31) and (4.6-32). Typically, we select $P = 2M$ and $Q = 2N$.
2. Form a padded image, $f_p(x, y)$, of size $P \times Q$ by appending the necessary number of zeros to $f(x, y)$.
3. Multiply $f_p(x, y)$ by $(-1)^{x+y}$ to center its transform.
4. Compute the DFT, $F(u, v)$, of the image from step 3.
5. Generate a real, symmetric filter function, $H(u, v)$, of size $P \times Q$ with center at coordinates $(P/2, Q/2)$.[†] Form the product $G(u, v) = H(u, v)F(u, v)$ using array multiplication; that is, $G(i, k) = H(i, k)F(i, k)$.
6. Obtain the processed image:

$$g_p(x, y) = \left\{\text{real}\left[\Im^{-1}[G(u, v)]\right]\right\}(-1)^{x+y}$$

where the real part is selected in order to ignore parasitic complex components resulting from computational inaccuracies, and the subscript p indicates that we are dealing with padded arrays.

7. Obtain the final processed result, $g(x, y)$, by extracting the $M \times N$ region from the top, left quadrant of $g_p(x, y)$.

As noted earlier, centering helps in visualizing the filtering process and in generating the filter functions themselves, but centering is not a fundamental requirement.

Figure 4.36 illustrates the preceding steps. The legend in the figure explains the source of each image. If it were enlarged, Fig. 4.36(c) would show black dots interleaved in the image because negative intensities are clipped to 0 for display. Note in Fig. 4.36(h) the characteristic dark border exhibited by lowpass filtered images processed using zero padding.

4.7.4 Correspondence Between Filtering in the Spatial and Frequency Domains

The link between filtering in the spatial and frequency domains is the convolution theorem. In Section 4.7.2, we defined filtering in the frequency domain as the multiplication of a filter function, $H(u, v)$, times $F(u, v)$, the Fourier transform of the input image. Given a filter $H(u, v)$, suppose that we want to find its equivalent representation in the spatial domain. If we let $f(x, y) = \delta(x, y)$, it follows from Table 4.3 that $F(u, v) = 1$. Then, from Eq. (4.7-1), the filtered output is $\Im^{-1}\{H(u, v)\}$. But this is the inverse transform of the frequency domain filter, which is the corresponding filter in the

[†] If $H(u, v)$ is to be generated from a *given* spatial filter, $h(x, y)$, then we form $h_p(x, y)$ by padding the spatial filter to size $P \times Q$, multiply the expanded array by $(-1)^{x+y}$, and compute the DFT of the result to obtain a centered $H(u, v)$. Example 4.15 illustrates this procedure.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

a b c

FIGURE 4.56 Results of highpass filtering the image in Fig. 4.41(a) using a GHPF with $D_0 = 30$, 60, and 160, corresponding to the circles in Fig. 4.41(b). Compare with Figs. 4.54 and 4.55.

order 2 and with D_0 set to the same values as in Fig. 4.54. The boundaries are much less distorted than in Fig. 4.54, even for the smallest value of cutoff frequency. Because the spot sizes in the center areas of the IHPF and the BHPF are similar [see Figs. 4.53(a) and (b)], the performance of the two filters on the smaller objects is comparable. The transition into higher values of cutoff frequencies is much smoother with the BHPF.

4.9.3 Gaussian Highpass Filters

The transfer function of the Gaussian highpass filter (GHPF) with cutoff frequency locus at a distance D_0 from the center of the frequency rectangle is given by

$$H(u, v) = 1 - e^{-D^2(u,v)/2D_0^2} \tag{4.9-4}$$

where $D(u, v)$ is given by Eq. (4.8-2). This expression follows directly from Eqs. (4.8-7) and (4.9-1). The third row in Fig. 4.52 shows a perspective plot, image, and cross section of the GHPF function. Following the same format as for the BHPF, we show in Fig. 4.56 comparable results using GHPFs. As expected, the results obtained are more gradual than with the previous two filters. Even the filtering of the smaller objects and thin bars is cleaner with the Gaussian filter. Table 4.5 contains a summary of the highpass filters discussed in this section.

TABLE 4.5
Highpass filters. D_0 is the cutoff frequency and n is the order of the Butterworth filter.

Ideal	Butterworth	Gaussian
$H(u, v) = \begin{cases} 1 & \text{if } D(u, v) \le D_0 \\ 0 & \text{if } D(u, v) > D_0 \end{cases}$	$H(u, v) = \dfrac{1}{1 + [D_0/D(u, v)]^{2n}}$	$H(u, v) = 1 - e^{-D^2(u,v)/2D_0^2}$

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

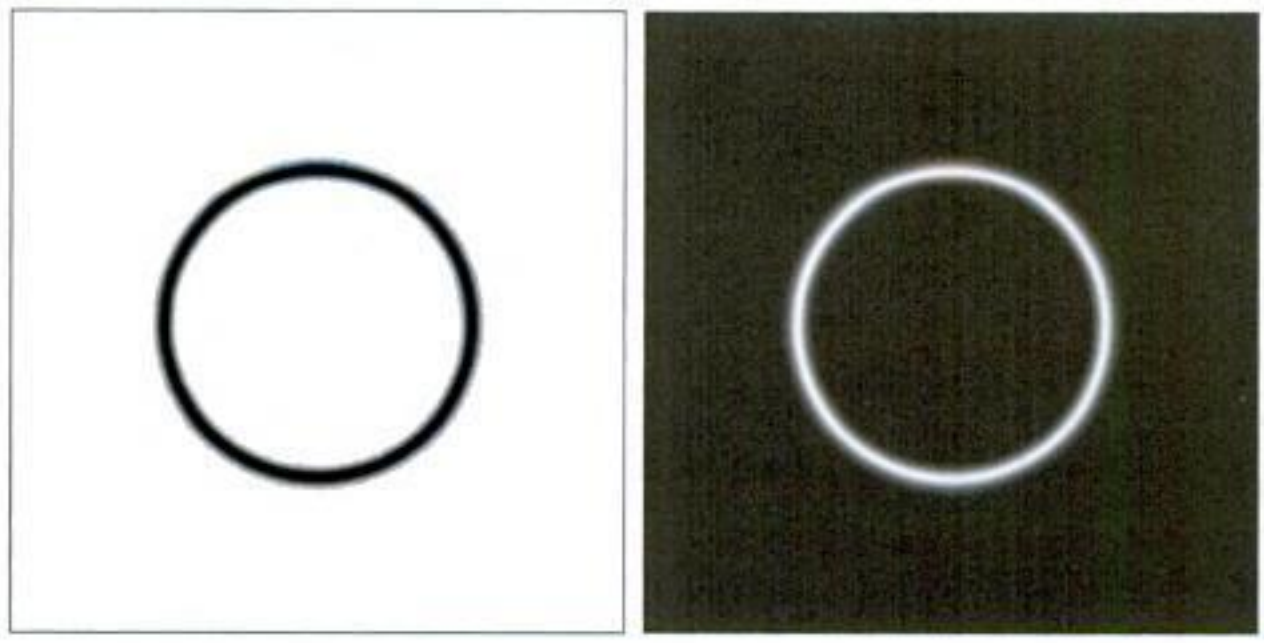
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a b

FIGURE 4.63
(a) Bandreject Gaussian filter. (b) Corresponding bandpass filter. The thin black border in (a) was added for clarity; it is not part of the data.

center of the frequency rectangle, $(M/2, N/2)$. The distance computations for each filter are thus carried out using the expressions

$$D_k(u, v) = \left[(u - M/2 - u_k)^2 + (v - N/2 - v_k)^2\right]^{1/2} \qquad (4.10\text{-}3)$$

and

$$D_{-k}(u, v) = \left[(u - M/2 + u_k)^2 + (v - N/2 + v_k)^2\right]^{1/2} \qquad (4.10\text{-}4)$$

For example, the following is a Butterworth notch reject filter of order n, containing three notch pairs:

$$H_{\mathrm{NR}}(u, v) = \prod_{k=1}^{3}\left[\frac{1}{1 + [D_{0k}/D_k(u, v)]^{2n}}\right]\left[\frac{1}{1 + [D_{0k}/D_{-k}(u, v)]^{2n}}\right] \qquad (4.10\text{-}5)$$

where D_k and D_{-k} are given by Eqs. (4.10-3) and (4.10-4). The constant D_{0k} is the same for each pair of notches, but it can be different for different pairs. Other notch reject filters are constructed in the same manner, depending on the highpass filter chosen. As with the filters discussed earlier, a *notch pass filter* is obtained from a notch reject filter using the expression

$$H_{\mathrm{NP}}(u, v) = 1 - H_{\mathrm{NR}}(u, v) \qquad (4.10\text{-}6)$$

As the next three examples show, one of the principal applications of notch filtering is for selectively modifying local regions of the DFT. This type of processing typically is done interactively, working directly on DFTs obtained without padding. The advantages of working interactively with actual DFTs (as opposed to having to "translate" from padded to actual frequency values) outweigh any wraparound errors that may result from not using padding in the filtering process. Also, as we show in Section 5.4.4, even more powerful notch filtering techniques than those discussed here are based on unpadded DFTs. To get an idea of how DFT values change as a function of padding, see Problem 4.22.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

For each value of x and for $v = 0, 1, 2, \ldots, N - 1$, we see that $F(x, v)$ is simply the 1-D DFT of a row of $f(x, y)$. By varying x from 0 to $M - 1$ in Eq. (4.11-2), we compute a set of 1-D DFTs for all rows of $f(x, y)$. The computations in Eq. (4.11-1) similarly are 1-D transforms of the columns of $F(x, v)$.

Thus, we conclude that the 2-D DFT of $f(x, y)$ can be obtained by computing the 1-D transform of each row of $f(x, y)$ and then computing the 1-D transform along each column of the result. This is an important simplification because we have to deal only with one variable at a time. A similar development applies to computing the 2-D IDFT using the 1-D IDFT. However, as we show in the following section, we can compute the IDFT using an algorithm designed to compute the DFT.

We could have expressed Eq. (4.11-1) and (4.11-2) in the form of 1-D column transforms followed by row transforms. The final result would have been the same.

4.11.2 Computing the IDFT Using a DFT Algorithm

Taking the complex conjugate of both sides of Eq. (4.5-16) and multiplying the results by MN yields

$$MNf^*(x, y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F^*(u, v) e^{-j2\pi(ux/M + vy/N)} \tag{4.11-3}$$

Multiplication by MN in this development assumes the forms in Eqs. (4.5-15) and (4.5-16). A different constant multiplication scheme is required if the constants are distributed differently between the forward and inverse transforms.

But, we recognize the form of the right side of this result as the DFT of $F^*(u, v)$. Therefore, Eq. (4.11-3) indicates that if we substitute $F^*(u, v)$ into an algorithm designed to compute the 2-D forward Fourier transform, the result will be $MNf^*(x, y)$. Taking the complex conjugate and multiplying this result by MN yields $f(x, y)$, which is the inverse of $F(u, v)$.

Computing the 2-D inverse from a 2-D forward DFT algorithm that is based on successive passes of 1-D transforms (as in the previous section) is a frequent source of confusion involving the complex conjugates and multiplication by a constant, neither of which is done in the 1-D algorithms. The key concept to keep in mind is that we simply input $F^*(u, v)$ into whatever forward algorithm we have. The result will be $MNf^*(x, y)$. All we have to do with this result to obtain $f(x, y)$ is to take its complex conjugate and multiply it by the constant MN. Of course, when $f(x, y)$ is real, as typically is the case, $f^*(x, y) = f(x, y)$.

4.11.3 The Fast Fourier Transform (FFT)

Work in the frequency domain would not be practical if we had to implement Eqs. (4.5-15) and (4.5-16) directly. Brute-force implementation of these equations requires on the order of $(MN)^2$ summations and additions. For images of moderate size (say, 1024×1024 pixels), this means on the order of a trillion multiplications and additions for just one DFT, excluding the exponentials, which could be computed once and stored in a look-up table. This would be a challenge even for super computers. Without the discovery of the *fast Fourier transform* (FFT), which reduces computations to the order of $MN \log_2 MN$ multiplications and additions, it is safe to say that the material presented in this chapter would be of little practical value. The computational reductions afforded by the FFT are impressive indeed. For example, computing the 2-D FFT of a 1024×1024 image would require on the order of 20 million multiplication and additions, which is a significant reduction from the one trillion computations mentioned above.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

5 Image Restoration and Reconstruction

Things which we see are not by themselves what we see. . . .
It remains completely unknown to us what the objects may be by themselves and apart from the receptivity of our senses. We know nothing but our manner of perceiving them.

Immanuel Kant

Preview

As in image enhancement, the principal goal of restoration techniques is to improve an image in some predefined sense. Although there are areas of overlap, image enhancement is largely a subjective process, while image restoration is for the most part an objective process. Restoration attempts to recover an image that has been degraded by using a priori knowledge of the degradation phenomenon. Thus, restoration techniques are oriented toward modeling the degradation and applying the inverse process in order to recover the original image.

This approach usually involves formulating a criterion of goodness that will yield an optimal estimate of the desired result. By contrast, enhancement techniques basically are heuristic procedures designed to manipulate an image in order to take advantage of the psychophysical aspects of the human visual system. For example, contrast stretching is considered an enhancement technique because it is based primarily on the pleasing aspects it might present to the viewer, whereas removal of image blur by applying a deblurring function is considered a restoration technique.

The material developed in this chapter is strictly introductory. We consider the restoration problem only from the point where a degraded, *digital* image is given; thus we consider topics dealing with sensor, digitizer, and display degradations only superficially. These subjects, although of importance in the overall treatment of image restoration applications, are beyond the scope of the present discussion.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

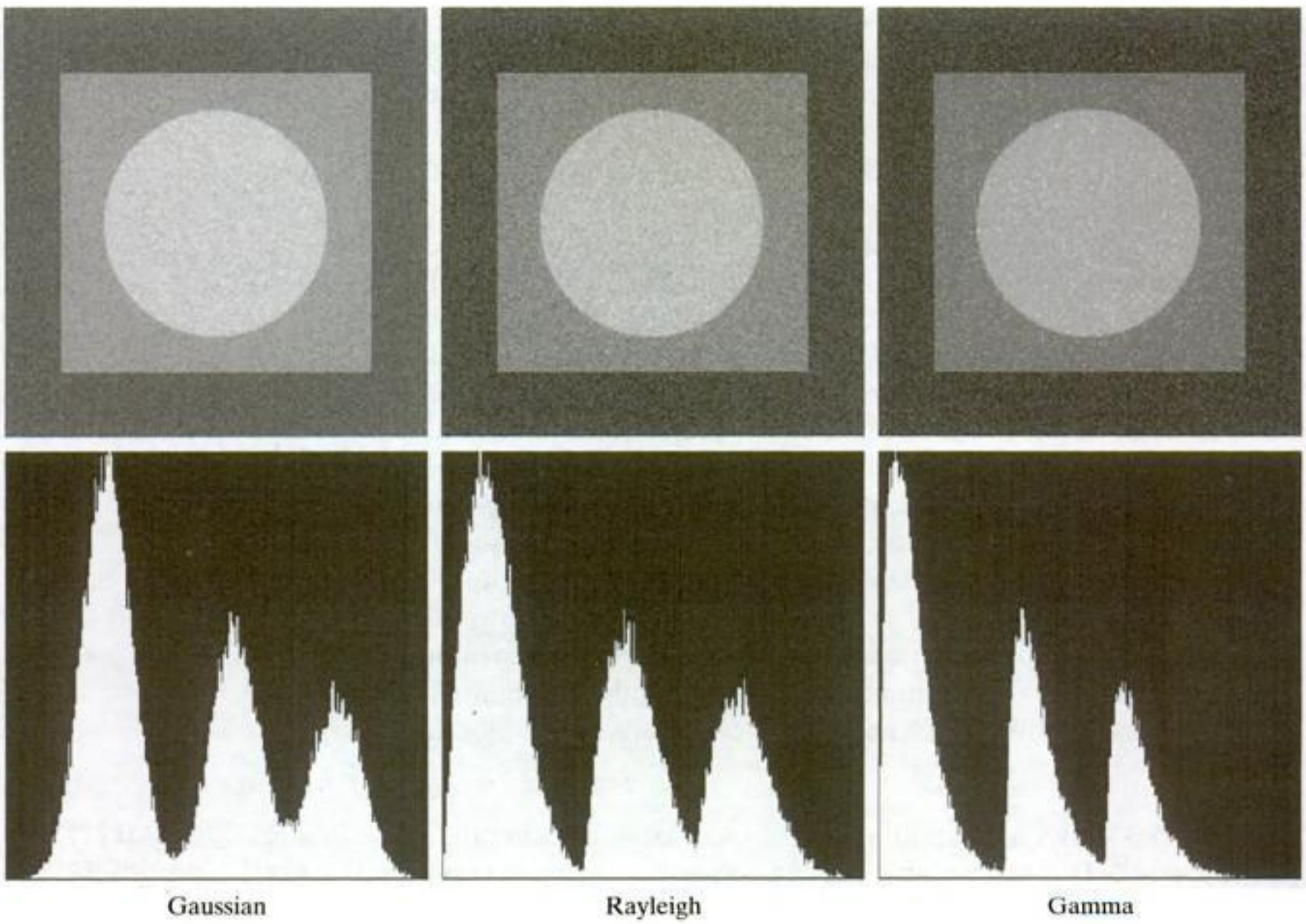
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a b c
d e f

FIGURE 5.4 Images and histograms resulting from adding Gaussian, Rayleigh, and gamma noise to the image in Fig. 5.3.

