1-Introduction

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.

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 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 of 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 human 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. 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.

1.2 The Origins of Digital Image Processing

Early 1920s: One of the first applications of digital imaging was in the news-paper industry

–The Bartlane cable picture transmission service
–Images were transferred by submarine cable between London and New York
–Pictures were coded for cable transfer and reconstructed at the receiving end on a telegraph printer

Mid to late 1920s: Improvements to the Bartlane system resulted in higher quality images

–New reproduction processes based on photographic techniques
–Increased number of tones in reproduced images

1960s: Improvements in computing technology and the onset of the space race led to a surge of work in digital image processing

–1964: Computers used to improve the quality of images of the moon taken by the Ranger 7probe

–Such techniques were used in other space missions including the Apollo landings.

1970s: Digital image processing begins to be used in medical applications

–1979: Sir Godfrey N. Hounsfield & Prof. Allan M. Cormack share the Nobel Prize in medicine for the invention of tomography, the technology behind Computerized Axial Tomography (CAT) scans

1980s -Today: The use of digital image processing techniques has exploded and they are now used for all kinds of tasks in all kinds of areas

–Image enhancement/restoration

–Artistic effects–Medical visualization

–Industrial inspection

–Law enforcement

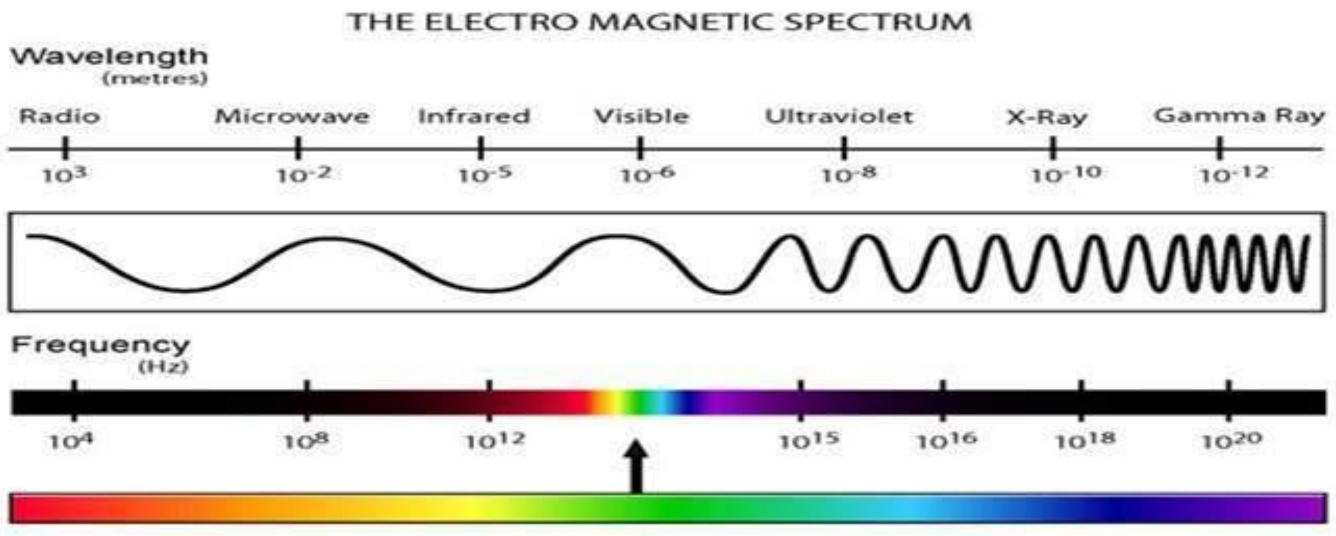
–Human computer interfaces.

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. 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. 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.

Digital Image processing is not just limited to adjust the spatial resolution of the everyday images captured by the camera. It is not just limited to increase the brightness of the photo, e.t.c. Rather it is far more than that.

Electromagnetic waves can be thought of as stream of particles, where each particle is moving with the speed of light. Each particle contains a bundle of energy. This bundle of energy is called a photon.



In this electromagnetic spectrum, we are only able to see the visible spectrum. Visible spectrum mainly includes seven different colors that are commonly term as (VIBGOYR).

But that does not nullify the existence of other stuff in the spectrum. Our human eye can only see the visible portion, in which we saw all the objects. But a camera can see the other things that a naked eye is unable to see. For example: x rays , gamma rays , etc.

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. Another major modality of nuclear imaging called *positron emission tomography* (PET). 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.

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 are also used extensively in industry and other areas, such as 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.

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.

Angiography is another major application in an area called contrast enhancement radiography. This procedure is used to obtain images of blood vessels, called *angiograms*. 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 the contrast of the blood vessels and enables a radiologist to see any irregularities or blockages.

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. 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.

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. Important tasks performed with a fluorescence microscope are 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).

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.

Another major area of visual processing is remote sensing, which usually includes several bands in the visual and infrared regions of the spectrum. The infrared system operates in the band 10.0 to , 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 relative percent of total electrical energy used by various regions of the world. A major area of imaging in the visible spectrum is in automated visual inspection of manufactured goods.

Imaging in the Microwave Band

The principal 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 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.

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 the individual's 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 image of a section of the patient. MRI can produce images in any plane.

Other Imaging Modalities

Although imaging in the electromagnetic spectrum is dominant by far, there are a number of other imaging modalities that are also 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 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 image 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.

Although ultrasound imaging is used routinely in manufacturing, the best known applications of this technique are in medicine, especially in obstetrics, where fetuses 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:

- The ultrasound system (a computer, ultrasound probe consisting of a source, a receiver, and a display) transmits high-frequency (1 to 5 MHz) sound pulses into the body.
- 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 are reflected.
- The reflected waves are picked up by the probe and relayed to the computer.
- 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.
- 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.

1.4 Fundamental Steps in Digital Image Processing

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.

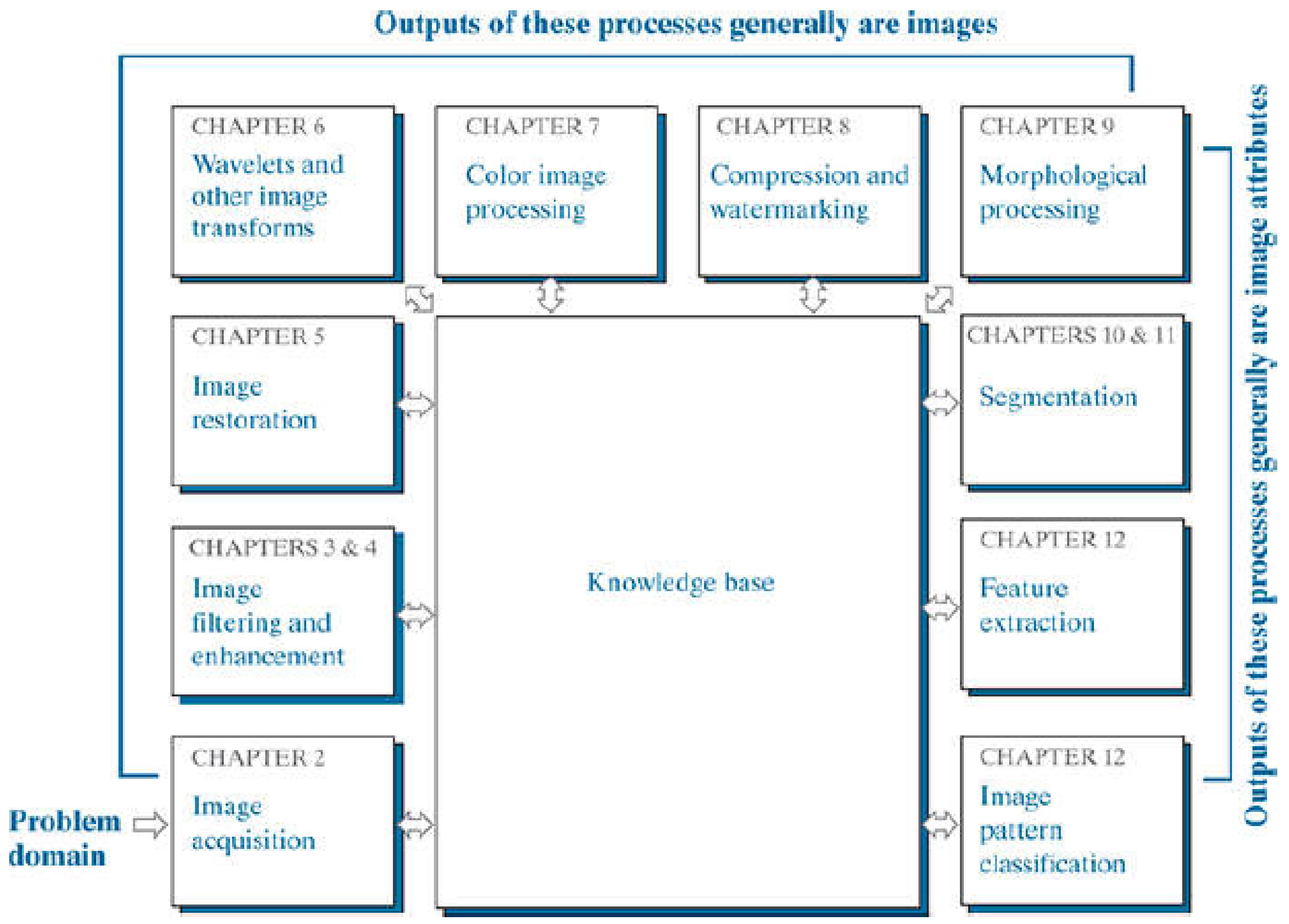


FIGURE 1.23

Fundamental steps in digital image processing. The chapter(s) indicated in the boxes is where discussed.

- ***Image acquisition*** is the first process in **Fig. 1.23.** 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 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. 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.

- ***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.
- ***Wavelets*** are the foundation for representing images in various degrees of resolution.
- ***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. Color is used also as the basis for extracting features of interest in an image.
- ***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.
- ***Segmentation*** partitions 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 automated object classification is to succeed.
- ***Feature extraction*** almost always follows 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. Feature extraction consists of feature detection and feature description. *Feature detection* refers to finding the features in an image, region, or boundary. *Feature description* assigns quantitative attributes to the detected features. For example, we might detect corners in a region, and describe those corners by their orientation and location; both of these descriptors are quantitative attributes.
- ***Image pattern classification*** is the process that assigns a label (e.g., “vehicle”) to an object based on its feature descriptors.

1.5 Components of an Image Processing System

Figure 1.24 shows the basic components comprising a typical general-purpose system used for digital image processing.

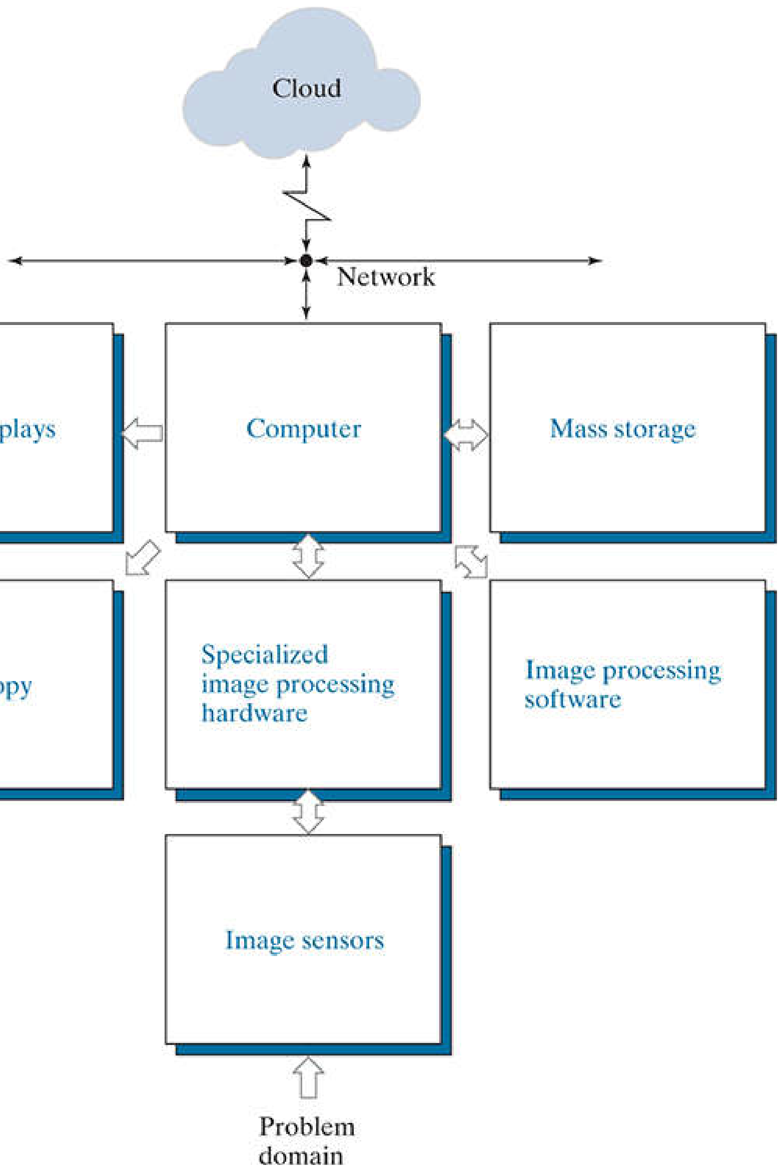


Figure 1.24 Components of a general-purpose image processing system.

Two subsystems are required to acquire digital images. The first is a physical ***sensor*** that responds to the energy radiated by the object we wish to image. The second, called a ***digitizer,*** is a device for converting the output of the physical sensing device into digital form. For instance, in a digital video camera, the sensors (CCD chips) produce an electrical output proportional to light intensity. The digitizer converts these outputs to digital data.

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.

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.

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. Commercially available image processing software, such as the well-known MATLAB® Image Processing Toolbox, is also common in a well-equipped image processing system.

Mass storage is a must in image processing applications. An image of size 1024 x 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 image databases that contain 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.
Image displays in use today are mainly color, flat screen 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 and GPUs available commercially as part of the computer system.

Hardcopy devices for recording images include laser printers, film cameras, heat-sensitive devices, ink-jet 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.

Networking and ***cloud*** communication are almost default functions 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, transmission bandwidth is improving quickly as a result of optical fiber and other broadband technologies. Image data compression continues to play a major role in the transmission of large amounts of image data.

2-Digital Image Fundamentals

2.1-Elements of Visual Perception

Although the field of digital image processing is built on a foundation of mathematics, human intuition and analysis often play a role in the choice of one technique versus another, and this choice often is made based on subjective, visual judgments.

Structure of the Human Eye

Figure 2.1 shows a simplified cross section of the human eye. The eye is nearly a sphere (with a diameter of about 20 mm) enclosed by three membranes: the ***cornea*** and ***sclera*** outer cover; the ***choroid',*** and the ***retina.***

The cornea is a tough, transparent tissue that covers the anterior surface of the eye. Continuous with the cornea, the sclera is an opaque membrane that encloses the remainder of the optic globe.

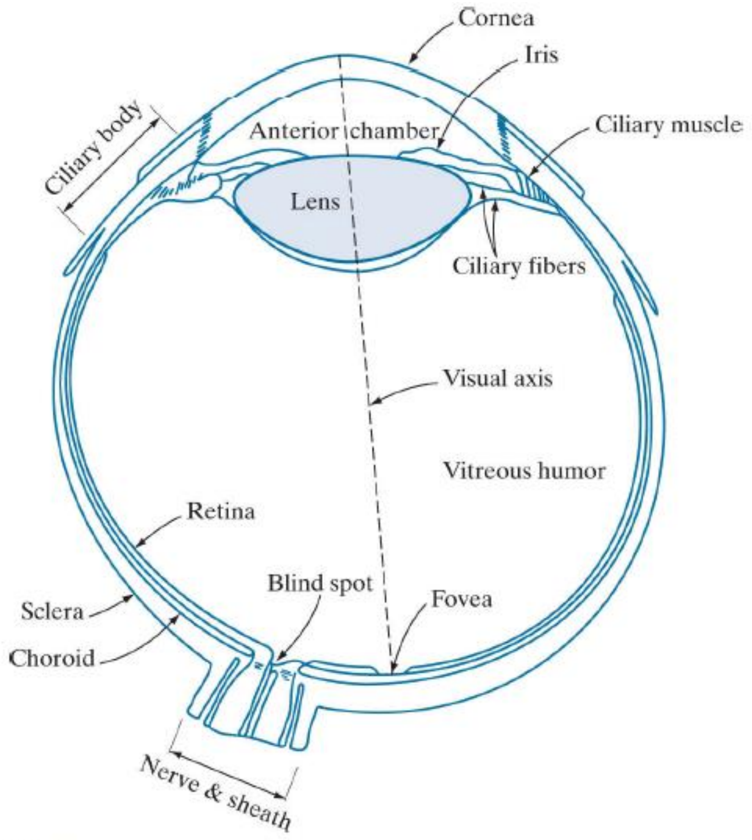


FIGURE 2.1
Simplified diagram of a cross section of the human eye.

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 can lead to severe eye damage as a result of inflammation that restricts blood flow. The choroid coat is heavily pigmented, which helps 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*** consists of concentric layers of fibrous cells and is suspended by fibers that attach to the ciliary body. It is composed of 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, 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 higher absorption at shorter wavelengths.
Both infrared and ultraviolet light are absorbed by proteins within the lens 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 focused, light from an object is imaged on the retina. Pattern vision is afforded by discrete light receptors distributed over the surface of the retina. There are two types of receptors: ***cones*** and ***rods.*** There are between 6 and 7 million cones in each eye. 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 because each cone is connected to its own nerve end. Muscles rotate the eye until the image of a region 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 retina. The larger area of distribution, and the fact that several rods are connected to a single nerve ending, reduces the amount of detail discernible by these receptors. Rods capture an overall image 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 appear as colorless forms in moonlight 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 where the optic nerve emerges from the eye. The absence of receptors in this area causes 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 visual axis.