We see a close correspondence in comparing the histograms in Fig. 5.4 with the PDFs in Fig. 5.2. The histogram for the salt-and-pepper example has an extra peak at the white end of the intensity scale because the noise components were pure black and white, and the lightest component of the test pattern (the circle) is light gray. With the exception of slightly different overall intensity, it is difficult to differentiate visually between the first five images in Fig. 5.4, even though their histograms are significantly different. The salt-and-pepper appearance of the image corrupted by impulse noise is the only one that is visually indicative of the type of noise causing the degradation. ■

5.2.3 Periodic Noise

Periodic noise in an image arises typically from electrical or electromechanical interference during image acquisition. This is the only type of spatially dependent noise that will be considered in this chapter. As discussed in Section 5.4, periodic noise can be reduced significantly via frequency domain filtering. For example, consider the image in Fig. 5.5(a). This image is severely corrupted by (spatial) sinusoidal noise of various frequencies. The Fourier transform of a pure

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

5.3 Restoration in the Presence of Noise Only—Spatial Filtering

When the only degradation present in an image is noise, Eqs. (5.1-1) and (5.1-2) become

$$g(x, y) = f(x, y) + \eta(x, y) \tag{5.3-1}$$

and

$$G(u, v) = F(u, v) + N(u, v) \tag{5.3-2}$$

The noise terms are unknown, so subtracting them from $g(x, y)$ or $G(u, v)$ is not a realistic option. In the case of periodic noise, it usually is possible to estimate $N(u, v)$ from the spectrum of $G(u, v)$, as noted in Section 5.2.3. In this case $N(u, v)$ can be subtracted from $G(u, v)$ to obtain an estimate of the original image. In general, however, this type of knowledge is the exception, rather than the rule.

Spatial filtering is the method of choice in situations when only additive random noise is present. Spatial filtering is discussed in detail in Chapter 3. With the exception of the nature of the computation performed by a specific filter, the mechanics for implementing all the filters that follow are exactly as discussed in Sections 3.4 through 3.6.

5.3.1 Mean Filters

In this section we discuss briefly the noise-reduction capabilities of the spatial filters introduced in Section 3.5 and develop several other filters whose performance is in many cases superior to the filters discussed in that section.

Arithmetic mean filter

We assume that m and n are odd integers.

This is the simplest of the mean filters. Let S_{xy} represent the set of coordinates in a rectangular subimage window (neighborhood) of size $m \times n$, centered at point (x, y). The arithmetic mean filter computes the average value of the corrupted image $g(x, y)$ in the area defined by S_{xy}. The value of the restored image $\hat{f}$ at point (x, y) is simply the arithmetic mean computed using the pixels in the region defined by S_{xy}. In other words,

$$\hat{f}(x, y) = \frac{1}{mn} \sum_{(s,t) \in S_{xy}} g(s, t) \tag{5.3-3}$$

This operation can be implemented using a spatial filter of size $m \times n$ in which all coefficients have value $1/mn$. A mean filter smooths local variations in an image, and noise is reduced as a result of blurring.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

An adaptive expression for obtaining $\hat{f}(x, y)$ based on these assumptions may be written as

$$\hat{f}(x, y) = g(x, y) - \frac{\sigma_\eta^2}{\sigma_L^2}\left[g(x, y) - m_L\right] \tag{5.3-12}$$

The only quantity that needs to be known or estimated is the variance of the overall noise, σ_η^2. The other parameters are computed from the pixels in S_{xy} at each location (x, y) on which the filter window is centered. A tacit assumption in Eq. (5.3-12) is that $\sigma_\eta^2 \leq \sigma_L^2$. The noise in our model is additive and position independent, so this is a reasonable assumption to make because S_{xy} is a subset of $g(x, y)$. However, we seldom have exact knowledge of σ_η^2. Therefore, it is possible for this condition to be violated in practice. For that reason, a test should be built into an implementation of Eq. (5.3-12) so that the ratio is set to 1 if the condition $\sigma_\eta^2 > \sigma_L^2$ occurs. This makes this filter nonlinear. However, it prevents nonsensical results (i.e., negative intensity levels, depending on the value of m_L) due to a potential lack of knowledge about the variance of the image noise. Another approach is to allow the negative values to occur, and then rescale the intensity values at the end. The result then would be a loss of dynamic range in the image.

EXAMPLE 5.4: Illustration of adaptive, local noise-reduction filtering.

■ Figure 5.13(a) shows the circuit-board image, corrupted this time by additive Gaussian noise of zero mean and a variance of 1000. This is a significant level of noise corruption, but it makes an ideal test bed on which to compare relative filter performance. Figure 5.13(b) is the result of processing the noisy image with an arithmetic mean filter of size 7×7. The noise was smoothed out, but at the cost of significant blurring in the image. Similar comments are applicable to Fig. 5.13(c), which shows the result of processing the noisy image with a geometric mean filter, also of size 7×7. The differences between these two filtered images are analogous to those we discussed in Example 5.2; only the degree of blurring is different.

Figure 5.13(d) shows the result of using the adaptive filter of Eq. (5.3-12) with $\sigma_\eta^2 = 1000$. The improvements in this result compared with the two previous filters are significant. In terms of overall noise reduction, the adaptive filter achieved results similar to the arithmetic and geometric mean filters. However, the image filtered with the adaptive filter is much sharper. For example, the connector fingers at the top of the image are significantly sharper in Fig. 5.13(d). Other features, such as holes and the eight legs of the dark component on the lower left-hand side of the image, are much clearer in Fig. 5.13(d). These results are typical of what can be achieved with an adaptive filter. As mentioned earlier, the price paid for the improved performance is additional filter complexity.

The preceding results used a value for σ_η^2 that matched the variance of the noise exactly. If this quantity is not known and an estimate is used that is too low, the algorithm will return an image that closely resembles the original because the corrections will be smaller than they should be. Estimates that are too high

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

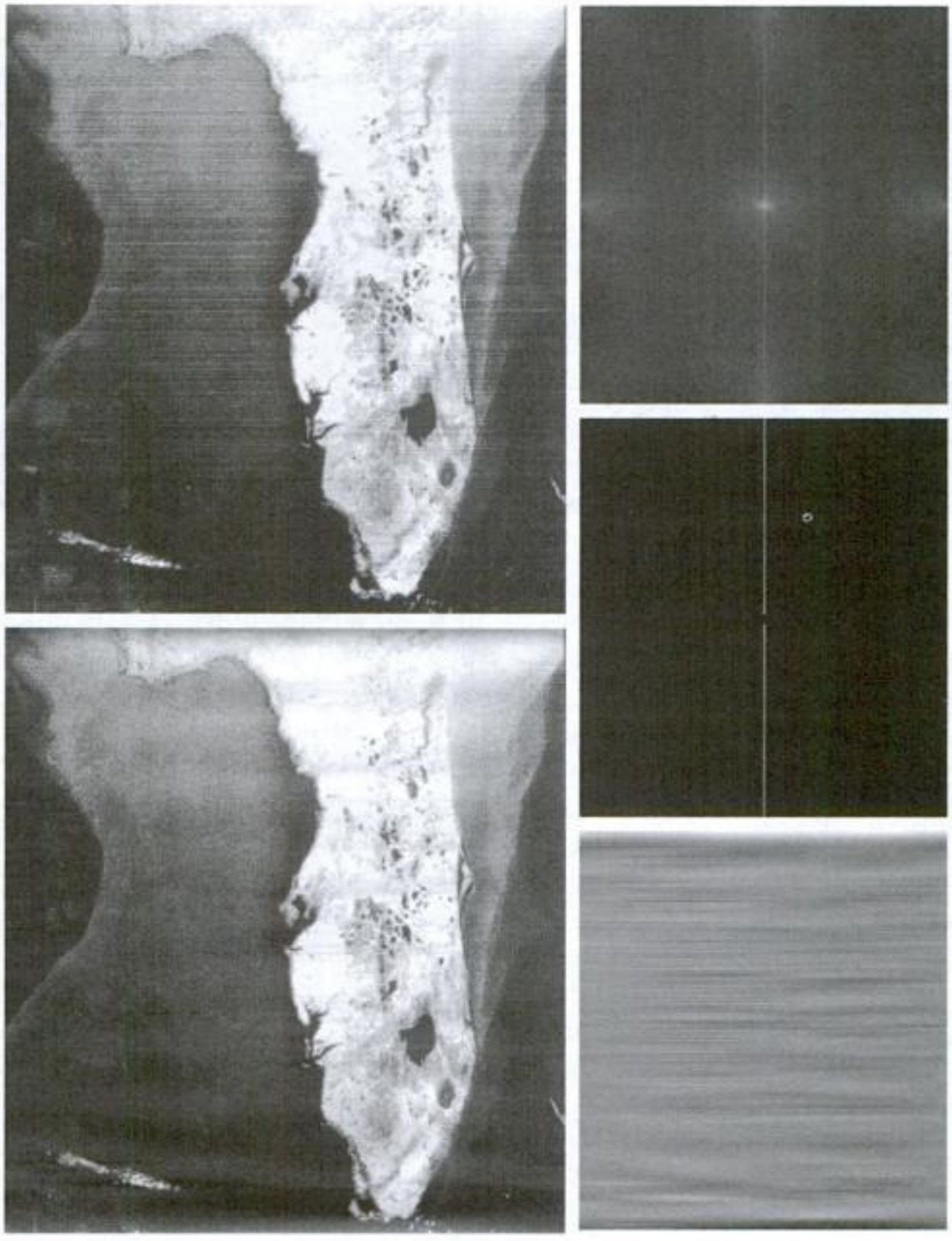
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a b
c
e d

FIGURE 5.19
(a) Satellite image of Florida and the Gulf of Mexico showing horizontal scan lines. (b) Spectrum. (c) Notch pass filter superimposed on (b). (d) Spatial noise pattern. (e) Result of notch reject filtering. (Original image courtesy of NOAA.)

question. The starlike components were caused by the interference, and several pairs of components are present, indicating that the pattern contains more than just one sinusoidal component.

When several interference components are present, the methods discussed in the preceding sections are not always acceptable because they may remove too much image information in the filtering process (a highly undesirable feature when images are unique and/or expensive to acquire). In addition, the interference components generally are not single-frequency bursts. Instead, they tend to have broad skirts that carry information about the interference pattern. These skirts are not always easily detectable from the normal transform background. Alternative filtering methods that reduce the effect of

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

optics, the impulse becomes a point of light and $h(x, \alpha, y, \beta)$ is commonly referred to as the *point spread function* (PSF). This name arises from the fact that all physical optical systems blur (spread) a point of light to some degree, with the amount of blurring being determined by the quality of the optical components.

Substituting Eq. (5.5-10) into Eq. (5.5-9) yields the expression

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta)h(x, \alpha, y, \beta)\, d\alpha\, d\beta \tag{5.5-11}$$

which is called the *superposition* (or *Fredholm*) *integral of the first kind*. This expression is a fundamental result that is at the core of linear system theory. It states that if the response of H to an impulse is known, the response to *any* input $f(\alpha, \beta)$ can be calculated by means of Eq. (5.5-11). In other words, a linear system H is completely characterized by its impulse response.

If H is position invariant, then, from Eq. (5.5-5),

$$H[\delta(x - \alpha, y - \beta)] = h(x - \alpha, y - \beta) \tag{5.5-12}$$

Equation (5.5-11) reduces in this case to

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta)h(x - \alpha, y - \beta)\, d\alpha\, d\beta \tag{5.5-13}$$

This expression is the *convolution integral* introduced for one variable in Eq. (4.2-20) and extended to 2-D in Problem 4.11. This integral tells us that knowing the impulse response of a linear system allows us to compute its response, g, to *any* input f. The result is simply the convolution of the impulse response and the input function.

In the presence of additive noise, the expression of the linear degradation model [Eq. (5.5-11)] becomes

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta)h(x, \alpha, y, \beta)\, d\alpha\, d\beta + \eta(x, y) \tag{5.5-14}$$

If H is position invariant, Eq. (5.5-14) becomes

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta)h(x - \alpha, y - \beta)\, d\alpha\, d\beta + \eta(x, y) \tag{5.5-15}$$

The values of the noise term $\eta(x, y)$ are random, and are assumed to be independent of position. Using the familiar notation for convolution, we can write Eq. (5.5-15) as

$$g(x, y) = h(x, y) \star f(x, y) + \eta(x, y) \tag{5.5-16}$$

or, based on the convolution theorem (see Section 4.6.6), we can express it in the frequency domain as

$$G(u, v) = H(u, v)F(u, v) + N(u, v) \tag{5.5-17}$$

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

noise into the restoration process. The method is founded on considering images and noise as random variables, and the objective is to find an estimate $\hat{f}$ of the uncorrupted image f such that the mean square error between them is minimized. This error measure is given by

Note that entire images are being considered random variables, as discussed at the end of Section 2.6.8.

$$e^2 = E\{(f - \hat{f})^2\} \tag{5.8-1}$$

where $E\{\cdot\}$ is the expected value of the argument. It is assumed that the noise and the image are uncorrelated; that one or the other has zero mean; and that the intensity levels in the estimate are a linear function of the levels in the degraded image. Based on these conditions, the minimum of the error function in Eq. (5.8-1) is given in the frequency domain by the expression

$$\begin{aligned}\hat{F}(u, v) &= \left[\frac{H^*(u, v)S_f(u, v)}{S_f(u, v)|H(u, v)|^2 + S_\eta(u, v)}\right]G(u, v) \\ &= \left[\frac{H^*(u, v)}{|H(u, v)|^2 + S_\eta(u, v)/S_f(u, v)}\right]G(u, v) \\ &= \left[\frac{1}{H(u, v)}\frac{|H(u, v)|^2}{|H(u, v)|^2 + S_\eta(u, v)/S_f(u, v)}\right]G(u, v)\end{aligned} \tag{5.8-2}$$

where we used the fact that the product of a complex quantity with its conjugate is equal to the magnitude of the complex quantity squared. This result is known as the *Wiener filter*, after N. Wiener [1942], who first proposed the concept in the year shown. The filter, which consists of the terms inside the brackets, also is commonly referred to as the *minimum mean square error filter* or the *least square error filter*. We include references at the end of the chapter to sources containing detailed derivations of the Wiener filter. Note from the first line in Eq. (5.8-2) that the Wiener filter does not have the same problem as the inverse filter with zeros in the degradation function, unless the entire denominator is zero for the same value(s) of u and v.