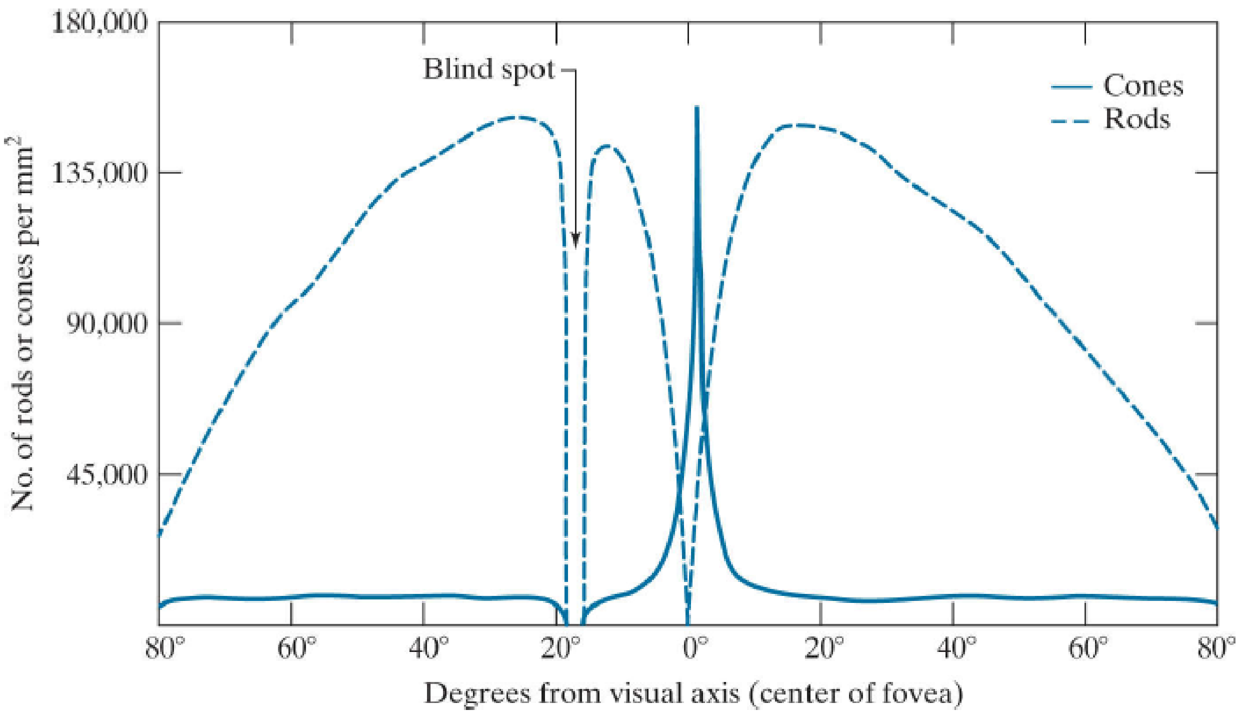


FIGURE 2.2
Distribution of rods and cones in the retina.

Image Formation in the Eye

In an ordinary photographic camera, the lens has a fixed focal length. 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 center of the lens and the imaging sensor (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 by flattening or thickening the lens for distant or near objects, respectively. The distance between the center of the lens and the retina along the visual axis is approximately 17 mm. The range of focal lengths is approximately 14 mm to 17 mm, the latter taking place when the eye is relaxed and focused at distances greater than about 3 m. The geometry in Fig. 2.3 illustrates how to obtain the dimensions of an image formed on the retina. For example, suppose that a person is looking at a tree 15 m high at a distance of 100 m. Letting ***h*** denote the height of that object in the retinal image, the geometry of Fig. 2.3 yields 15/100 = ***h***/17 or ***h*** = 2.5 mm. As indicated earlier in this section, the retinal image is focused primarily on the region of the fovea. Perception then takes place by the relative excitation of light receptors, which transform radiant energy into electrical impulses that ultimately are decoded by the brain.

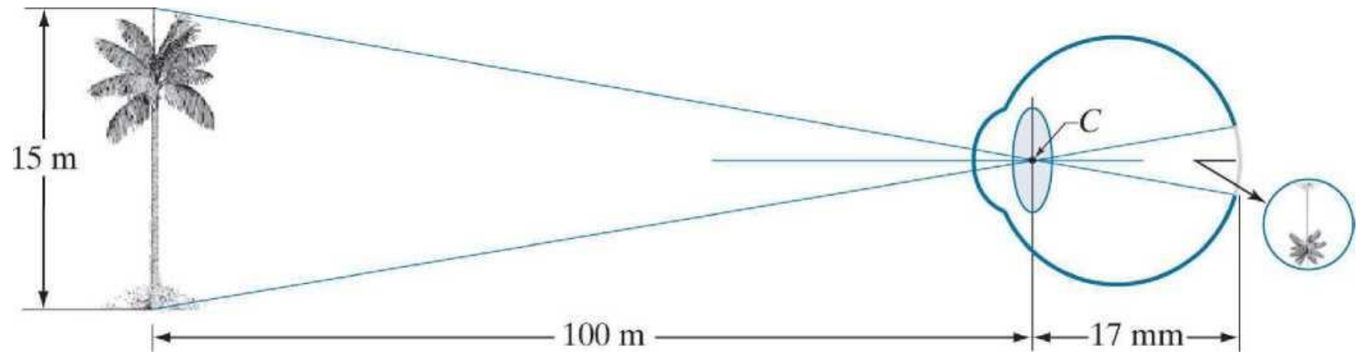


FIGURE 2.3 Graphical representation of the eye looking at a palm tree. Point C is the focal center of the lens.

Brightness Adaptation and Discrimination

Because digital images are displayed as sets of discrete intensities, the eye's ability to discriminate between different intensity levels is an important consideration in presenting image processing results. The range of light intensity levels to which the human visual system can adapt is enormous—on the order of 10^{10}— from the scotopic threshold to the glare limit. Experimental evidence indicates that ***subjective brightness*** (intensity as perceived by the human visual system) is a logarithmic function of the light intensity incident on the eye. Figure 2.4, a plot of light intensity versus subjective brightness, illustrates this characteristic. The long solid curve represents the range of intensities to which the visual system can adapt. In photopic vision alone, the range is about 10^{e}. The transition from scotopic to photopic vision is gradual over the approximate range from 0.001 to 0.1 millilambert (—3 to -1 mL in the log scale), as the double branches of the adaptation curve in this range show.

The key point in interpreting the impressive dynamic range depicted in Fig. 2.4 is that the visual system cannot operate over such a range ***simultaneously.*** Rather, it accomplishes this large variation by changing its overall sensitivity, a phenomenon known as ***brightness adaptation.*** The total range of distinct intensity levels the eye can discriminate simultaneously is rather small when compared with the total adaptation range. For a given set of conditions, the current sensitivity level of the visual system is called the ***brightness adaptation level,*** which may correspond, for example, to brightness B_a in Fig. 2.4. The short intersecting curve represents the range of subjective brightness that the eye can perceive when adapted to ***this*** level. This range is rather restricted, having a level B_b at, and below which, all stimuli are perceived as indistinguishable blacks. The upper portion of the curve is not actually restricted but, if extended too far, loses its meaning because much higher intensities would simply raise the adaptation level higher than B_a.

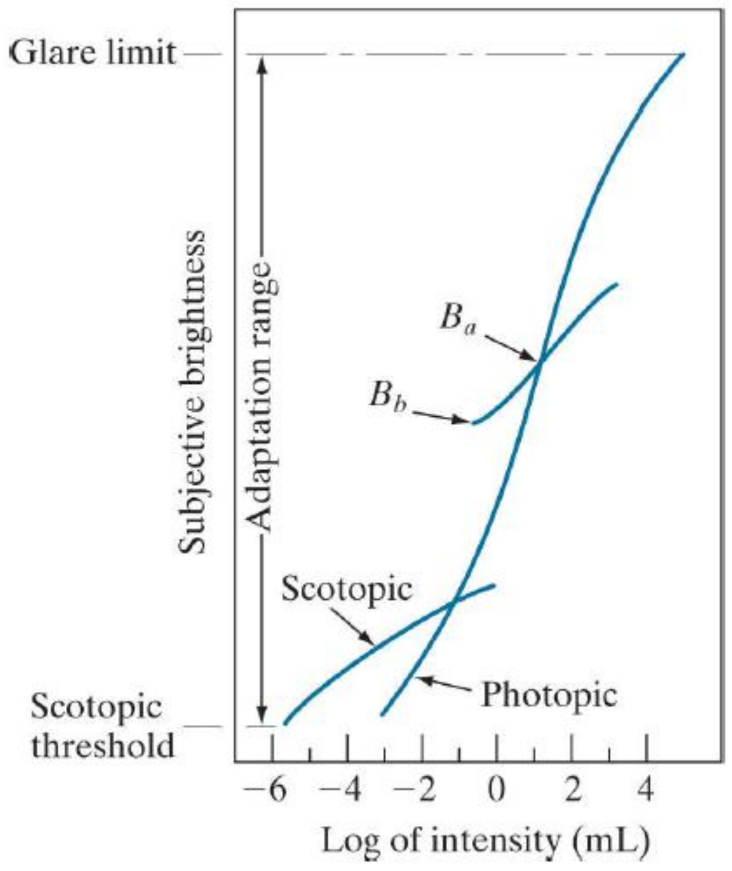


FIGURE 2.4
Range of subjective brightness sensations showing a particular adaptation level, B_a.

2.2-Light and the Electromagnetic Spectrum

Sir Isaac Newton discovered that when a beam of sunlight passes 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 is a 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.

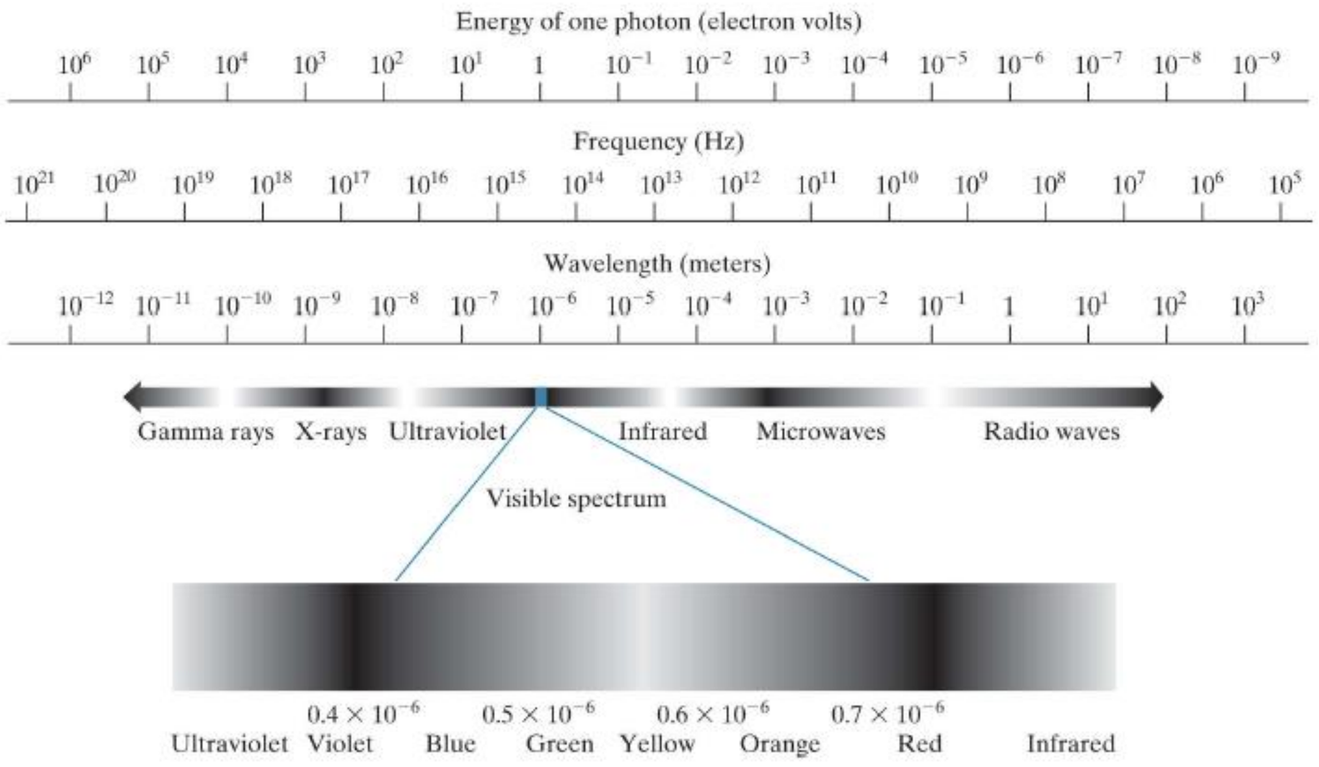


FIGURE 2.10
The electromagnetic spectrum. The visible spectrum is shown zoomed to facilitate explanations, but note that it encompasses a very narrow range of the total EM spectrum.

The electromagnetic spectrum can be expressed in terms of wavelength, frequency, or energy. Wavelength (λ) and frequency (ν) are related by the expression

$$\lambda = \frac{c}{\nu} \tag{2-1}$$

where c is the speed of light (2.998×10^8 m/s). Figure 2.11 shows a schematic representation of one wavelength.

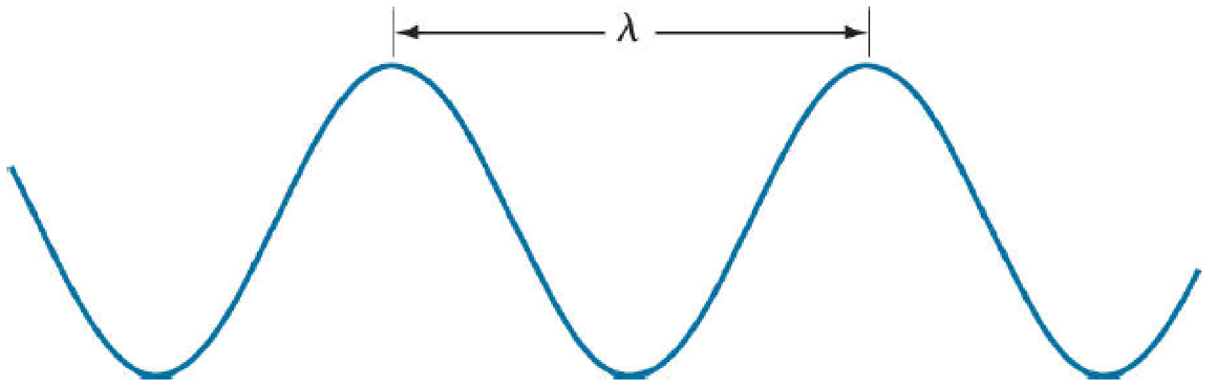


FIGURE 2.11
Graphical representation of one wavelength.

The energy of the various components of the electromagnetic spectrum is given by the expression

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

where *h* is Planck's constant. The units of wavelength are meters, with the terms *microns* (denoted /an 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 Hz 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 A (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, called a *photon.* We see from Eq. (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. High-energy electromagnetic radiation, especially in the X-ray and gamma ray bands, is particularly harmful to living organisms.

The colors perceived in an object are determined by the nature of the light *reflected* by 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.

Light that is void of color is called *monochromatic* (or *achromatic)* light. The only attribute of monochromatic light is its intensity. 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.

Chromatic (color) light spans the electromagnetic energy spectrum from approximately 0.43 to 0.79 *pm,* as noted previously. In addition to frequency, three other quantities are used to describe 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 (Im), gives a measure of the amount of energy an observer *perceives* from a light source.

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 familiar 3-D scene. For example, the illumination may originate from a source of electromagnetic energy, such as a 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 image. In some applications, the reflected or transmitted energy is focused onto a photo converter (e.g., a phosphor screen) that 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 incident energy into digital images. The idea is simple: Incoming energy is transformed into a voltage by a combination of the input electrical power and sensor material that is responsive to the type of energy being detected. The output voltage waveform is the response of the sensor, and a digital quantity is obtained by digitizing that response.

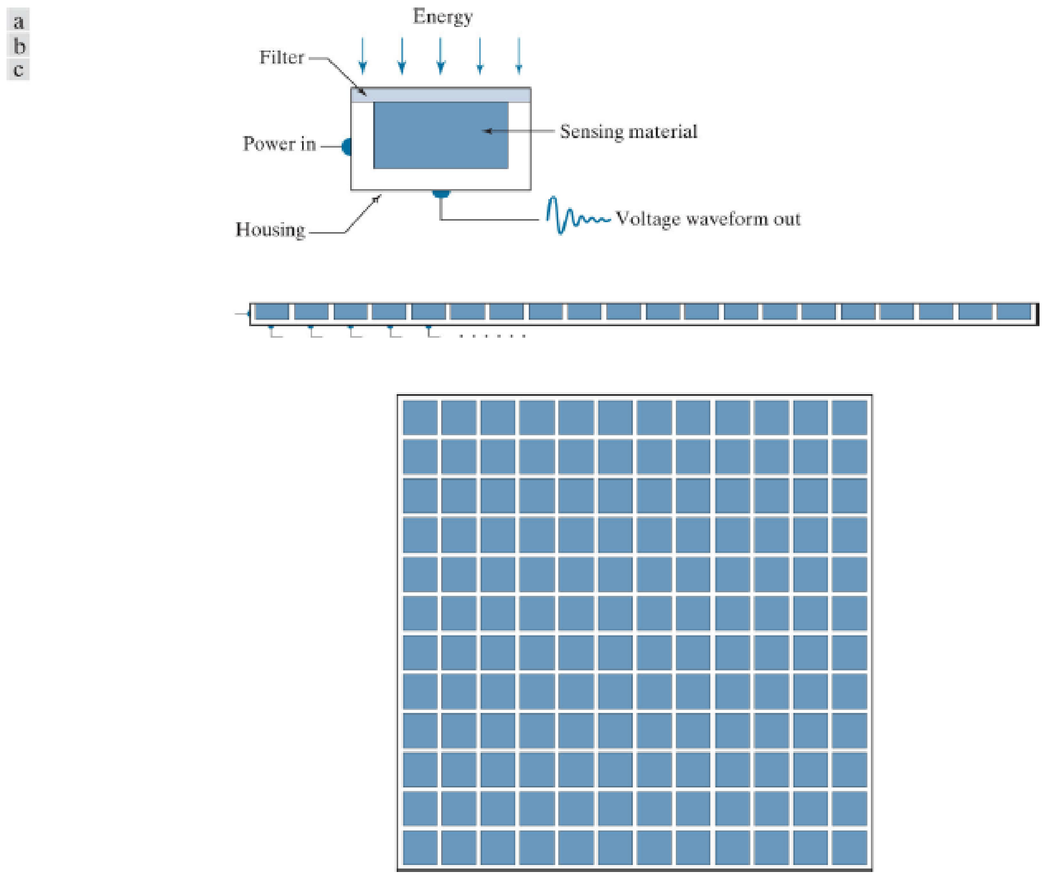


FIGURE 2.12
(a) Single sensing element. (b) Line sensor. (c) Array sensor.