The terms in Eq. (5.8-2) are as follows:

$H(u, v)$ = degradation function
$H^*(u, v)$ = complex conjugate of $H(u, v)$
$|H(u, v)|^2 = H^*(u, v)H(u, v)$
$S_\eta(u, v) = |N(u, v)|^2$ = power spectrum of the noise [see Eq. (4.6–18)]†
$S_f(u, v) = |F(u, v)|^2$ = power spectrum of the undegraded image

†The term $|N(u, v)|^2$ also is referred to as the *autocorrelation* of the noise. This terminology comes from the correlation theorem (first line of entry 7 in Table 4.3). When the two functions are the same, correlation becomes *autocorrelation* and the right side of that entry becomes $N^*(u, v)N(u, v)$, which is equal to $|N(u, v)|^2$. Similar comments apply to $|F(u, v)|^2$, which is the autocorrelation of the image. We discuss correlation in more detail in Chapter 12.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

now is much easier to read. In the third row of Fig. 5.29, the noise variance has decreased more than five orders of magnitude from the first row. In fact, image 5.29(g) has no visible noise. The inverse filter result is interesting in this case. The noise is still quite visible, but the text can be seen through a "curtain" of noise. This is a good example of the comments made regarding Eq. (5.7-2). In other words, as is evident in Fig. 5.29(h), the inverse filter was quite capable of essentially eliminating the blur in the image. However, the noise still dominates the result. If we could "look" behind the noise in Figs. 5.29(b) and (e), the characters also would show little blurring. The Wiener filter result in Fig. 5.29(i) is excellent, being quite close visually to the original image in Fig. 5.26(a). These types of results are representative of what is possible with Wiener filtering, as long as a reasonable estimate of the degradation function is available. ■

5.9 Constrained Least Squares Filtering

The problem of having to know something about the degradation function H is common to all methods discussed in this chapter. However, the Wiener filter presents an additional difficulty: The power spectra of the undegraded image and noise must be known. We showed in the previous section that it is possible to achieve excellent results using the approximation given in Eq. (5.8-6). However, a constant estimate of the ratio of the power spectra is not always a suitable solution.

The method discussed in this section requires knowledge of only the mean and variance of the noise. As discussed in Section 5.2.4, these parameters usually can be calculated from a given degraded image, so this is an important advantage. Another difference is that the Wiener filter is based on minimizing a statistical criterion and, as such, it is optimal in an average sense. The algorithm presented in this section has the notable feature that it yields an optimal result for *each* image to which it is applied. Of course, it is important to keep in mind that these optimality criteria, while satisfying from a theoretical point of view, are not related to the dynamics of visual perception. As a result, the choice of one algorithm over the other will almost always be determined (at least partially) by the perceived visual quality of the resulting images.

By using the definition of convolution given in Eq. (4.6-23), and as explained in Section 2.6.6, we can express Eq. (5.5-16) in vector-matrix form:

$$\mathbf{g} = \mathbf{Hf} + \boldsymbol{\eta} \tag{5.9-1}$$

Consult the book Web site for a brief review of vectors and matrices.

For example, suppose that $g(x, y)$ is of size $M \times N$. Then we can form the first N elements of the vector $\mathbf{g}$ by using the image elements in first row of $g(x, y)$, the next N elements from the second row, and so on. The resulting vector will have dimensions $MN \times 1$. These are also the dimensions of $\mathbf{f}$ and $\boldsymbol{\eta}$, as these vectors are formed in the same manner. The matrix $\mathbf{H}$ then has dimensions $MN \times MN$. Its elements are given by the elements of the convolution given in Eq. (4.6-23).

It would be reasonable to arrive at the conclusion that the restoration problem can now be reduced to simple matrix manipulations. Unfortunately, this is not the case. For instance, suppose that we are working with images of medium size; say $M = N = 512$. Then the vectors in Eq. (5.9-1) would be of dimension

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

where the second line follows from Eq. (4.2-10). As noted earlier, this is a line integral (along the line $L(\rho, 0)$ in this case). Also, note that $g(\rho, \theta) = 0$ when $|\rho| > r$. When $|\rho| \leq r$ the integral is evaluated from $y = -\sqrt{r^2 - \rho^2}$ to $y = \sqrt{r^2 - \rho^2}$. Therefore,

$$g(\rho, \theta) = \int_{-\sqrt{r^2-\rho^2}}^{\sqrt{r^2-\rho^2}} f(\rho, y)dy$$

$$= \int_{-\sqrt{r^2-\rho^2}}^{\sqrt{r^2-\rho^2}} A dy$$

Carrying out the integration yields

$$g(\rho, \theta) = g(\rho) = \begin{cases} 2A\sqrt{r^2 - \rho^2} & |\rho| \leq r \\ 0 & \text{otherwise} \end{cases}$$

where we used the fact mentioned above that $g(\rho, \theta) = 0$ when $|\rho| > r$. Figure 5.38(b) shows the result, which agrees with the projections illustrated in Figs. 5.32 and 5.33. Note that $g(\rho, \theta) = g(\rho)$; that is, g is independent of θ because the object is symmetric about the origin. ■

When the Radon transform, $g(\rho, \theta)$, is displayed as an image with ρ and θ as rectilinear coordinates, the result is called a *sinogram*, similar in concept to displaying the Fourier spectrum (unlike the Fourier transform, however, $g(\rho, \theta)$ is always a real function). Like the Fourier transform, a sinogram contains the data necessary to reconstruct $f(x,y)$. As is the case with displays of the Fourier spectrum, sinograms can be readily interpreted for simple regions, but become increasingly difficult to "read" as the region being projected becomes more complex. For example, Fig. 5.39(b) is the sinogram of the rectangle on the left. The vertical and horizontal axes correspond to θ and ρ, respectively. Thus, the bottom row is the projection of the rectangle in the horizontal direction (i.e., $\theta = 0°$), and the middle row is the projection in the vertical direction ($\theta = 90°$). The fact that the nonzero portion of the bottom row is smaller than the nonzero portion of the middle row tells us that the object is narrower in the horizontal direction. The fact that the sinogram is symmetric in both directions about the center of the image tells us that we are dealing with an object that is symmetric and parallel to the x and y axes. Finally, the sinogram is smooth, indicating that the object has a uniform intensity. Other than these types of general observations, we cannot say much more about this sinogram.

Figure 5.39(c) shows an image of the *Shepp-Logan phantom*, a widely used synthetic image designed to simulate the absorption of major areas of the brain, including small tumors. The sinogram of this image is considerably more difficult to interpret, as Fig. 5.39(d) shows. We still can infer some symmetry properties, but that is about all we can say. Visual analysis of sinograms is of limited practical use, but sometimes it is helpful in algorithm development.

To generate arrays with rows of the same size, the minimum dimension of the ρ-axis in sinograms corresponds to the largest dimension encountered during projection. For example, the minimum size of a sinogram of a square of size $M \times M$ obtained using increments of 1° is $180 \times Q$, where Q is the smallest integer greater than $\sqrt{2}M$.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

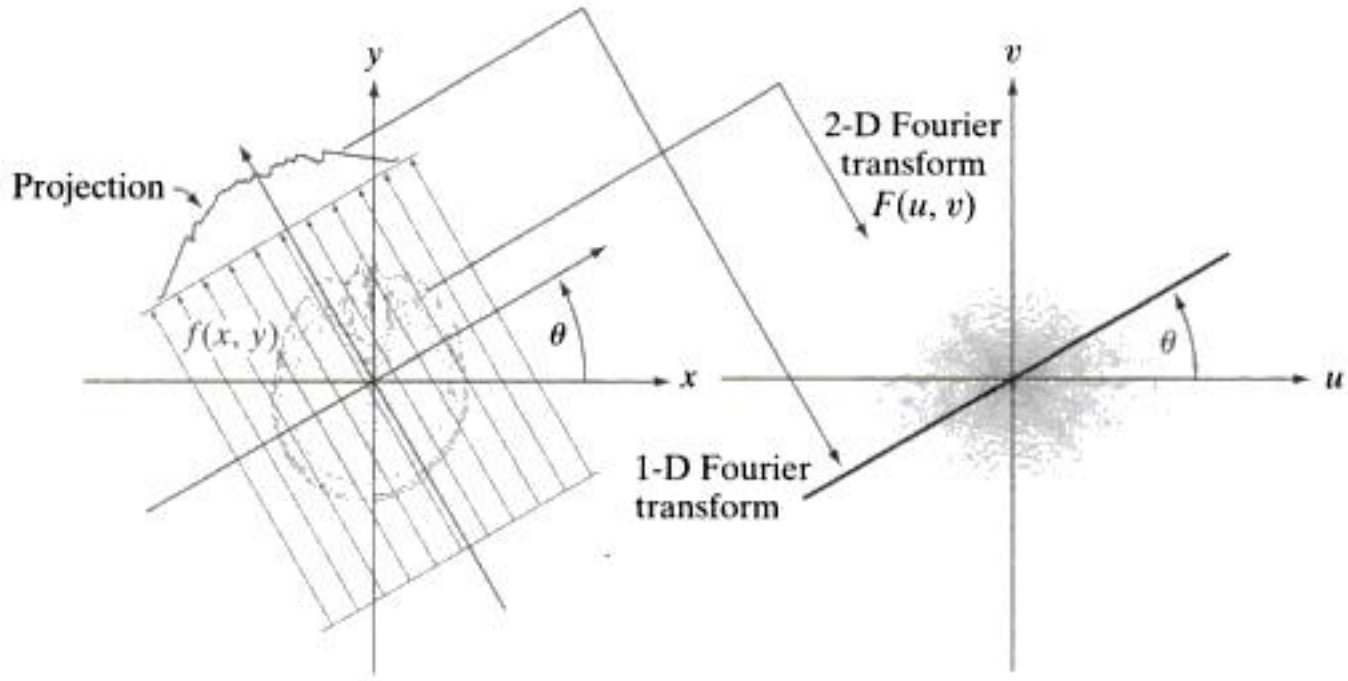


FIGURE 5.41 Illustration of the Fourier-slice theorem. The 1-D Fourier transform of a projection is a slice of the 2-D Fourier transform of the region from which the projection was obtained. Note the correspondence of the angle θ.

Fourier transform of $F(u, v)$.[†] However, this is expensive computationally, as it involves inverting a 2-D transform. The approach discussed in the following section is much more efficient.

5.11.5 Reconstruction Using Parallel-Beam Filtered Backprojections

As we saw in Section 5.11.1 and in Example 5.18, obtaining backprojections directly yields unacceptably blurred results. Fortunately, there is a straightforward solution to this problem based simply on filtering the projections before computing the backprojections. From Eq. (4.5-8), the 2-D inverse Fourier transform of $F(u, v)$ is

$$f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(u, v) e^{j2\pi(ux+vy)}\, du\, dv \qquad (5.11\text{-}12)$$

If, as in Eqs. (5.11-10) and (5.11-11), we let $u = \omega \cos\theta$ and $v = \omega \sin\theta$, then the differentials become $du\, dv = \omega\, d\omega\, d\theta$, and we can express Eq. (5.11-12) in polar coordinates:

The relationship $du\, dv = \omega\, d\omega\, d\theta$ is from basic integral calculus, where the Jacobian is used as the basis for a change of variables.

$$f(x, y) = \int_{0}^{2\pi} \int_{0}^{\infty} F(\omega \cos\theta, \omega \sin\theta) e^{j2\pi\omega(x \cos\theta + y \sin\theta)} \omega\, d\omega\, d\theta \qquad (5.11\text{-}13)$$

Then, using the Fourier-slice theorem,

$$f(x, y) = \int_{0}^{2\pi} \int_{0}^{\infty} G(\omega, \theta) e^{j2\pi\omega(x \cos\theta + y \sin\theta)} \omega\, d\omega\, d\theta \qquad (5.11\text{-}14)$$

[†]Keep in mind that blurring will still be present in an image recovered using the inverse Fourier transform, because the result is equivalent to the result obtained using the approach discussed in the previous section.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

Umbaugh [2005], and Petrou and Bosdogianni [1999]. This last reference also presents a nice tie between two-dimensional frequency domain filters and the corresponding digital filters. On the design of 2-D digital filters, see Lu and Antoniou [1992].

Basic references for computed tomography are Rosenfeld and Kak [1982], Kak and Slaney [2001], and Prince and Links [2006]. For further reading on the Shepp-Logan phantom see Shepp and Logan [1974], and for additional details on the origin of the Ram-Lak filter see Ramachandran and Lakshminarayanan [1971]. The paper by O'Connor and Fessler [2006] is representative of current research in the signal and image processing aspects of computed tomography.

For software techniques to implement most of the material discussed in this chapter see Gonzalez, Woods, and Eddins [2004].

Problems

Detailed solutions to the problems marked with a star can be found in the book Web site. The site also contains suggested projects based on the material in this chapter.

★5.1 The white bars in the test pattern shown are 7 pixels wide and 210 pixels high. The separation between bars is 17 pixels. What would this image look like after application of

(a) A 3×3 arithmetic mean filter?

(b) A 5×5 arithmetic mean filter?

(c) A 9×9 arithmetic mean filter?

Note: This problem and the ones that follow it, related to filtering this image, may seem a bit tedious. However, they are worth the effort, as they help develop a real understanding of how these filters work. After you understand how a particular filter affects the image, your answer can be a brief verbal description of the result. For example, "the resulting image will consist of vertical bars 3 pixels wide and 206 pixels high." Be sure to describe any deformation of the bars, such as rounded corners. You may ignore image border effects, in which the masks only partially contain image pixels.

5.2 Repeat Problem 5.1 using a geometric mean filter.

★5.3 Repeat Problem 5.1 using a harmonic mean filter.

5.4 Repeat Problem 5.1 using a contraharmonic mean filter with $Q = 1.5$.

★5.5 Repeat Problem 5.1 using a contraharmonic mean filter with $Q = -1.5$.

5.6 Repeat Problem 5.1 using a median filter.

★5.7 Repeat Problem 5.1 using a max filter.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

every item for which it is responsible. Unfortunately, the photos of the coins in question are blurred to the point where the date and other small markings are not readable. The cause of the blurring was the camera being out of focus when the pictures were taken. As an image processing expert and friend of the professor, you are asked as a favor to determine whether computer processing can be utilized to restore the images to the point where the professor can read the markings. You are told that the original camera used to take the photos is still available, as are other representative coins of the same era. Propose a step-by-step solution to this problem.

5.28 Sketch the Radon transform of the following square images. Label quantitatively all the important features of your sketches. Figure (a) consists of one dot in the center, and (b) has two dots along the diagonal. Describe your solution to (c) by an intensity profile. Assume a parallel-beam geometry.

(a) **★(b)** **(c)**

5.29 Show that the Radon transform [Eq. (5.11-3)] of the Gaussian shape $f(x, y) = A \exp\left(\frac{-x^2 - y^2}{2\sigma^2}\right)$ is $g(\rho, \theta) = A\sqrt{2\pi}.\sigma.\exp(-\rho^2)$. (*Hint:* Refer to Example 5.17, where we used symmetry to simplify integration.)

5.30 **★(a)** Show that the Radon transform [Eq. (5.11-3)] of the unit impulse $\delta(x, y)$ is a straight vertical line in the $\rho\theta$-plane passing through the origin.

(b) Show that the radon transform of the impulse $\delta(x - x_0, y - y_0)$ is a sinusoidal curve in the $\rho\theta$-plane.

5.31 Prove the validity of the following properties of the Radon transform [Eq. (5.11-3)]:

★(a) Linearity: The Radon transform is a linear operator. (See Section 2.6.2 regarding the definition of linear operators.)