Image Acquisition Using a Single Sensing Element

In order to generate a 2-D image using a single sensing element, 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 sensor is mounted on a lead screw that provides motion in the perpendicular direction. A light source is contained inside the drum. As the light passes through the film, its intensity is modified by the film density before it is captured by the sensor. This “modulation” of the light intensity causes corresponding variations in the sensor voltage, which are ultimately converted to image intensity levels by digitization.

This method is an inexpensive way to obtain high-resolution images because mechanical motion can be controlled with high precision. The main disadvantages of this method are that it is slow and not readily portable. Other similar mechanical arrangements use a flat imaging bed, with the sensor moving in two linear directions. These types of mechanical digitizers sometimes are referred to as ***transmission microdensitometers.*** Systems in which light is reflected from the medium, instead of passing through it, are called ***reflection microdensitometers.*** Another example of imaging with a single sensing element 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.

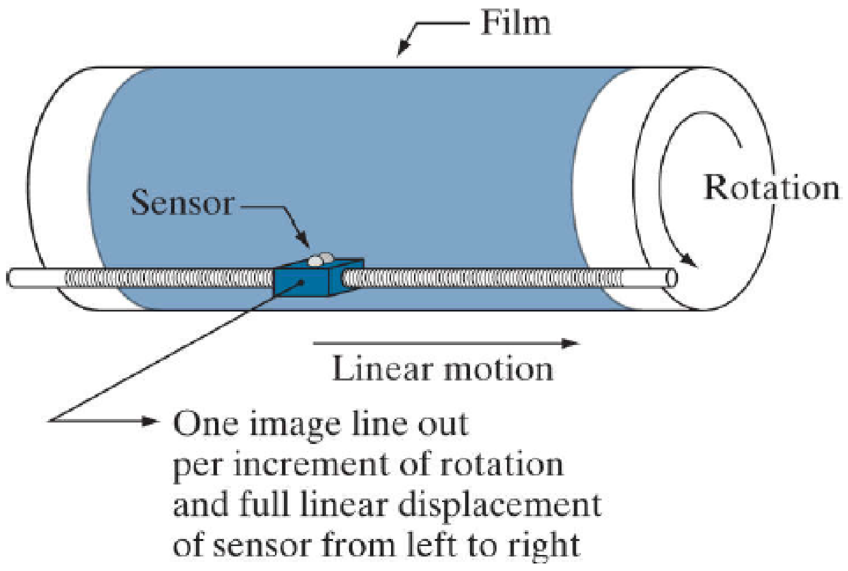

Film
Sensor
Rotation
Linear motion
One image line out
per increment of rotation
and full linear displacement
of sensor from left to right

FIGURE 2.13
Combining a single sensing element with mechanical motion to generate a 2-D image.

Image Acquisition Using Sensor Strips

A geometry used more frequently than single sensors is an in-line sensor strip, as in Fig. 2.12(b). 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 arrangement is 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 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. An imaging strip gives one line of an image at a time, and the motion of the strip relative to the scene completes the other dimension of a 2-D image. Lenses or other focusing schemes are used to project the area to be scanned onto the sensors.

a b

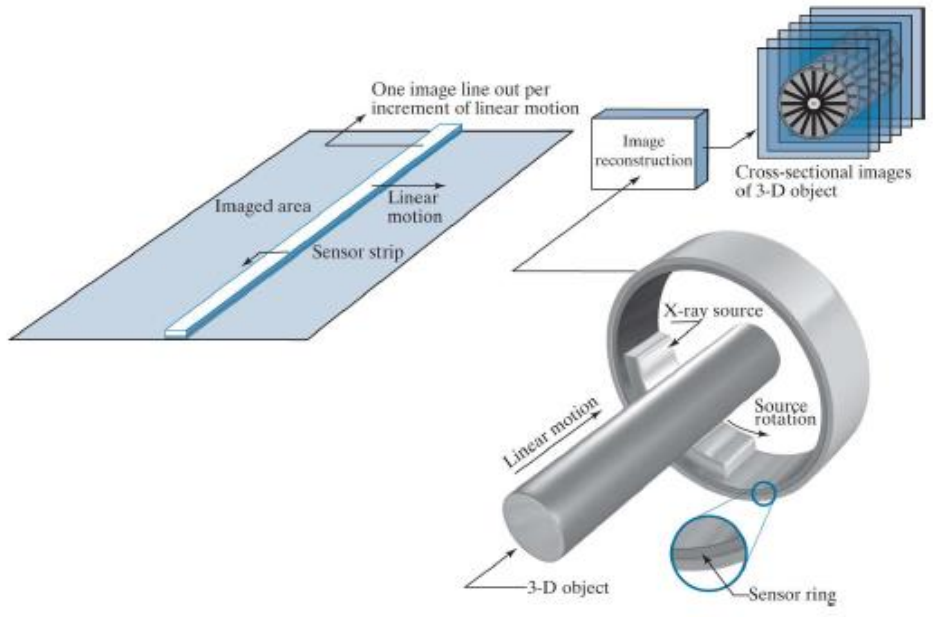
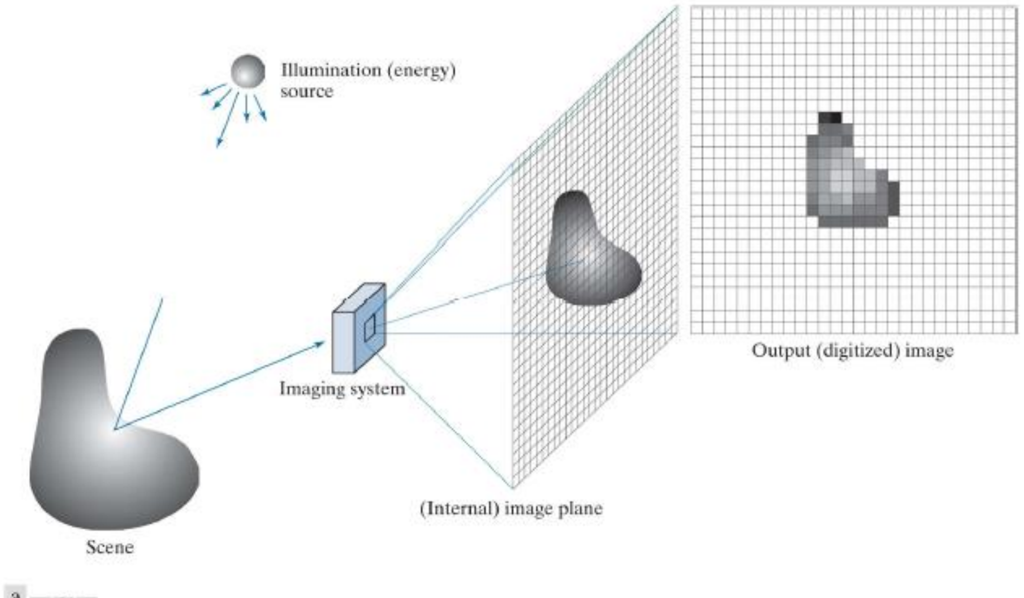


FIGURE 2.14
(a) Image acquisition using a linear sensor strip. (b) Image acquisition using a circular sensor strip.

Sensor strips 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 X-ray sensitive sensors opposite the source collect the energy that passes through the object. This is the basis for medical and industrial computerized axial tomography (CAT) imaging. The output of the sensors is processed by reconstruction algorithms whose objective is to transform the sensed data into meaningful cross-sectional images. In other words, images are not obtained directly from the sensors by motion alone; they also require extensive computer 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 their applications are very similar to the basic imaging approach shown in Fig. 2.14(b)

Image Acquisition Using Sensor Arrays

A typical sensor for cameras is a CCD (charge-coupled device) array, which can be manufactured with a broad range of sensing properties and can be packaged in rugged arrays of 4000 x 4000 elements or more. CCD sensors are used widely in digital cameras and other lightsensing 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.



a b c d e

FIGURE 2.15
An example of digital image acquisition. (a) Illumination (energy) source. (b) A scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

Figure 2.15 shows the principal manner in which array sensors are used. This figure shows the energy from an illumination source being reflected from a scene (as mentioned at the beginning of this section, the energy also could be transmitted through the scene). 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 focal plane of the lens, 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)**

A Simple Image Formation Model

We denote images by two-dimensional functions of the form *f(x,* y). The value of fat spatial coordinates (x, y) is a scalar quantity whose physical meaning is determined by the source of the image, and whose values are proportional to energy radiated by a physical source (e.g., electromagnetic waves). As a consequence, *f(x, y)* must be nonnegative and finite; that is,

Image intensities can become negative during processing, or as a result of interpretation. For example, in radar images, objects moving toward the radar 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.

$$0 <= f/(x,y) < \infty \qquad (2\text{-}3)$$

Function *f(x, y)* is 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):

$$f(x, y) = i(x, y)r(x, y) \qquad (2\text{-}4)$$

where

$$0 <= i(x,y) < \infty \qquad \mathbf{(2\text{-}5)}$$

and

$$0 < r(x, y) < 1 \qquad (2\text{-}6)$$

Thus, 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.

2.4-Image Sampling and Quantization

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 a digital format. This requires two processes: *sampling* and *quantization*.

Basic Concepts in Sampling and Quantization

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 digitize it, we have to sample the function in both coordinates and also in amplitude. Digitizing the coordinate values is called *sampling*. Digitizing the amplitude values is called *quantization*.

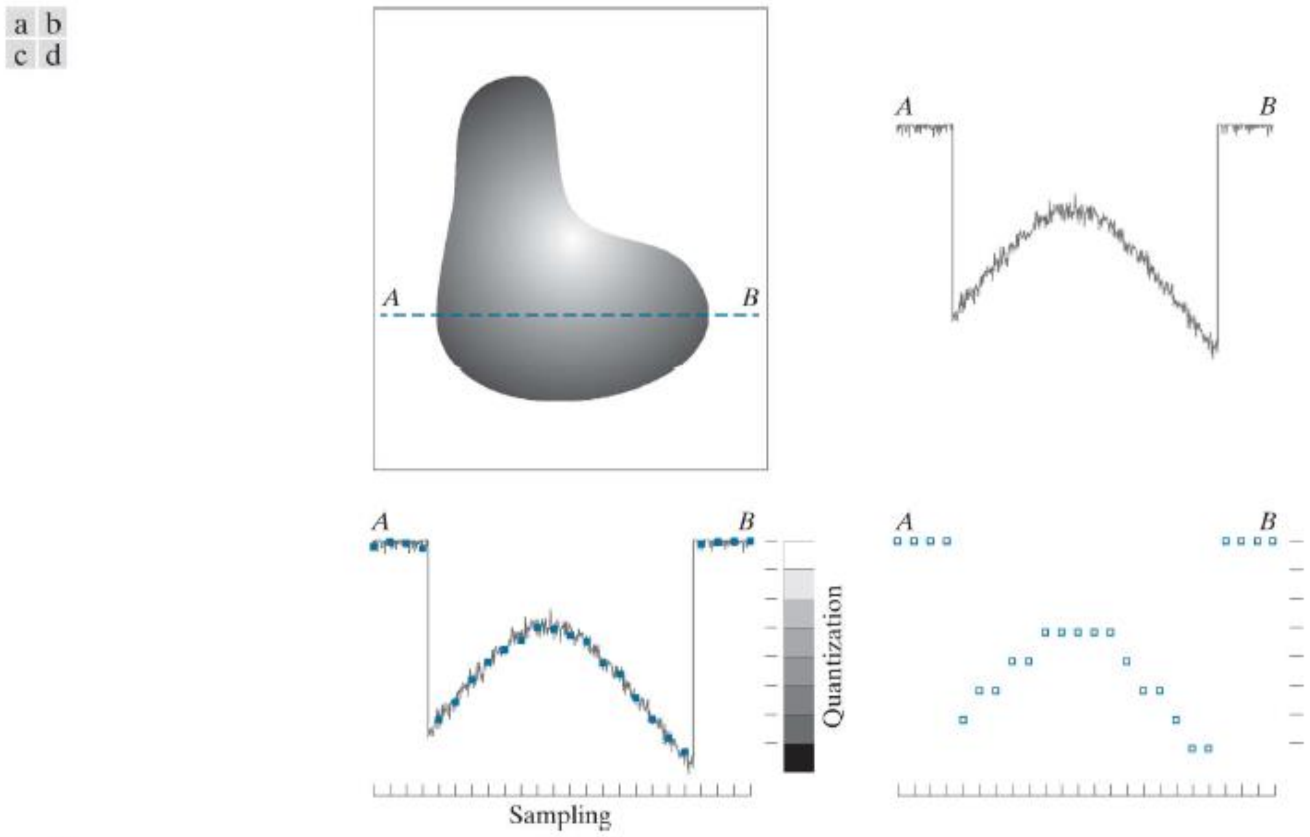


FIGURE 2.16
(a) Continuous image. (b) A scan line showing intensity variations along line *AB* in the continuous image. (c) Sampling and quantization. (d) Digital scan line. (The black border in (a) is included for clarity. It is not part of the image).

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 samples are shown as small dark squares superimposed on the function, and their (discrete) spatial locations are indicated by corresponding tick marks in the bottom of the figure. The set of dark squares constitute 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 vertical gray bar in Fig. 2.16(c) depicts 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, depending on the vertical proximity of a sample to a vertical tick mark. The digital samples resulting from both sampling and quantization are shown as white squares in Fig. 2.16(d). Starting at the top of the continuous image and carrying out this procedure downward, 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.

When a sensing array is used for image acquisition, no motion is required. The number of sensors in the array establishes the limits of sampling in both directions. Quantization of the sensor outputs is as explained above. **Figure 2.17** illustrates this concept. **Figure 2.17(a)** shows a continuous image projected onto the plane of a 2-D sensor. **Figure 2.17(b)** shows the image after sampling and quantization. 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 will show later in this section, image content also plays a role in the choice of these parameters.

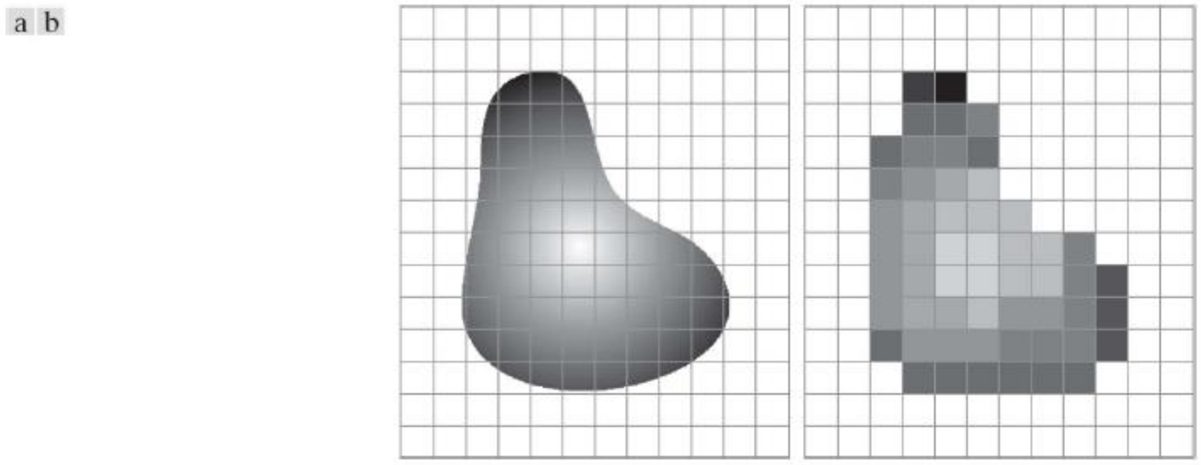


FIGURE 2.17
(a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.

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 digital image, 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,..., *M* -1 and y = 0,1,2,..., *N* -1. Thus, for example, the value of the digital image at the origin is *f(0,* 0), and its value at the next coordinates along the first row is *f(*0, 1). Here, the notation (0,1) is used to denote 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 a digital image at any coordinates (x, y) is denoted f(x, y), where x and y are integers. When we need to refer to specific coordinates we use the notation where the arguments 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.*

Figure 2.18 shows three ways of representing *t(x, y).* Figure 2.18(a) is a plot of the function, with two axes determining spatial location and the third axis being the values of f as a function of x and y. This representation is useful when working with grayscale sets whose elements are expressed as triplets of the form (x, y, z), where x and yare spatial coordinates and z is the value of fat coordinates (x, y).