(b) Translation property: The radon transform of $f(x - x_0, y - y_0)$ is $g(\rho - x_0 \cos_\theta - y_0 \sin_\theta, \theta)$.

★(c) Convolution property: Show that the Radon transform of the convolution of two functions is equal to the convolution of the Radon transforms of the two functions.

5.32 Provide the steps leading from Eq. (5.11-14) to (5.11-15). You will need to use the property $G(\omega, \theta + 180°) = G(-\omega, \theta)$.

★5.33 Prove the validity of Eq. (5.11-25).

5.34 Prove the validity of Eq. (5.11-27).

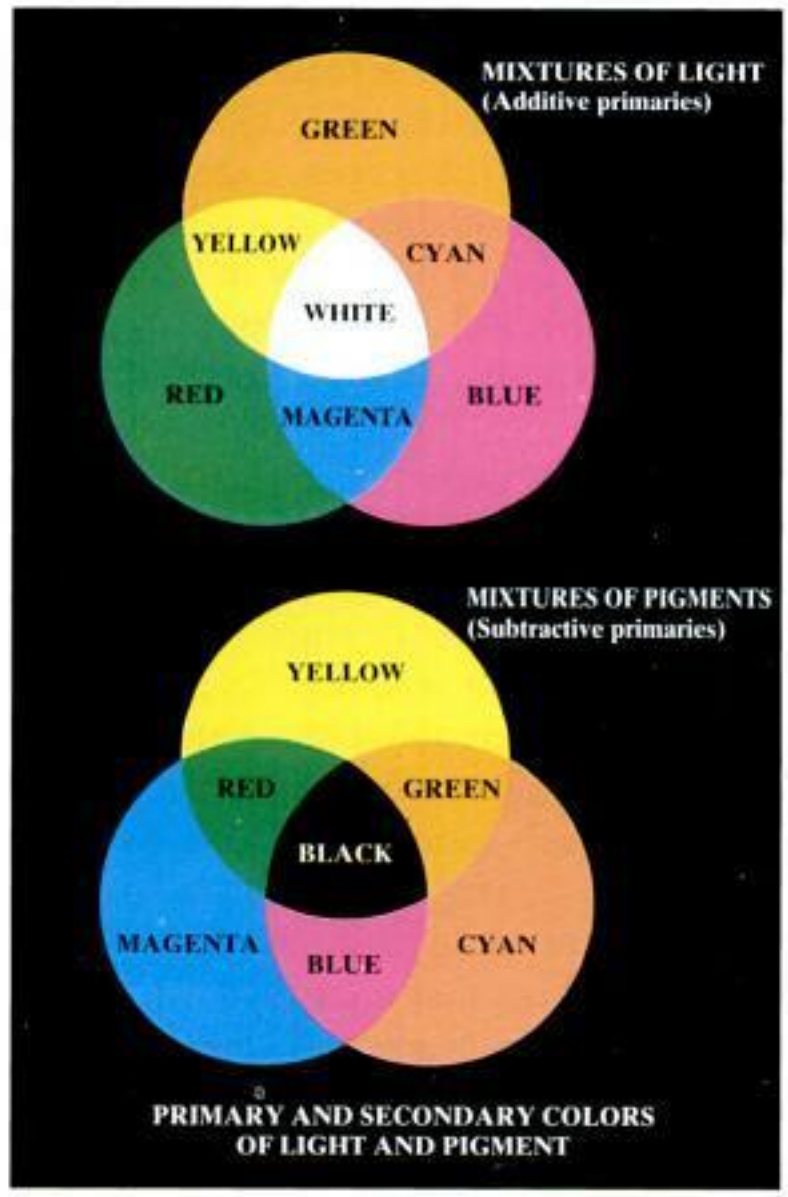
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

human eye, colors are seen as variable combinations of the so-called *primary colors* red (R), green (G), and blue (B). For the purpose of standardization, the CIE (Commission Internationale de l'Eclairage—the International Commission on Illumination) designated in 1931 the following specific wavelength values to the three primary colors: blue = 435.8 nm, green = 546.1 nm, and red = 700 nm. This standard was set before the detailed experimental curves shown in Fig. 6.3 became available in 1965. Thus, the CIE standards correspond only approximately with experimental data. We note from Figs. 6.2 and 6.3 that no single color may be called red, green, or blue. Also, it is important to keep in mind that having three specific primary color wavelengths for the purpose of standardization does not mean that these three fixed RGB components acting alone can generate all spectrum colors. Use of the word *primary* has been widely misinterpreted to mean that the three standard primaries, when mixed in various intensity proportions, can produce *all* visible colors. As you will see shortly, this interpretation is not correct unless the wavelength also is allowed to vary, in which case we would no longer have three fixed, standard primary colors.

The primary colors can be added to produce the *secondary* colors of light—magenta (red plus blue), cyan (green plus blue), and yellow (red plus green). Mixing the three primaries, or a secondary with its opposite primary color, in the right intensities produces white light. This result is shown in Fig. 6.4(a), which also illustrates the three primary colors and their combinations to produce the secondary colors.



a
b

FIGURE 6.4 Primary and secondary colors of light and pigments. (Courtesy of the General Electric Co., Lamp Business Division.)

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

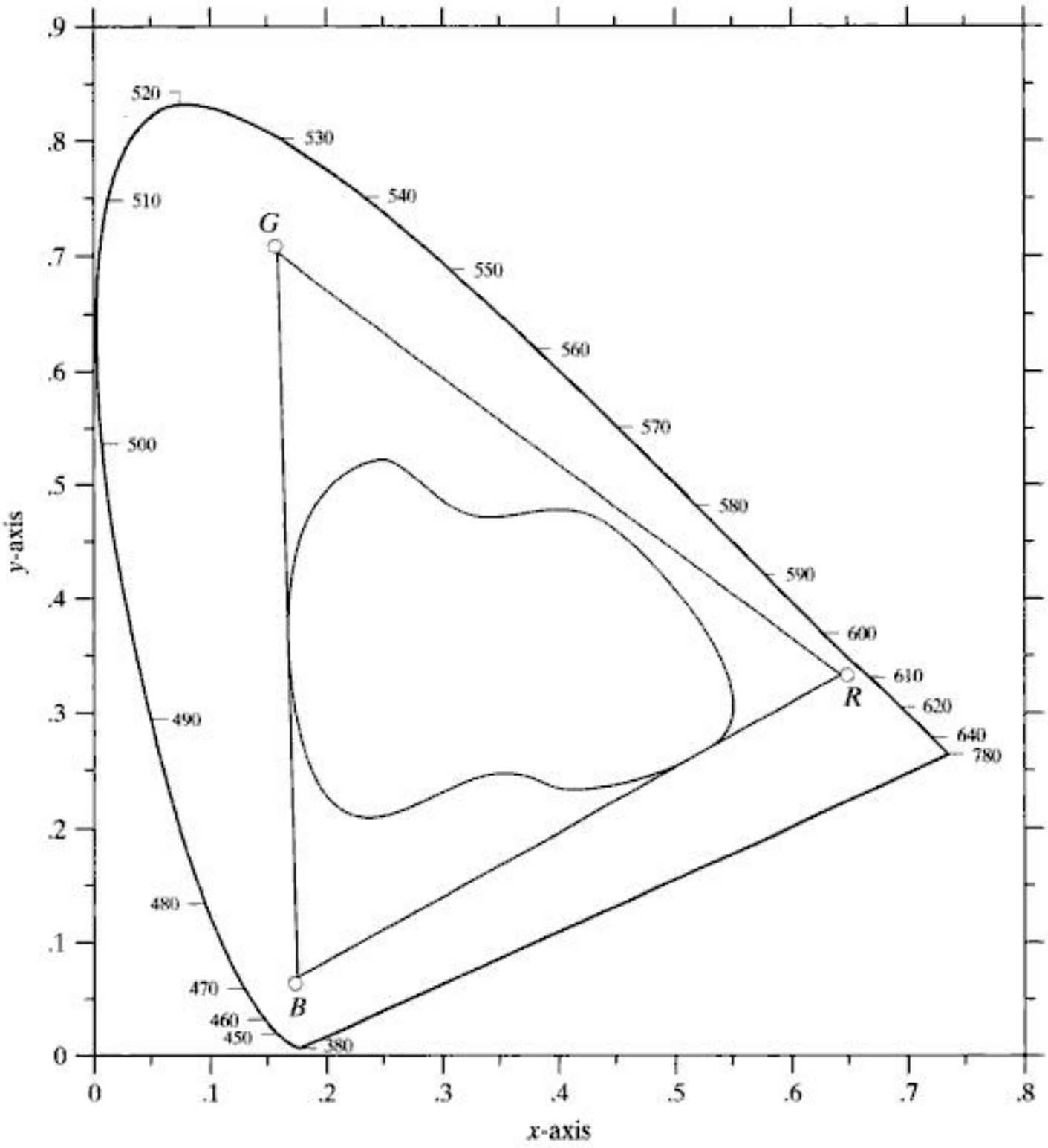


FIGURE 6.6
Typical color gamut of color monitors (triangle) and color printing devices (irregular region).

devices. The boundary of the color printing gamut is irregular because color printing is a combination of additive and subtractive color mixing, a process that is much more difficult to control than that of displaying colors on a monitor, which is based on the addition of three highly controllable light primaries.

6.2 Color Models

The purpose of a color model (also called *color space* or *color system*) is to facilitate the specification of colors in some standard, generally accepted way. In essence, a color model is a specification of a coordinate system and a subspace within that system where each color is represented by a single point.

Most color models in use today are oriented either toward hardware (such as for color monitors and printers) or toward applications where color manipulation is a goal (such as in the creation of color graphics for animation). In

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

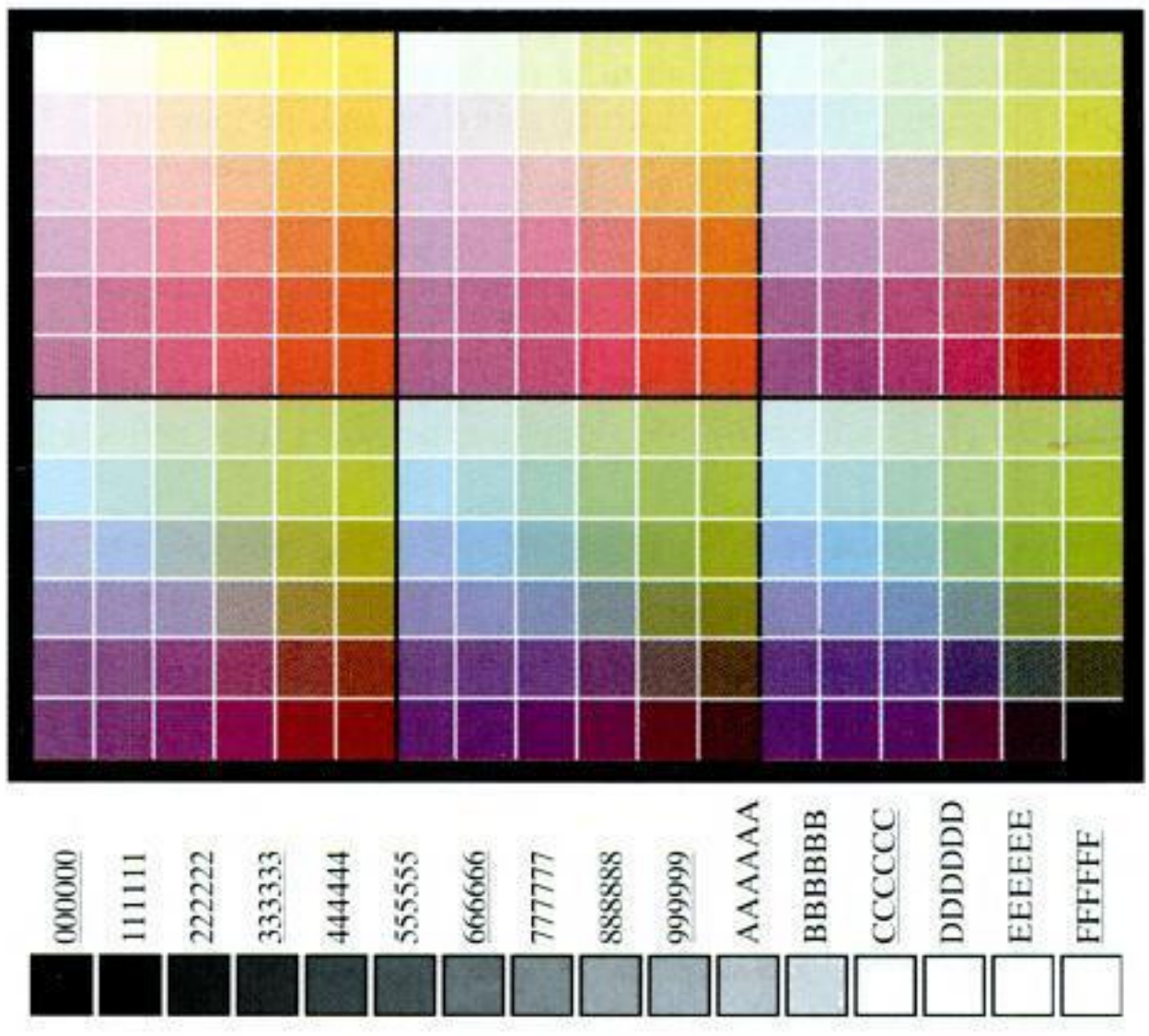
Number System	Color Equivalents					
Hex	00	33	66	99	CC	FF
Decimal	0	51	102	153	204	255

TABLE 6.1
Valid values of each RGB component in a safe color.

Each of the 216 safe colors is formed from three RGB values as before, but each value can only be 0, 51, 102, 153, 204, or 255. Thus, RGB triplets of these values give us $(6)^3 = 216$ possible values (note that all values are divisible by 3). It is customary to express these values in the hexagonal number system, as shown in Table 6.1. Recall that hex numbers 0, 1, 2, . . . , 9, A, B, C, D, E, F correspond to decimal numbers 0, 1, 2, . . . , 9, 10, 11, 12, 13, 14, 15. Recall also that $(0)_{16} = (0000)_2$ and $(F)_{16} = (1111)_2$. Thus, for example, $(FF)_{16} = (255)_{10} = (11111111)_2$ and we see that a grouping of two hex numbers forms an 8-bit byte.

Since it takes three numbers to form an RGB color, each safe color is formed from three of the two digit hex numbers in Table 6.1. For example, the purest red is FF0000. The values 000000 and FFFFFF represent black and white, respectively. Keep in mind that the same result is obtained by using the more familiar decimal notation. For instance, the brightest red in decimal notation has $R = 255$ (FF) and $G = B = 0$.

Figure 6.10(a) shows the 216 safe colors, organized in descending RGB values. The square in the top left array has value FFFFFF (white), the second square to its right has value FFFFCC, the third square has value FFFF99, and



a
b

FIGURE 6.10
(a) The 216 safe RGB colors. (b) All the grays in the 256-color RGB system (grays that are part of the safe color group are shown underlined).