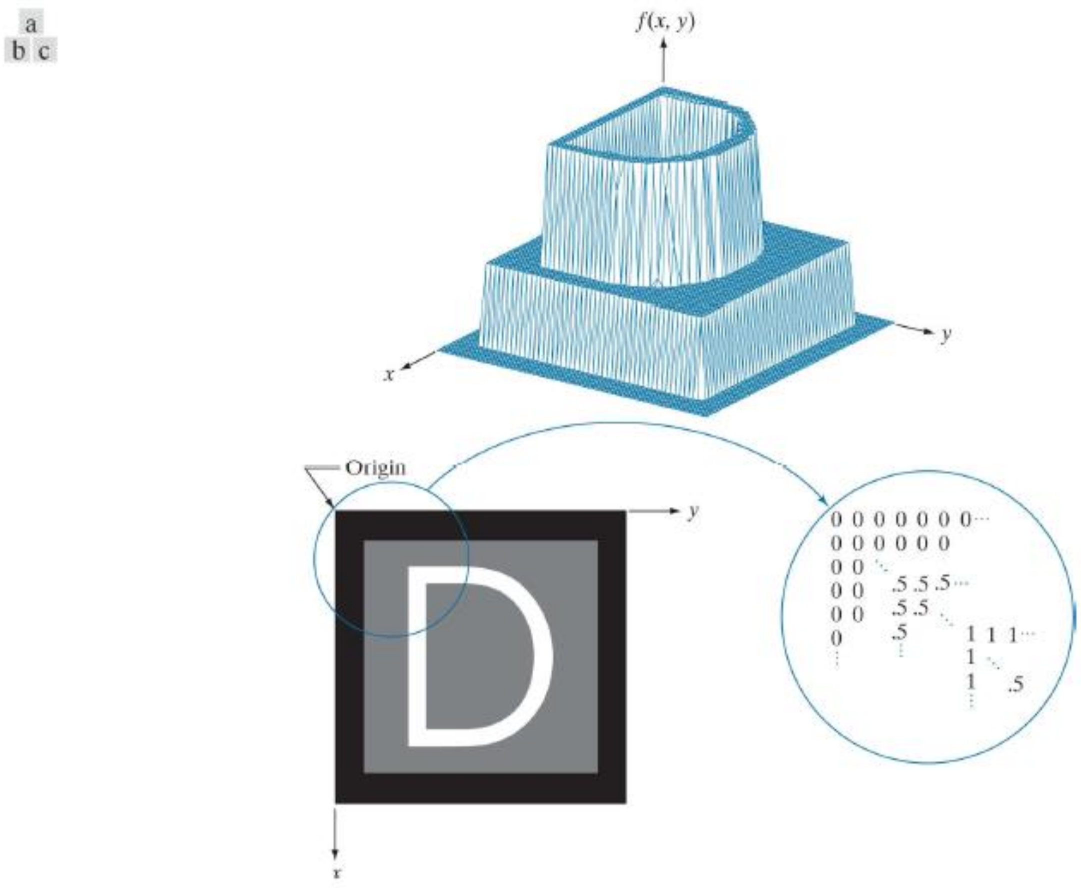


FIGURE 2.18
(a) Image plotted as a surface. (b) Image displayed as a visual intensity array. (c) Image shown as a 2-D numerical array. (The numbers 0, .5, and 1 represent black, gray, and white, respectively.)

The representation in Fig. 2.18(b) is more common, and it shows ***f(x, y)*** as it would appear on a computer display or photograph. Here, the intensity of each point in the display is proportional to the value of fat 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 converts these three values to black, gray, or white, respectively, as in Fig. 2.18(b). This type of representation includes color images, and allows us to view results at a glance.

As **Fig. 2.18(c)** shows, the third representation is an array (matrix) composed of the numerical values of *f(x, y)*. This is the representation used for computer processing. In equation form, we write the representation of an *M* x *N* numerical array as

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \quad (2\text{-}9)$$

The right side of this equation is a digital image represented as an array of real numbers. Each element of this array is called an *image element*, *picture element*, *pixel*, or *pel*. We use the terms *image* and *pixel* throughout the book to denote a digital image and its elements. **Figure 2.19** shows a graphical representation of an image array, where the *x*- and *y*-axis are used to denote the rows and columns of the array. Specific pixels are values of the array at a fixed pair of coordinates. As mentioned earlier, we generally use $f(i, j)$ when referring to a pixel with coordinates (i, j).

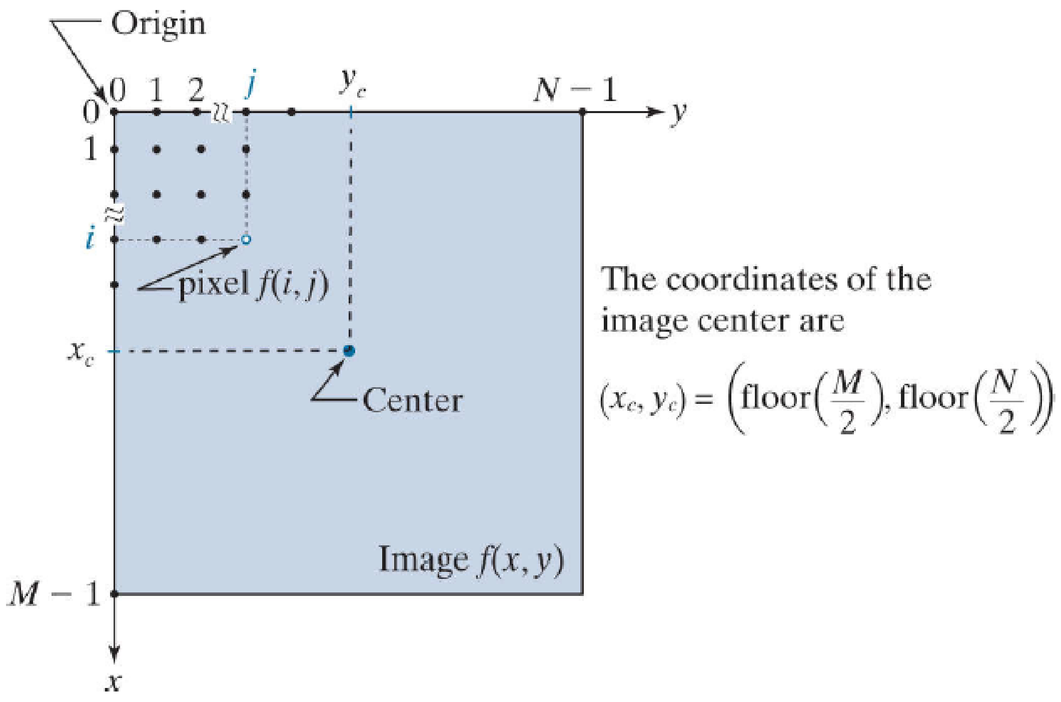

Origin
0 1 2 j y_c $N-1$
y
0
1
i
x_c
$M-1$
x
pixel $f(i, j)$
Center
Image $f(x, y)$
The coordinates of the image center are
$(x_c, y_c) = \left(\text{floor}\left(\frac{M}{2}\right), \text{floor}\left(\frac{N}{2}\right)\right)$

FIGURE 2.19

Linear vs. Coordinate Indexing

The convention discussed in the previous section, in which the location of a pixel is given by its 2-D coordinates, is referred to as *coordinate indexing*, or *subscript indexing*. Another type of indexing used extensively in programming image processing algorithms is *linear indexing*, which consists of a 1-D string of nonnegative integers based on computing offsets from coordinates (0, 0). There are two principal types of linear indexing, one is based on a row scan of an image, and the other on a column scan.

Figure 2.22 illustrates the principle of linear indexing based on a column scan. The idea is to scan an image column by column, starting at the origin and proceeding down and then to the right. The linear index is based on counting pixels as we scan the image in the manner shown in Fig. 2.22 . Thus, a scan of the first (leftmost) column yields linear indices 0 through M — 1. A scan of the second column yields indices M through $2M$ — 1, and so on, until the last pixel in the last column is assigned the linear index value MN - 1. Thus, a linear index, denoted by *a*, has one of MN possible values: 0,1,2,.... MN - 1, as Fig. 2.22 shows. The important thing to notice here is that each pixel is assigned a linear index value that identifies it uniquely.

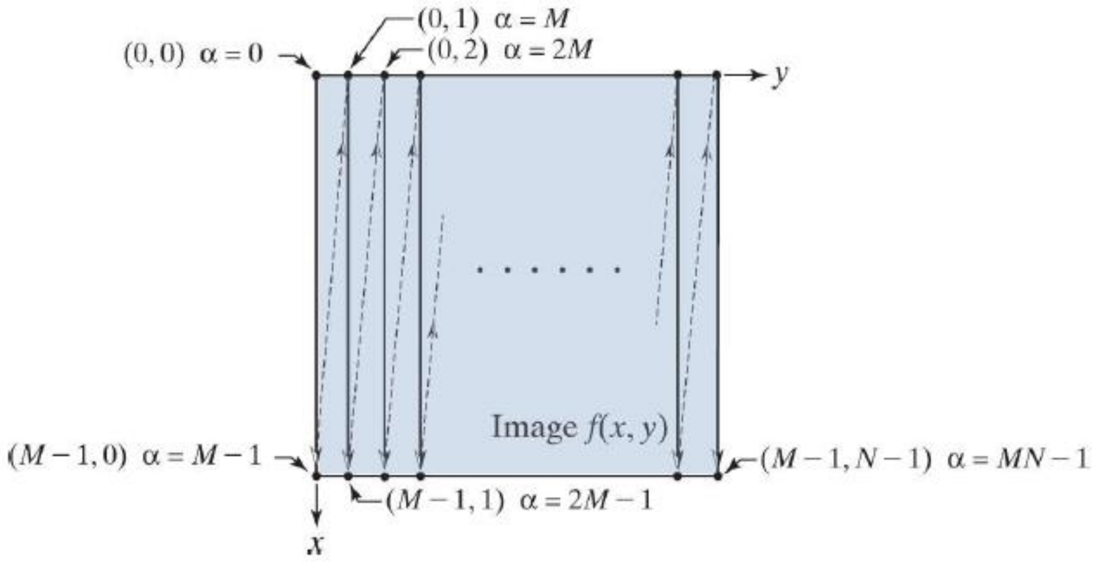


FIGURE 2.22

The formula for generating linear indices based on a column scan is straightforward and can be determined by inspection. For any pair of coordinates (x, y), the corresponding linear index value is

$$a = My + x \qquad (2\text{-}14)$$

Conversely, the coordinate indices for a given linear index value *a* are given by the equations
When working with modular number systems, it is more accurate to write *x* = *a* imd *M*, where the symbol = means *congruence.* However, our interest here is just on converting from linear to coordinate indexing, so we use the more familiar equal sign.

$$x = \alpha \ mod \ M \qquad (2\text{-}15)$$

and

$$y = (\alpha - x)/M \qquad (2\text{-}16)$$

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 several ways, with *line pairs per unit distance*, and *dots (pixels) per unit distance* being 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 2 *W*, and there are *WI2* 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 (i.e., per 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 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 was printed at 2400 dpi.