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

by experimentation that the regions shown in white in Fig. 6.42(h) are about the best this method can do in identifying the reddish components of the original image. The segmentation method discussed in the following section is capable of yielding considerably better results. ■

6.7.2 Segmentation in RGB Vector Space

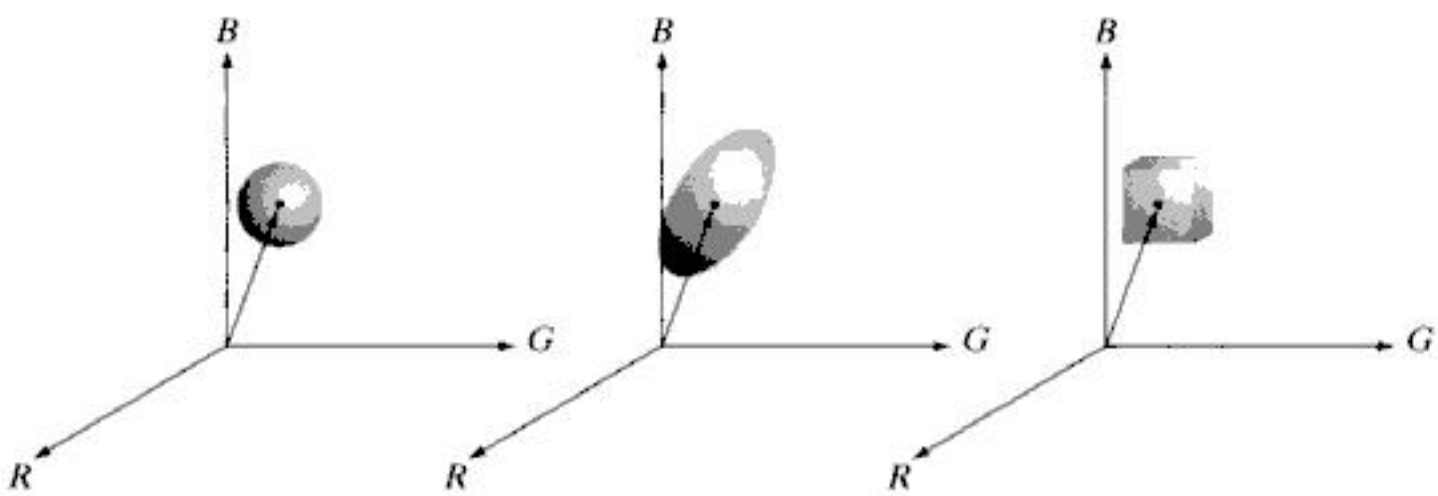
Although, as mentioned numerous times in this chapter, working in HSI space is more intuitive, segmentation is one area in which better results generally are obtained by using RGB color vectors. The approach is straightforward. Suppose that the objective is to segment objects of a specified color range in an RGB image. Given a set of sample color points representative of the colors of interest, we obtain an estimate of the "average" color that we wish to segment. Let this average color be denoted by the RGB vector $\mathbf{a}$. The objective of segmentation is to classify each RGB pixel in a given image as having a color in the specified range or not. In order to perform this comparison, it is necessary to have a measure of similarity. One of the simplest measures is the Euclidean distance. Let $\mathbf{z}$ denote an arbitrary point in RGB space. We say that $\mathbf{z}$ is *similar* to $\mathbf{a}$ if the distance between them is less than a specified threshold, D_0. The Euclidean distance between $\mathbf{z}$ and $\mathbf{a}$ is given by

$$\begin{aligned} D(\mathbf{z}, \mathbf{a}) &= \|\mathbf{z} - \mathbf{a}\| \\ &= [(\mathbf{z} - \mathbf{a})^T(\mathbf{z} - \mathbf{a})]^{\frac{1}{2}} \\ &= [(z_R - a_R)^2 + (z_G - a_G)^2 + (z_B - a_B)^2]^{\frac{1}{2}} \end{aligned} \tag{6.7-1}$$

where the subscripts R, G, and B denote the RGB components of vectors $\mathbf{a}$ and $\mathbf{z}$. The locus of points such that $D(\mathbf{z}, \mathbf{a}) \leq D_0$ is a solid sphere of radius D_0, as illustrated in Fig. 6.43(a). Points contained within the sphere satisfy the specified color criterion; points outside the sphere do not. Coding these two sets of points in the image with, say, black and white, produces a binary segmented image.

A useful generalization of Eq. (6.7-1) is a distance measure of the form

$$D(\mathbf{z}, \mathbf{a}) = [(\mathbf{z} - \mathbf{a})^T\mathbf{C}^{-1}(\mathbf{z} - \mathbf{a})]^{\frac{1}{2}} \tag{6.7-2}$$



a b c

FIGURE 6.43 Three approaches for enclosing data regions for RGB vector segmentation.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

morphological filters, image restoration, image compression, and many others. These processes are not equivalent to color processing carried out on the individual component images of a color image. The references in the following section provide a pointer to further results in this field.

Our treatment of noise in color images also points out that the vector nature of the problem, along with the fact that color images are routinely transformed from one working space to another, has implications on the issue of how to reduce noise in these images. In some cases, noise filtering can be done on a per-image basis, but others, such as median filtering, require special treatment to reflect the fact that color pixels are vector quantities, as mentioned in the previous paragraph.

Although segmentation is the topic of Chapter 10 and image data compression is the topic of Chapter 8, we gained the advantage of continuity by introducing them here in the context of color image processing. As will become evident in subsequent discussions, many of the techniques developed in those chapters are applicable to the discussion in this chapter.

References and Further Reading

For a comprehensive reference on the science of color, see Malacara [2001]. Regarding the physiology of color, see Gegenfurtner and Sharpe [1999]. These two references, along with the early books by Walsh [1958] and by Kiver [1965], provide ample supplementary material for the discussion in Section 6.1. For further reading on color models (Section 6.2), see Fortner and Meyer [1997], Poynton [1996], and Fairchild [1998]. For a detailed derivation of the HSI model equations in Section 6.2.3 see the paper by Smith [1978] or consult the book Web site. The topic of pseudocolor (Section 6.3) is closely tied to the general area of data visualization. Wolff and Yaeger [1993] is a good basic reference on the use of pseudocolor. The book by Thorell and Smith [1990] also is of interest. For a discussion on the vector representation of color signals (Section 6.4), see Plataniotis and Venetsanopoulos [2000].

References for Section 6.5 are Benson [1985], Robertson [1977], and CIE [1978]. See also the classic paper by MacAdam [1942]. The material on color image filtering (Section 6.6) is based on the vector formulation introduced in Section 6.4 and on our discussion of spatial filtering in Chapter 3. Segmentation of color images (Section 6.7) has been a topic of much attention during the past ten years. The papers by Liu and Yang [1994] and by Shafarenko et al. [1998] are representative of work in this field. A special issue of the *IEEE Transactions on Image Processing* [1997] also is of interest. The discussion on color edge detection (Section 6.7.3) is from Di Zenzo [1986]. The book by Plataniotis and Venetsanopoulos [2000] does a good job of summarizing a variety of approaches to the segmentation of color images. The discussion in Section 6.8 is based on the noise models introduced in Section 5.2. References on image compression (Section 6.9) are listed at the end of Chapter 8. For details of software implementation of many of the techniques discussed in this chapter, see Gonzalez, Woods, and Eddins [2004].

Problems

Detailed solutions to the problems marked with a star can be found in the book Web site. The site also contains suggested projects based on the material in this chapter.

6.1 Give the percentages of red *(X)*, green *(Y)*, and blue *(Z)* light required to generate the point labeled "Day Light" in Fig. 6.5.

★6.2 Consider any two valid colors c_1 and c_2 with coordinates (x_1, y_1) and (x_2, y_2) in the chromaticity diagram of Fig. 6.5. Derive the necessary general expression(s) for computing the relative percentages of colors c_1 and c_2 composing a given color that is known to lie on the straight line joining these two colors.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

$$\langle g_i(n), g_j(n + 2m)\rangle = \delta(i - j)\delta(m), \quad i, j = \{0, 1\} \tag{7.1-13}$$

which defines *orthonormality* for perfect reconstruction filter banks. In addition to Eq. (7.1-13), orthonormal filters can be shown to satisfy the following two conditions:

$$\begin{aligned} g_1(n) &= (-1)^n g_0(K_{\text{even}} - 1 - n) \\ h_i(n) &= g_i(K_{\text{even}} - 1 - n), \quad i = \{0, 1\} \end{aligned} \tag{7.1-14}$$

where the subscript on K_{even} is used to indicate that the number of filter coefficients must be divisible by 2 (i.e., an even number). As Eq. (7.1-14) indicates, synthesis filter g_1 is related to g_0 by order reversal and modulation. In addition, both h_0 and h_1 are order-reversed versions of synthesis filters, g_0 and g_1, respectively. Thus, an orthonormal filter bank can be developed around the impulse response of a single filter, called the *prototype*; the remaining filters can be computed from the specified prototype's impulse response. For biorthogonal filter banks, two prototypes are required; the remaining filters can be computed via Eq. (7.1-10) or (7.1-11). The generation of useful prototype filters, whether orthonormal or biorthogonal, is beyond the scope of this chapter. We simply use filters that have been presented in the literature and provide references for further study.

Before concluding the section with a 2-D subband coding example, we note that 1-D orthonormal and biorthogonal filters can be used as 2-D separable filters for the processing of images. As can be seen in Fig. 7.7, the separable filters are first applied in one dimension (e.g., vertically) and then in the other (e.g., horizontally) in the manner introduced in Section 2.6.7. Moreover, downsampling is performed in two stages—once before the second filtering operation to reduce the overall number of computations. The resulting filtered

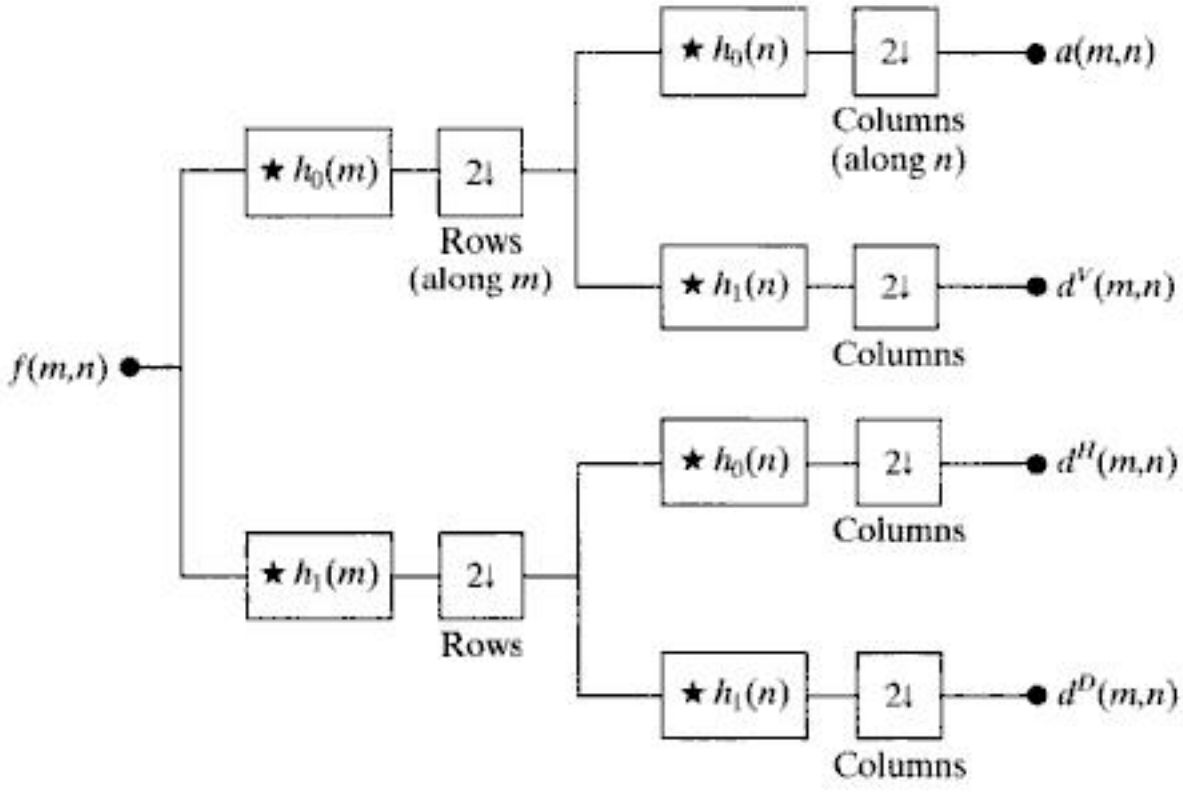


FIGURE 7.7
A two-dimensional, four-band filter bank for subband image coding.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

7.18 **(a)** The continuous wavelet transform of Problem 7.17 is computer generated. The function upon which it is based was first sampled at discrete intervals. What is continuous about the transform—or what distinguishes it from the discrete wavelet transform of the function?

★**(b)** Under what circumstances is the DWT a better choice than the CWT? Are there times when the CWT is better than the DWT?

★**7.19** Draw the FWT filter bank required to compute the transform in Problem 7.16. Label all inputs and outputs with the appropriate sequences.

7.20 The computational complexity of an M-point fast wavelet transform is $O(M)$. That is, the number of operations is proportional M. What determines the constant of proportionality?

7.21 ★**(a)** If the input to the three-scale FWT filter bank of Fig. 7.30(a) is the Haar scaling function $\varphi(n) = 1$ for $n = 0, 1, \ldots, 7$ and 0 elsewhere, what is the resulting transform with respect to Haar wavelets?

(b) What is the transform if the input is the corresponding Haar wavelet function $\psi(n) = \{1, 1, 1, 1, -1, -1, -1, -1\}$ for $n = 0, 1, \ldots, 7$?

(c) What input sequence produces transform $\{0, 0, 0, B, 0, 0, 0, 0\}$ with nonzero coefficient $W_\psi(1, 1) = B$?

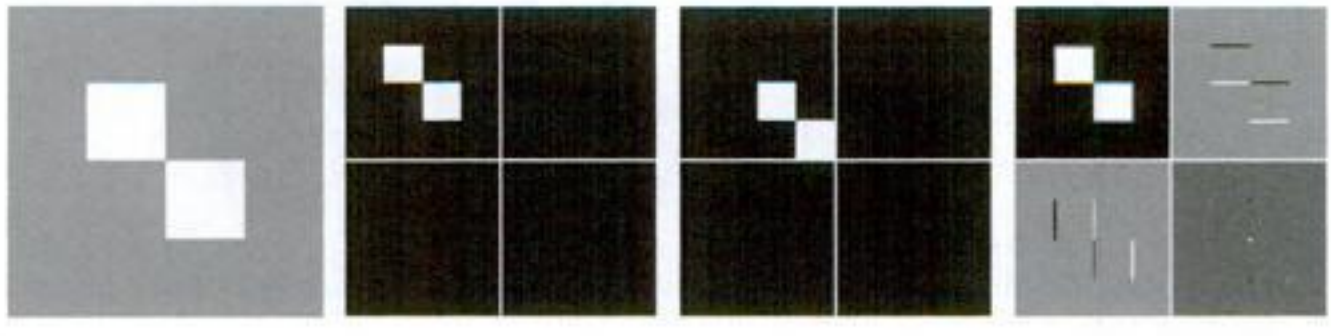
★**7.22** The two-dimensional fast wavelet transform is similar to the pyramidal coding scheme of Section 7.2.1. How are they similar? Given the three-scale wavelet transform in Fig. 7.10(a), how would you construct the corresponding approximation pyramid? How many levels would it have?

7.23 Compute the two-dimensional wavelet transform with respect to Haar wavelets of the 2×2 image in Problem 7.9. Draw the required filter bank and label all inputs and outputs with the proper arrays.

★**7.24** In the Fourier domain

$$f(x - x_0, y - y_0) \Leftrightarrow F(\mu, v)e^{-2\pi(\mu x_0/M + v y_0/N)}$$

and translation does not affect the display of $|F(\mu, v)|$. Using the following sequence of images, explain the translation property of wavelet transforms. The leftmost image contains two 16×16 white squares centered on a 64×64 gray background. The second image (from the left) is its single-scale wavelet transform with respect to Haar wavelets. The third is the wavelet transform of the original image after shifting it 16 pixels to the right and downward, and the final (rightmost) image is the wavelet transform of the original image after it has been shifted one pixel to the right and downward.



7.25 The following table shows the Haar wavelet and scaling functions for a four-scale fast wavelet transform. Sketch the additional basis functions needed for a full three-scale packet decomposition. Give the mathematical expression or expressions for determining them. Then order the basis functions according to frequency content and explain the results.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

The corresponding relative data redundancy of the larger representation is 0.9 ($R = 0.9$), indicating that 90% of its data is redundant.

In the context of digital image compression, b in Eq. (8.1-2) usually is the number of bits needed to represent an image as a 2-D array of intensity values. The 2-D intensity arrays introduced in Section 2.4.2 are the preferred formats for human viewing and interpretation—and the standard by which all other representations are judged. When it comes to compact image representation, however, these formats are far from optimal. Two-dimensional intensity arrays suffer from three principal types of data redundancies that can be identified and exploited:

1. *Coding redundancy.* A *code* is a system of symbols (letters, numbers, bits, and the like) used to represent a body of information or set of events. Each piece of information or event is assigned a sequence of *code symbols*, called a *code word*. The number of symbols in each code word is its *length*. The 8-bit codes that are used to represent the intensities in most 2-D intensity arrays contain more bits than are needed to represent the intensities.
2. *Spatial* and *temporal redundancy*. Because the pixels of most 2-D intensity arrays are correlated spatially (i.e., each pixel is similar to or dependent on neighboring pixels), information is unnecessarily replicated in the representations of the correlated pixels. In a video sequence, temporally correlated pixels (i.e., those similar to or dependent on pixels in nearby frames) also duplicate information.
3. *Irrelevant information.* Most 2-D intensity arrays contain information that is ignored by the human visual system and/or extraneous to the intended use of the image. It is redundant in the sense that it is not used.

The computer-generated images in Figs. 8.1(a) through (c) exhibit each of these fundamental redundancies. As will be seen in the next three sections, compression is achieved when one or more redundancy is reduced or eliminated.

a b c

FIGURE 8.1 Computer generated 256 × 256 × 8 bit images with (a) coding redundancy, (b) spatial redundancy, and (c) irrelevant information. (Each was designed to demonstrate one principal redundancy but may exhibit others as well.)

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

Gaussian density.† The number of quantization levels (and thus the number of bits) allotted to each quantizer is made proportional to $\log_2 \sigma^2_{T(u,v)}$. Thus the retained coefficients in Eq. (8.2-24)—which (in the context of the current discussion) are selected on the basis of maximum variance—are assigned bits in proportion to the logarithm of the coefficient variances.

Threshold coding implementation Zonal coding usually is implemented by using a single fixed mask for all subimages. Threshold coding, however, is inherently adaptive in the sense that the location of the transform coefficients retained for each subimage vary from one subimage to another. In fact, threshold coding is the adaptive transform coding approach most often used in practice because of its computational simplicity. The underlying concept is that, for any subimage, the transform coefficients of largest magnitude make the most significant contribution to reconstructed subimage quality, as demonstrated in the last example. Because the locations of the maximum coefficients vary from one subimage to another, the elements of $X(u, v)T(u, v)$ normally are reordered (in a predefined manner) to form a 1-D, run-length coded sequence. Figure 8.29(c) shows a typical *threshold mask* for one subimage of a hypothetical image. This mask provides a convenient way to visualize the threshold coding process for the corresponding subimage, as well as to mathematically describe the process using Eq. (8.2-24). When the mask is applied [via Eq. (8.2-24)] to the subimage for which it was derived, and the resulting $n \times n$ array is reordered to form an n^2-element coefficient sequence in accordance with the zigzag ordering pattern of Fig. 8.29(d), the reordered 1-D sequence contains several long runs of 0s [the zigzag pattern becomes evident by starting at 0 in Fig. 8.29(d) and following the numbers in sequence]. These runs normally are run-length coded. The nonzero or retained coefficients, corresponding to the mask locations that contain a 1, are represented using a variable-length code.

There are three basic ways to threshold a transformed subimage or, stated differently, to create a subimage threshold masking function of the form given in Eq. (8.2-23): (1) A single global threshold can be applied to all subimages; (2) a different threshold can be used for each subimage; or (3) the threshold can be varied as a function of the location of each coefficient within the subimage. In the first approach, the level of compression differs from image to image, depending on the number of coefficients that exceed the global threshold. In the second, called *N-largest coding*, the same number of coefficients is discarded for each subimage. As a result, the code rate is constant and known in advance. The third technique, like the first, results in a variable code rate, but offers the advantage that thresholding *and* quantization can be combined

The N in "N-largest coding" is not an image dimension, but refers to the number of coefficients that are kept.

†As each coefficient is a linear combination of the pixels in its subimage [see Eq. (8.2-10)], the central-limit theorem suggests that, as subimage size increases, the coefficients tend to become Gaussian. This result does not apply to the dc coefficient, however, because nonnegative images always have positive dc coefficients.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

9.2.1 Erosion

With A and B as sets in Z^2, the erosion of A by B, denoted $A \ominus B$, is defined as

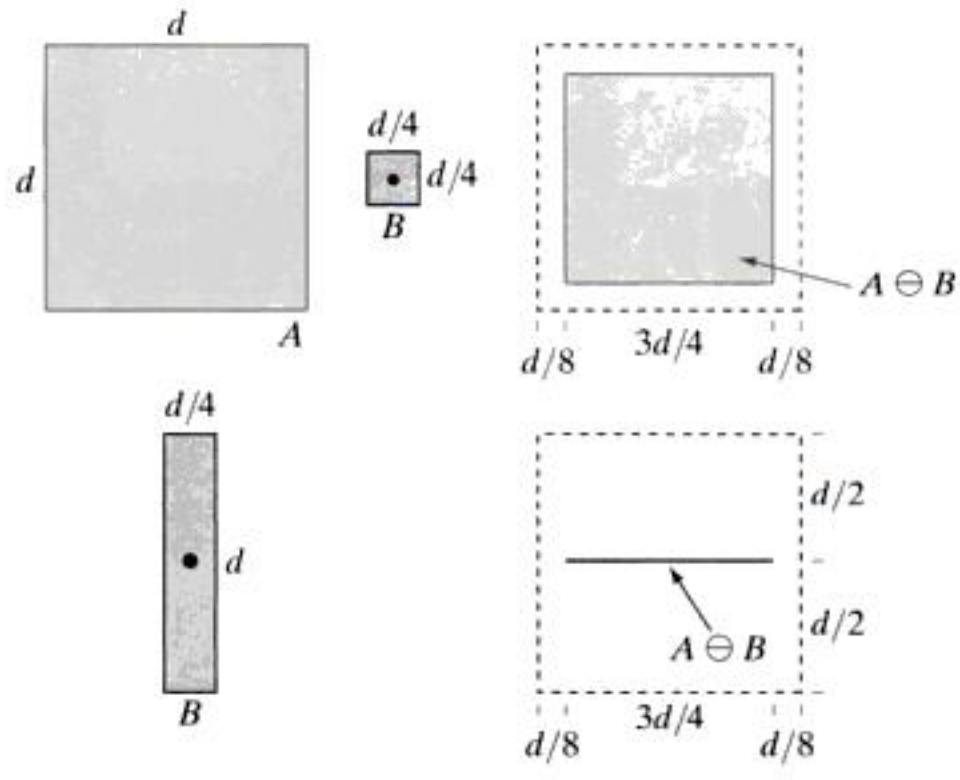
$$A \ominus B = \{z | (B)_z \subseteq A\} \tag{9.2-1}$$

In words, this equation indicates that the erosion of A by B is the set of all points z such that B, translated by z, is contained in A. In the following discussion, set B is assumed to be a structuring element. Equation (9.2-1) is the mathematical formulation of the example in Fig. 9.3(e), discussed at the end of the last section. Because the statement that B has to be contained in A is equivalent to B not sharing any common elements with the background, we can express erosion in the following equivalent form:

$$A \ominus B = \{z | (B)_z \cap A^c = \varnothing\} \tag{9.2-2}$$

where, as defined in Section 2.6.4, A^c is the complement of A and $\varnothing$ is the empty set.

Figure 9.4 shows an example of erosion. The elements of A and B are shown shaded and the background is white. The solid boundary in Fig. 9.4(c) is the limit beyond which further displacements of the origin of B would cause the structuring element to cease being completely contained in A. Thus, the locus of points (locations of the origin of B) within (and including) this boundary, constitutes the erosion of A by B. We show the erosion shaded in Fig. 9.4(c). Keep in mind that that erosion is simply the *set* of



a b c
d e

FIGURE 9.4 (a) Set A. (b) Square structuring element, B. (c) Erosion of A by B, shown shaded. (d) Elongated structuring element. (e) Erosion of A by B using this element. The dotted border in (c) and (e) is the boundary of set A, shown only for reference.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

Let the origin of each shape be located at its center of gravity. Let D be enclosed by a small window, W. The *local background* of D with respect to W is defined as the set difference $(W - D)$, as shown in Fig. 9.12(b). Figure 9.12(c) shows the complement of A, which is needed later. Figure 9.12(d) shows the erosion of A by D (the dashed lines are included for reference). Recall that the erosion of A by D is the set of locations of the *origin* of D, such that D is completely contained in A. Interpreted another way, $A \ominus D$ may be viewed geometrically as the set of all locations of the origin of D at which D found a match (hit) in A. Keep in mind that in Fig. 9.12 A consists only of the three *disjoint* sets C, D, and E.

Figure 9.12(e) shows the erosion of the complement of A by the local background set $(W - D)$. The outer shaded region in Fig. 9.12(e) is part of the erosion. We note from Figs. 9.12(d) and (e) that the set of locations for which D *exactly* fits inside A is the *intersection* of the erosion of A by D and the erosion of A^c by $(W - D)$ as shown in Fig. 9.12(f). This intersection is precisely the location sought. In other words, if B denotes the set composed of D and its background, the match (or set of matches) of B in A, denoted $A \circledast B$, is

$$A \circledast B = (A \ominus D) \cap \left[A^c \ominus (W - D)\right] \tag{9.4-1}$$

We can generalize the notation somewhat by letting $B = (B_1, B_2)$, where B_1 is the set formed from elements of B associated with an object and B_2 is the set of elements of B associated with the corresponding background. From the preceding discussion, $B_1 = D$ and $B_2 = (W - D)$. With this notation, Eq. (9.4-1) becomes

$$A \circledast B = (A \ominus B_1) \cap (A^c \ominus B_2) \tag{9.4-2}$$

Thus, set $A \circledast B$ contains all the (origin) points at which, simultaneously, B_1 found a match ("hit") in A and B_2 found a match in A^c. By using the definition of set differences given in Eq. (2.6-19) and the dual relationship between erosion and dilation given in Eq. (9.2-5), we can write Eq. (9.4-2) as

$$A \circledast B = (A \ominus B_1) - (A \oplus \hat{B}_2) \tag{9.4-3}$$

However, Eq. (9.4-2) is considerably more intuitive. We refer to any of the preceding three equations as the *morphological hit-or-miss transform*.

The reason for using a structuring element B_1 associated with objects and an element B_2 associated with the background is based on an assumed definition that two or more objects are distinct only if they form disjoint (disconnected) sets. This is guaranteed by requiring that each object have at least a one-pixel-thick background around it. In some applications, we may be interested in detecting certain patterns (combinations) of 1s and 0s within a set, in which case a background is not required. In such instances, the hit-or-miss transform reduces to simple erosion. As indicated previously, erosion is still a set of matches, but without the additional requirement of a background match for detecting individual objects. This simplified pattern detection scheme is used in some of the algorithms developed in the following section.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

convexity. One simple approach to reduce this effect is to limit growth so that it does not extend past the vertical and horizontal dimensions of the original set of points. Imposing this limitation on the example in Fig. 9.19 resulted in the image shown in Fig. 9.20. Boundaries of greater complexity can be used to limit growth even further in images with more detail. For example, we could use the maximum dimensions of the original set of points along the vertical, horizontal, and diagonal directions. The price paid for refinements such as this is additional complexity and increased computational requirements of the algorithm.

9.5.5 Thinning

The thinning of a set A by a structuring element B, denoted $A \otimes B$, can be defined in terms of the hit-or-miss transform:

$$\begin{aligned} A \otimes B &= A - (A \circledast B) \\ &= A \cap (A \circledast B)^c \end{aligned} \tag{9.5-6}$$

As in the previous section, we are interested only in pattern matching with the structuring elements, so no background operation is required in the hit-or-miss transform. A more useful expression for thinning A symmetrically is based on a *sequence* of structuring elements:

$$\{B\} = \{B^1, B^2, B^3, \ldots, B^n\} \tag{9.5-7}$$

where B^i is a rotated version of B^{i-1}. Using this concept, we now define thinning by a sequence of structuring elements as

$$A \otimes \{B\} = ((\ldots((A \otimes B^1) \otimes B^2)\ldots) \otimes B^n) \tag{9.5-8}$$

The process is to thin A by *one pass* with B^1, then thin the result with one pass of B^2, and so on, until A is thinned with one pass of B^n. The entire process is repeated until no further changes occur. Each individual thinning pass is performed using Eq. (9.5-6).

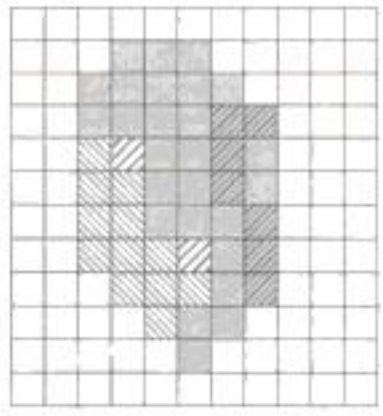


FIGURE 9.20 Result of limiting growth of the convex hull algorithm to the maximum dimensions of the original set of points along the vertical and horizontal directions.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

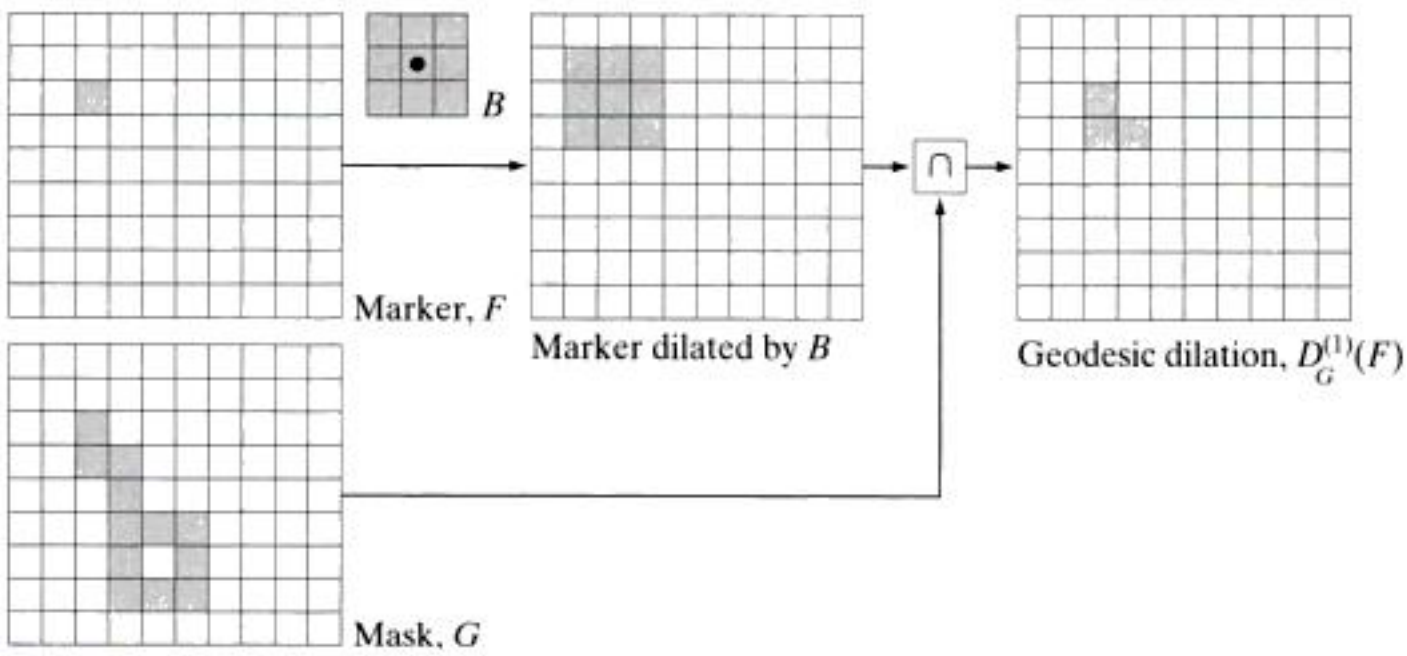


FIGURE 9.26
Illustration of geodesic dilation.

mask G will limit the growth (dilation) of marker F. Figure 9.26 shows a simple example of a geodesic dilation of size 1. The steps in the figure are a direct implementation of Eq. (9.5-21).

Similarly, the *geodesic erosion* of size 1 of marker F with respect to mask G is defined as

$$E_G^{(1)}(F) = (F \ominus B) \cup G \tag{9.5-23}$$

where $\cup$ denotes set union (or OR operation). The geodesic erosion of size n of F with respect to G is defined as

$$E_G^{(n)}(F) = E_G^{(1)}\left[E_G^{(n-1)}(F)\right] \tag{9.5-24}$$

with $E_G^{(0)}(F) = F$. The set union operation in Eq. (9.5-23) is performed at each iterative step, and guarantees that geodesic erosion of an image remains greater than or equal to its mask image. As expected from the forms in Eqs. (9.5-21) and (9.5-23), geodesic dilation and erosion are *duals* with respect to set complementation (see Problem 9.29). Figure 9.27 shows a simple example of geodesic erosion of size 1. The steps in the figure are a direct implementation of Eq. (9.5-23).

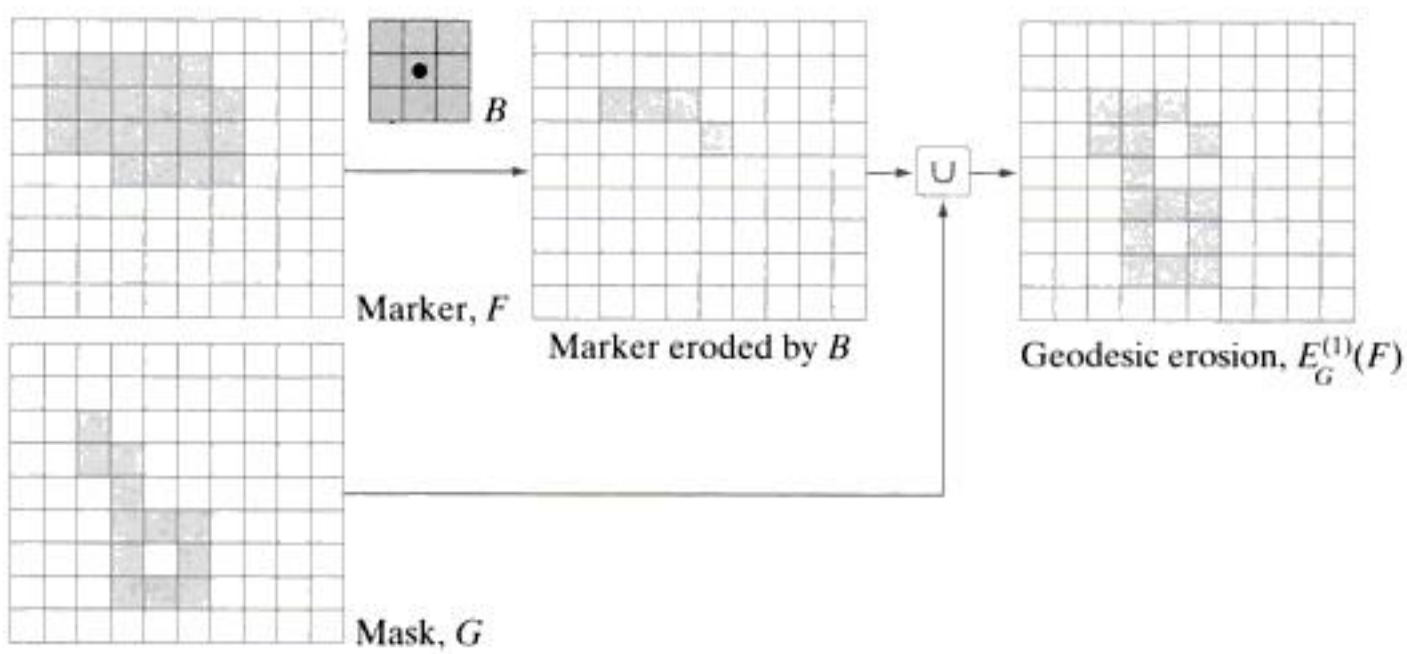


FIGURE 9.27
Illustration of geodesic erosion.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

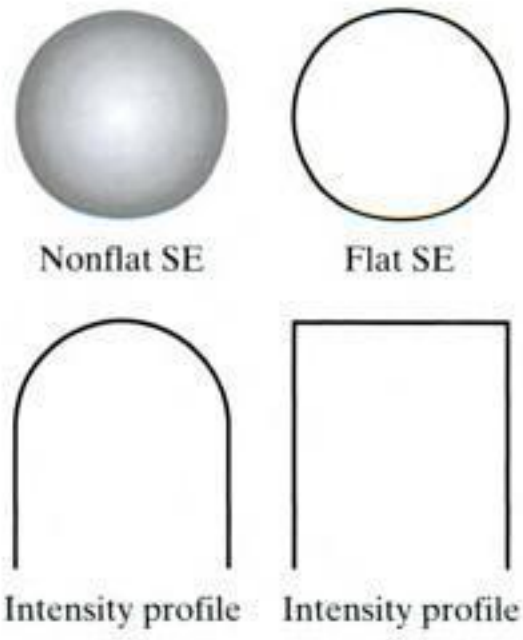
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

9.6 Gray-Scale Morphology

In this section, we extend to gray-scale images the basic operations of dilation, erosion, opening, and closing. We then use these operations to develop several basic gray-scale morphological algorithms.

Throughout the discussion that follows, we deal with digital functions of the form $f(x, y)$ and $b(x, y)$, where $f(x, y)$ is a gray-scale image and $b(x, y)$ is a structuring element. The assumption is that these functions are discrete in the sense introduced in Section 2.4.2. That is, if Z denotes the set of real integers, then the coordinates (x, y) are integers from the Cartesian product Z^2 and f and b are functions that assign an intensity value (a real number from the set of real numbers, R) to each distinct pair of coordinates (x, y). If the intensity levels are integers also, then Z replaces R.

Structuring elements in gray-scale morphology perform the same basic functions as their binary counterparts: They are used as "probes" to examine a given image for specific properties. Structuring elements in gray-scale morphology belong to one of two categories: *nonflat* and *flat*. Figure 9.34 shows an example of each. Figure 9.34(a) is a hemispherical gray-scale SE shown as an image, and Fig. 9.34(c) is a horizontal intensity profile through its center. Figure 9.34(b) shows a flat structuring element in the shape of a disk and Fig. 9.34(d) is its corresponding intensity profile (the shape of this profile explains the origin of the word "flat"). The elements in Fig. 9.34 are shown as continuous quantities for clarity; their computer implementation is based on digital approximations (e.g., see the rightmost disk SE in Fig. 9.2). Due to a number of difficulties discussed later in this section, gray-scale SEs are used infrequently in practice. Finally, we mention that, as in the binary case, the origin of structuring elements must be clearly identified. Unless mentioned otherwise, all the examples in this section are based on symmetrical, flat structuring elements of unit height whose origins are at the center. The *reflection* of an SE in gray-scale morphology is as defined in Section 9.1, and we denote it in the following discussion by $\hat{b}(x, y) = b(-x - y)$.



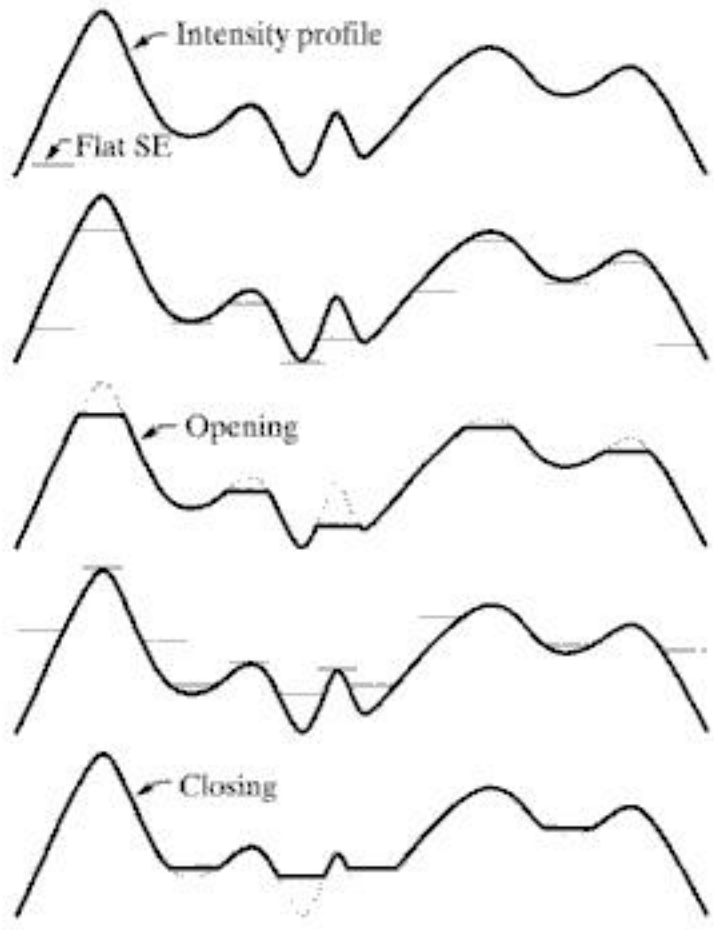
a b
c d

FIGURE 9.34 Nonflat and flat structuring elements, and corresponding horizontal intensity profiles through their center. All examples in this section are based on flat SEs.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a
b
c
d
e

FIGURE 9.36
Opening and closing in one dimension. (a) Original 1-D signal. (b) Flat structuring element pushed up underneath the signal. (c) Opening. (d) Flat structuring element pushed down along the top of the signal. (e) Closing.

Figure 9.36(d) is a graphical illustration of closing. Observe that the structuring element is pushed down on top of the curve while being translated to all locations. The closing, shown in Fig. 9.36(e), is constructed by finding the lowest points reached by any part of the structuring element as it slides against the upper side of the curve.

The gray-scale opening operation satisfies the following properties:

(a) $f \circ b \hookleftarrow f$
(b) If $f_1 \hookleftarrow f_2$, then $(f_1 \circ b) \hookleftarrow (f_2 \circ b)$
(c) $(f \circ b) \circ b = f \circ b$

The notation $e \hookleftarrow r$ is used to indicate that the domain of e is a subset of the domain of r, and also that $e(x, y) \leq r(x, y)$ for any (x, y) in the domain of e.

Similarly, the closing operation satisfies the following properties:

(a) $f \hookleftarrow f \bullet b$
(b) If $f_1 \hookleftarrow f_2$, then $(f_1 \bullet b) \hookleftarrow (f_2 \bullet b)$
(c) $(f \bullet b) \bullet b = f \bullet b$

The usefulness of these properties is similar to that of their binary counterparts.

EXAMPLE 9.10: Illustration of gray-scale opening and closing.

Figure 9.37 extends to 2-D the 1-D concepts illustrated in Fig. 9.36. Figure 9.37(a) is the same image we used in Example 9.9, and Fig. 9.37(b) is the opening obtained using a disk structuring element of unit height and radius of 3 pixels. As expected, the intensity of all bright features decreased, depending on the sizes of the features relative to the size of the SE. Comparing this figure with Fig. 9.35(b), we see that, unlike the result of erosion, opening had negligible effect on the dark features of the image, and the effect on the background was negligible. Similarly, Fig. 9.37(c) shows the closing of the image with a disk of radius 5 (the small round

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

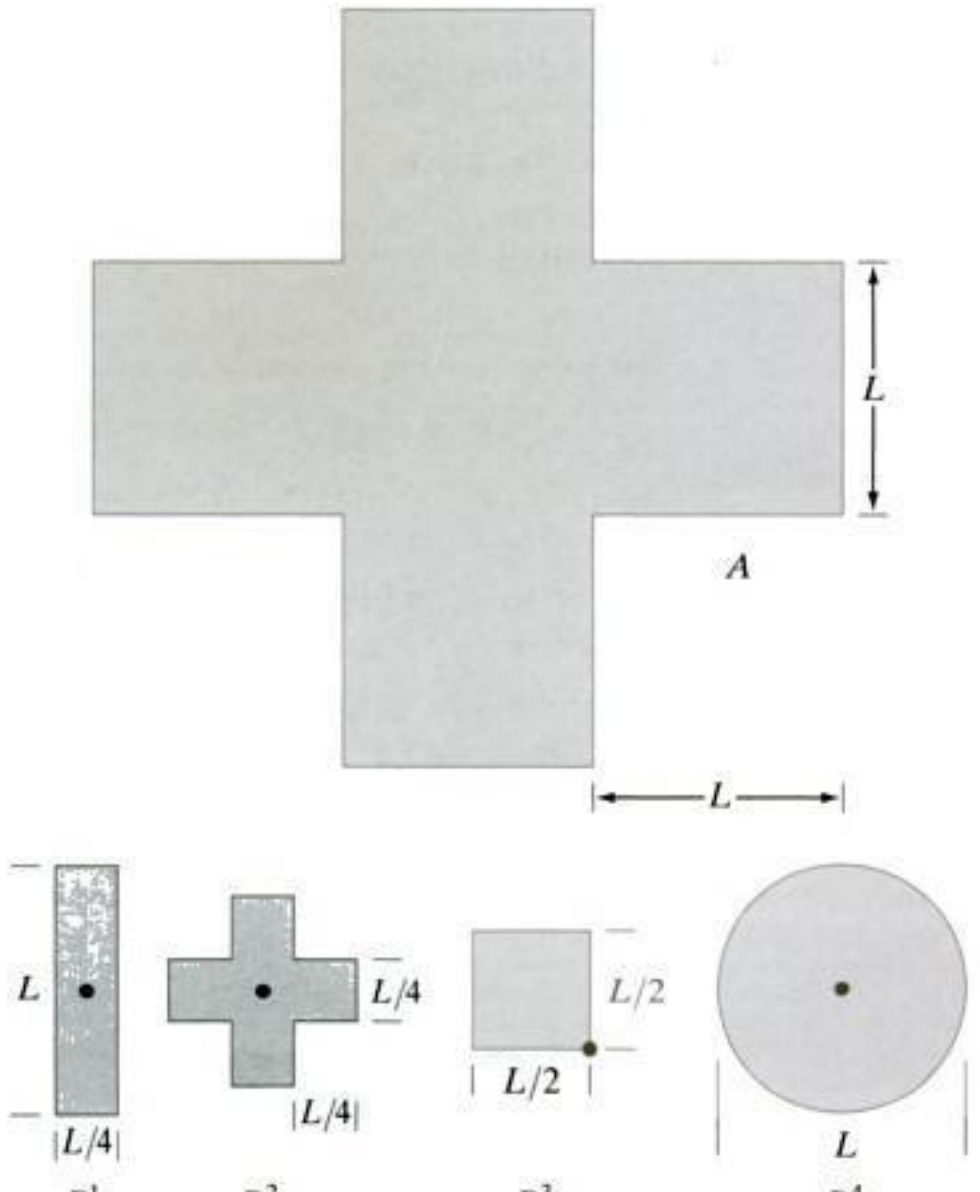
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



★**9.7** **(a)** What is the limiting effect of repeatedly dilating an image? Assume that a trivial (one point) structuring element is not used.

(b) What is the smallest image from which you can start in order for your answer in part (a) to hold?

9.8 **(a)** What is the limiting effect of repeatedly eroding an image? Assume that a trivial (one point) structuring element is not used.

(b) What is the smallest image from which you can start in order for your answer in part (a) to hold?

★**9.9** An alternative definition of erosion is

$$A \ominus B = \{w \in Z^2 | w + b \in A, \text{ for every } b \in B\}$$

Show that this definition is equivalent to the definition in Eq. (9.2-2).

9.10 **(a)** Show that the definition of erosion given in Problem 9.9 is equivalent to yet another definition of erosion:

$$A \ominus B = \bigcap_{b \in B} (A)_{-b}$$

(If $-b$ is replaced with b, this expression is called the *Minkowsky subtraction* of two sets.)

(b) Show that the expression in (a) also is equivalent to the definition in Eq. (9.2-2).

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

approach is to focus on selecting the types of sensors most likely to enhance the objects of interest while diminishing the contribution of irrelevant image detail. A good example is the use of infrared imaging by the military to detect objects with strong heat signatures, such as equipment and troops in motion.

Most of the segmentation algorithms in this chapter are based on one of two basic properties of intensity values: discontinuity and similarity. In the first category, the approach is to partition an image based on abrupt changes in intensity, such as edges. The principal approaches in the second category are based on partitioning an image into regions that are similar according to a set of predefined criteria. Thresholding, region growing, and region splitting and merging are examples of methods in this category. In this chapter, we discuss and illustrate a number of these approaches and show that improvements in segmentation performance can be achieved by combining methods from distinct categories, such as techniques in which edge detection is combined with thresholding. We discuss also image segmentation based on morphology. This approach is particularly attractive because it combines several of the positive attributes of segmentation based on the techniques presented in the first part of the chapter. We conclude the chapter with a brief discussion on the use of motion cues for segmentation.

See Sections 6.7 and 10.3.8 for a discussion of segmentation techniques based on more than just gray (intensity) values.

10.1 Fundamentals

Let R represent the entire spatial region occupied by an image. We may view image segmentation as a process that partitions R into n subregions, $R_1, R_2, \ldots, R_n$, such that

(a) $\bigcup_{i=1}^{n} R_i = R$.

(b) R_i is a connected set, $i = 1, 2, \ldots, n$.

(c) $R_i \cap R_j = \varnothing$ for all i and j, $i \neq j$.

(d) $Q(R_i) = \text{TRUE}$ for $i = 1, 2, \ldots, n$.

(e) $Q(R_i \cup R_j) = \text{FALSE}$ for any adjacent regions R_i and R_j.

See Section 2.5.2 regarding connected sets.

Here, $Q(R_k)$ is a logical predicate defined over the points in set R_k, and $\varnothing$ is the null set. The symbols $\cup$ and $\cap$ represent set union and intersection, respectively, as defined in Section 2.6.4. Two regions R_i and R_j are said to be *adjacent* if their union forms a connected set, as discussed in Section 2.5.2.

Condition (a) indicates that the segmentation must be complete; that is, every pixel must be in a region. Condition (b) requires that points in a region be connected in some predefined sense (e.g., the points must be 4- or 8-connected, as defined in Section 2.5.2). Condition (c) indicates that the regions must be disjoint. Condition (d) deals with the properties that must be satisfied by the pixels in a segmented region—for example, $Q(R_i) = \text{TRUE}$ if all pixels in R_i have the same intensity level. Finally, condition (e) indicates that two adjacent regions R_i and R_j must be different in the sense of predicate Q.[†]

[†] In general, Q can be a compound expression such as, for example, $Q(R_i) = \text{TRUE}$ if the average intensity of the pixels in R_i is less than m_i, AND if the standard deviation of their intensity is greater than σ_i, where m_i and σ_i are specified constants.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

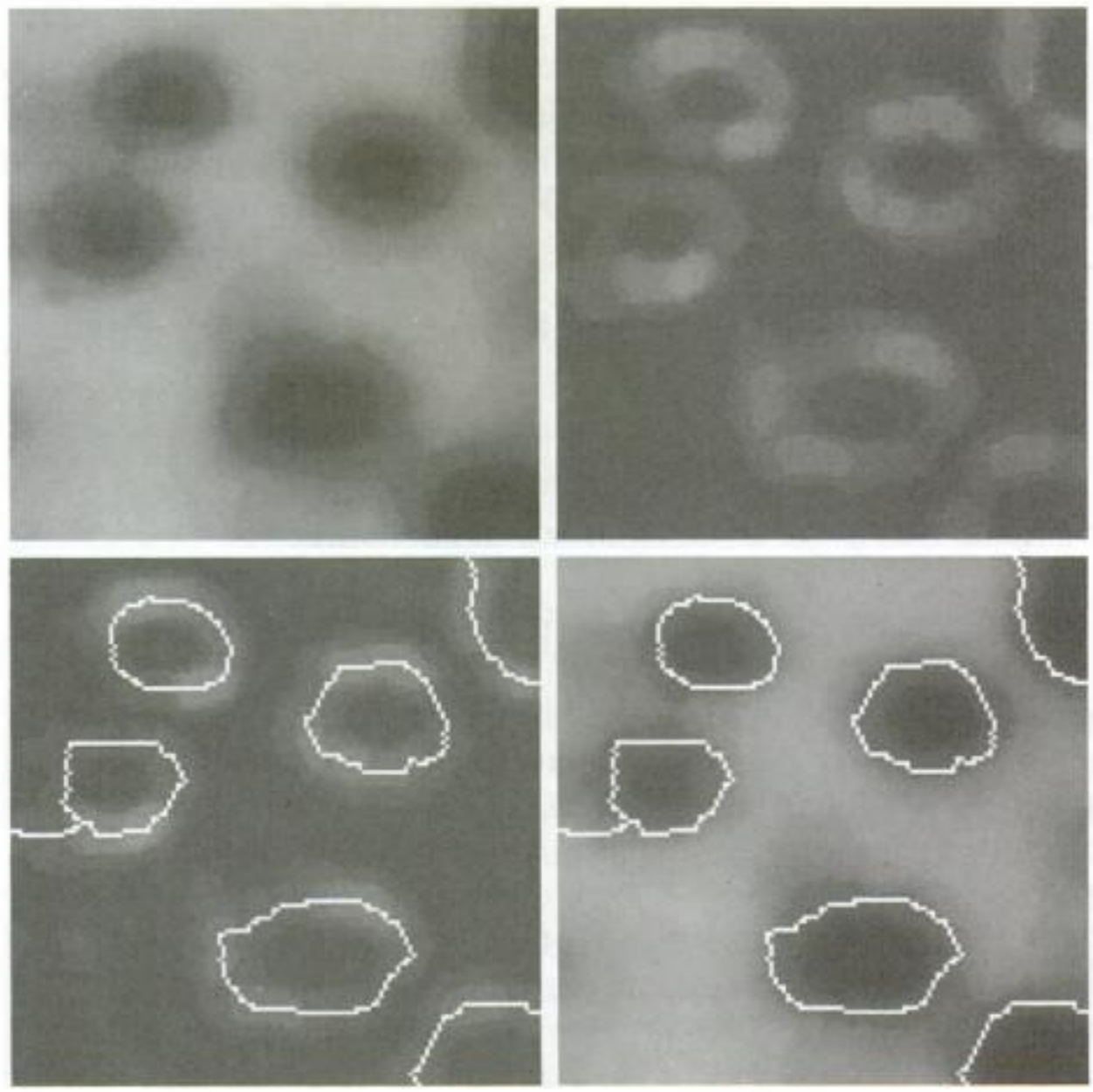
You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.



a b
c d

FIGURE 10.56
(a) Image of blobs. (b) Image gradient. (c) Watershed lines. (d) Watershed lines superimposed on original image. (Courtesy of Dr. S. Beucher, CMM/Ecole des Mines de Paris.)

EXAMPLE 10.25: Illustration of the watershed segmentation algorithm.

■ Consider the image and its gradient in Figs. 10.56(a) and (b), respectively. Application of the watershed algorithm just described yielded the watershed lines (white paths) of the gradient image in Fig. 10.56(c). These segmentation boundaries are shown superimposed on the original image in Fig. 10.56(d). As noted at the beginning of this section, the segmentation boundaries have the important property of being connected paths. ■

10.5.4 The Use of Markers

Direct application of the watershed segmentation algorithm in the form discussed in the previous section generally leads to *oversegmentation* due to noise and other local irregularities of the gradient. As Fig. 10.57 shows, oversegmentation can be serious enough to render the result of the algorithm virtually useless. In this case, this means a large number of segmented regions. A practical solution to this problem is to limit the number of allowable regions by incorporating a preprocessing stage designed to bring additional knowledge into the segmentation procedure.

An approach used to control oversegmentation is based on the concept of markers. A *marker* is a connected component belonging to an image. We have *internal* markers, associated with objects of interest, and *external* markers, associated with the background. A procedure for marker selection typically will consist of two principal steps: (1) preprocessing; and (2) definition of a set of criteria that markers must satisfy. To illustrate, consider Fig. 10.57(a) again.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

made. The difference between two images in a dynamic imaging problem has the tendency to cancel all stationary components, leaving only image elements that correspond to noise and to the moving objects.

In practice, obtaining a reference image with only stationary elements is not always possible, and building a reference from a set of images containing one or more moving objects becomes necessary. This applies particularly to situations describing busy scenes or in cases where frequent updating is required. One procedure for generating a reference image is as follows. Consider the first image in a sequence to be the reference image. When a nonstationary component has moved completely out of its position in the reference frame, the corresponding background in the present frame can be duplicated in the location originally occupied by the object in the reference frame. When all moving objects have moved completely out of their original positions, a reference image containing only stationary components will have been created. Object displacement can be established by monitoring the changes in the positive ADI, as indicated in the preceding section.

EXAMPLE 10.27: Building a reference image.

■ Figures 10.60(a) and (b) show two image frames of a traffic intersection. The first image is considered the reference, and the second depicts the same scene some time later. The objective is to remove the principal moving objects in the reference image in order to create a static image. Although there are other smaller moving objects, the principal moving feature is the automobile at the intersection moving from left to right. For illustrative purposes we focus on this object. By monitoring the changes in the positive ADI, it is possible to determine the initial position of a moving object, as explained previously. Once the area occupied by this object is identified, the object can be removed from the image by subtraction. By looking at the frame in the sequence at which the positive ADI stopped changing, we can copy from this image the area previously occupied by the moving object in the initial frame. This area then is pasted onto the image from which the object was cut out, thus restoring the background of that area. If this is done for all moving objects, the result is a reference image with only static components against which we can compare subsequent frames for motion detection. The result of removing the eastbound moving vehicle in this case is shown in Fig. 10.60(c). ■

a b c

FIGURE 10.60 Building a static reference image. (a) and (b) Two frames in a sequence. (c) Eastbound automobile subtracted from (a) and the background restored from the corresponding area in (b). (Jain and Jain.)

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

Segmentation by watersheds was shown in Section 10.5 to be a powerful concept. Early references dealing with segmentation by watersheds are Serra [1988], Beucher [1990], and Beucher and Meyer [1992]. The paper by Baccar et al. [1996] discusses segmentation based on data fusion and morphological watersheds. Progress ten years later is evident in a special issue of *Pattern Recognition* [2000], devoted entirely to this topic. As indicated in our discussion in Section 10.5, one of the key issues with watersheds is the problem of over segmentation. The papers by Najmanand and Schmitt [1996], Haris et al. [1998], and Bleau and Leon [2000] are illustrative of approaches for dealing with this problem. Bieniek and Moga [2000] discuss a watershed segmentation algorithm based on connected components.

The material in Section 10.6.1 is from Jain, R. [1981]. See also Jain, Kasturi, and Schunck [1995]. The material in Section 10.6.2 is from Rajala, Riddle, and Snyder [1983]. See also the papers by Shariat and Price [1990] and by Cumani et al. [1991]. The books by Sonka et al. [1999], Shapiro and Stockman [2001], Snyder and Qi [2004], and Davies [2005] provide additional reading on motion estimation. See also Alexiadis and Sergiadis [2007].

Problems

★**10.1** Prove the validity of Eq. (10.2-2). (*Hint:* Use a Taylor series expansion and keep only the linear terms.)

★**10.2** A binary image contains straight lines oriented horizontally, vertically, at 45°, and at −45°. Give a set of 5 × 5 masks that can be used to detect 1-pixel breaks in these lines. Assume that the intensities of the lines and background are 1 and 0, respectively.

10.3 Propose a technique for detecting gaps of length ranging between 2 and K pixels in line segments of a binary image. Assume that the lines are 1 pixel thick. Base your technique on 8-neighbor connectivity analysis, rather than attempting to construct masks for detecting the gaps.

10.4 Refer to Fig. 10.7 in answering the following questions.

★**(a)** Some of the lines joining the pads and center element in Fig. 10.7(e) are single lines, while others are double lines. Explain why.

(b) Propose a method for eliminating the components in Fig. 10.7(f) that are part of the line oriented at −45°.

10.5 Refer to the edge models in Fig. 10.8.

★**(a)** Suppose that we compute the gradient magnitude of each of these models using the Prewitt operators in Fig. 10.14. Sketch what a horizontal profile through the center of each gradient image would look like.

(b) Sketch a horizontal profile for each corresponding angle image.

(*Note:* Answer this question without generating the gradient and angle images. Simply provide sketches of the profiles that show what you would *expect* the profiles of the magnitude and angle images to look like.)

10.6 Consider a horizontal intensity profile through the middle of a binary image that contains a step edge running vertically through the center of the image. Draw what the profile would look like after the image has been blurred by an averaging mask of size $n \times n$, with coefficients equal to $1/n^2$. For simplicity, assume that the image was scaled so that its intensity levels are 0 on the left of the edge and 1 on its right. Also, assume that the size of the mask is much smaller than the image, so that image border effects are not a concern near the center of the horizontal intensity profile.

★**10.7** Suppose that we had used the edge models shown in the next page, instead of the ramp model in Fig. 10.10. Sketch the gradient and Laplacian of each profile.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

You have either reached a page that is unavailable for viewing or reached your viewing limit for this book.

DIGITAL IMAGE PROCESSING

THIRD EDITION

Rafael C. Gonzalez | Richard E. Woods

"A book that continues to builds on highly successful earlier editions and the author's twenty+ years of academic and industrial experience in image processing."

This edition of *Digital Image Processing* is a major revision and is based on the most extensive survey the authors have ever conducted. The survey involved faculty, students, and independent readers of the book in 134 institutions from 32 countries. The results have prompted the following new and reorganized material.

NEW TO THIS EDITION

- 80 New homework problems
- 400 New images, 200 new drawings and tables
- Redesigned and upgraded companion web site to correspond this new edition
- New chapters on Discrete Fourier Transform, Frequency Domain Processing, and Data Compression
- New coverage on Fuzzy Sets, Computerized Tomography,Morphological Reconstruction, Gray Scale Morphology, Advanced Morphological, Marr-Hildreth and Canny Edge Detection Algorithms
- Expanded coverage of Image Thresholding

Rafael C. Gonzalez is currently a Professor Emeritus of Electrical and Computer Engineering at University of Tennessee, Knoxville. Gonzalez is the founder of the Image & Pattern Analysis Laboratory and the Robotics & Computer Vision Laboratory at the University of Tennessee.

Richard E. Woods' professional experiences range from entrepreneurial to the more traditional academic, consulting, governmental, and industrial pursuits. He founded MedData Interactive. He was also a founder and Vice President of Perceptics Corporation.

ISBN 978-81-317-2695-2

PEARSON Prentice Hall

This edition is manufactured in India and is authorized for sale only in India, Bangladesh, Bhutan, Pakistan, Nepal, Sri Lanka and the Maldives.

9 788131 726952

www.pearsoned.co.in