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. For example, to say that an image has a resolution of 1024 x 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 instance, 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 spatial samples (pixels) 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. The most common number is 8 bits, with 16 bits being used in some applications in which enhancement of specific intensityranges 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 not as common.

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. However, keep in mind that *discernible* changes in intensity are influenced also by noise and saturation values, and by the capabilities of human perception to analyze and interpret details in the context of an entire scene.

Image Interpolation

Interpolation is used in tasks such as zooming, shrinking, rotating, and geometrically correcting digital images. 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.

Interpolation is the process of using known data to estimate values at unknown locations. We begin the discussion of this topic with a short example. Suppose that an image of size 500 x 500 pixels has to be enlarged 1.5 times to 750 x 750 pixels. A simple way to visualize zooming is to create an imaginary 750 x 750 grid with the same pixel spacing as the original image, then shrink it so that it exactly overlays the original image. Obviously, the pixel spacing in the shrunken 750 x 750 grid will be less than the pixel spacing in the original image. To assign an intensity value to any point in the overlay, we look for its closest pixel in the underlying original image and assign the intensity of that pixel to the new pixel in the 750 x 750 grid. When intensities have been assigned to all the points in the overlay grid, we expand it back to the specified size to obtain the resized 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. This approach is simple but, it has the tendency to produce undesirable artifacts, such as severe distortion of straight edges. 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.

$$v(x, y) = ax + by + cxy + d \qquad \textbf{(2-17)}$$

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). 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

$$v(x,y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j \tag{2-18}$$

The sixteen coefficients are determined from the sixteen equations with sixteen unknowns that can be written using the sixteen nearest neighbors of point (x, y). Observe that Eq. (2-18) reduces in form to Eq. (2-17) 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 applications, such as Adobe Photoshop and Corel Photopaint.

2.5 Some Basic Relationships Between Pixels

Neighbors of a Pixel

A pixel p at coordinates (x, y) has two horizontal and two vertical neighbors with coordinates

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

This set of pixels, called the *4-neighbors* of p, is denoted $N_4(p)$.

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 $N_D(p)$. These neighbors, together with the 4-neighbors, are called the *8-neighbors* of p, denoted by $N_a(p)$. The set of image locations of the neighbors of a point p is called the *neighborhood* of p. The neighborhood is said to be *closed* if it contains p. Otherwise, the neighborhood is said to be *open.*

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 grayscale image, the idea is the same, but set V typically contains more elements. For example, if we are dealing with the adjacency of pixels whose values are in the range 0 to 255, set *V* could be any subset of these 256 values. We consider three types of adjacency:

1. 4-*adjacency.* Two pixels p and *q* with values from *V* are 4-adjacent if *q* is in the set W_4(p).
2. *8-adjacency.* Two pixels p and *q* with values from *V*are 8-adjacent if *q* is in the set $N_8(p)$.
3. m-adjacency (also called *mixed adjacency).* Two pixels p and *q* with values from *V* are m-adjacent if
   - **a.** *q* is in $N_4(p)$, *or*
   - **b.** q *is in* N_D(p*)* and *the set* $N_4(p)$ *n* $N_4(q)$ *has no pixels whose values are from* V

A ***digital path*** (or ***curve)*** from pixel p with coordinates f(x_0,y_0) to pixel ***q*** with coordinates (x_n, y_n) is a sequence of distinct pixels with coordinates

(x_0,y_0), (x_1,y_1),…. (x_n,y_n)

where points (x_i,y_i) and (x_{i-1},y_{i-1}) are adjacent for 1<= **i <= 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.*

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 component, and that component is connected, then S is called a* **connected set.**

Let **R** *represent a subset of pixels in an image. We call* **R** *a* **region** *of the image if* **R** *is a connected set. Two regions,* **R$_t$** *and* **Rj** *are said to be* **adjacent** *if their union forms a connected set. Regions that are not adjacent are said to be* **disjoint.**

The **boundary** *(also called the* **border** *or* **contour)** *of a region* **R** *is the set of pixels in* **R** *that are adjacent to pixels in the complement of* **R.** *Stated 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.*

Distance Measures

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

a. $D(p,q) \geq 0 \quad (D(p,q) = 0 \text{ iff } p = q)$,
b. $D(p,q) = D(q,p)$, and
c. $D(p,s) \leq D(p,q) + D(q,s)$.

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

$$D_e(p,q) = [(x-u)^2 + (y-v)^2]^{\frac{1}{2}} \tag{2-19}$$

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

$$D_4(p,q) = |x-u| + |y-v| \tag{2-20}$$

In this case, pixels having a D_4 distance from (x, y) that is less than or equal to some value d 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:

```
    2
  2 1 2
2 1 0 1 2
  2 1 2
    2
```

The pixels with are the D$_4$-neighbors of (x, y).

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

$$D_8(p,q) = \max(|x-u|, |y-v|) \tag{2-21}$$

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

$$\begin{matrix} 2 & 2 & 2 & 2 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{matrix}$$

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

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

Elementwise Versus Matrix Operations

An *elementwise operation* involving one or more images is carried out on a *pixel-by-pixel* basis. We know that images can be viewed equivalently as matrices.

The elementwise product of two matrices is also called the *Hadamard product* of the matrices.

The *elementwise product* (often denoted using the symbol $\odot$ or $\otimes$) of these two images is

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

The symbol $\ominus$ is often used to denote *elementwise division*.

That is, the elementwise product is obtained by multiplying pairs of *corresponding* pixels. On the other hand, the *matrix product* of the images is formed using the rules of matrix multiplication:

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

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, $\mathcal{H}$, that produces an output image, $g(x, y)$, from a given input image, $f(x, y)$:

$$\mathcal{H}[f(x,y)]=g(x,y)$$

Given two arbitrary constants, ***a*** and ***b*,** and two arbitrary images **$f_1(x, y)$ and $f_2(x, y)$** is said to be a ***linear operator*** if

$$\mathcal{H}[af_1(x,y)+bf_2(x,y)] = a\mathcal{H}[f_1(x,y)]+b\mathcal{H}[f_2(x,y)] = ag_1(x,y)+bg_2(x,y) \qquad (2\text{-}23)$$

This equation indicates that the output of a linear operation applied to the sum of two inputs is the same as performing the operation individually on the inputs and then summing the results. In addition, the output of a linear operation on a constant multiplied by 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*.** By definition, an operator that fails to satisfy Eq. (2-23) is said to be ***nonlinear*.**

(2-23) . 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 $a = 1$ and $b = -1$. To test for linearity, we again start with the left side of **Eq. (2-23)**

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

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. (2-23)** are not equal in this case, so we have proved that the max operator is nonlinear.

Arithmetic Operations

Arithmetic operations between two images *f(x, y)* and *g{x, y)* are denoted as

$$s(x, y) = fix, y) + g(x, y) \qquad (2\text{-}24)$$
$$d(x,y) = f(x,y)-g(x,y)$$
$$Vix.y) = f(x.y) \text{ x } g\{x,y)$$
$$v(x,y) = f(x,y) + (x,y)$$

These are elementwise operations which, as noted earlier in this section, means that they are performed between corresponding pixel pairs in *f* and *g* for $x = 0,1,2,M — 1$ and $y = 0,1,2,N -1$. As usual, *M* and *N* are the row and column sizes of the images. Clearly, *s,d,p,* and *v* are images of size $M X N$ also. Note that image arithmetic in the manner just defined involves images of the same size.

Set Operations

A *set* is a collection of distinct objects. If *a* is an *element* of set *A*, then we write

$$a \in A$$

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

$$A \notin A$$

The set with no elements is called the *null* or *empty set,* and is denoted by $\emptyset$.
A set is denoted by the contents of two braces: {.}. For example, the expression

$$C = \{c \mid c = — d, d \in D]$$

means that *C* is the set of elements, c, such that *c* is formed by multiplying each of the elements of set *D* by -1.

If every element of a set *A* is also an element of a set *B*, then *A* is said to be a *subset* of 6, denoted as

$$A \subseteq B \tag{2-35}$$

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

$$C = A \cup B$$

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

$$D = A \cap B \tag{2-37}$$

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

$$A \cap B = 0 \tag{2-38}$$

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

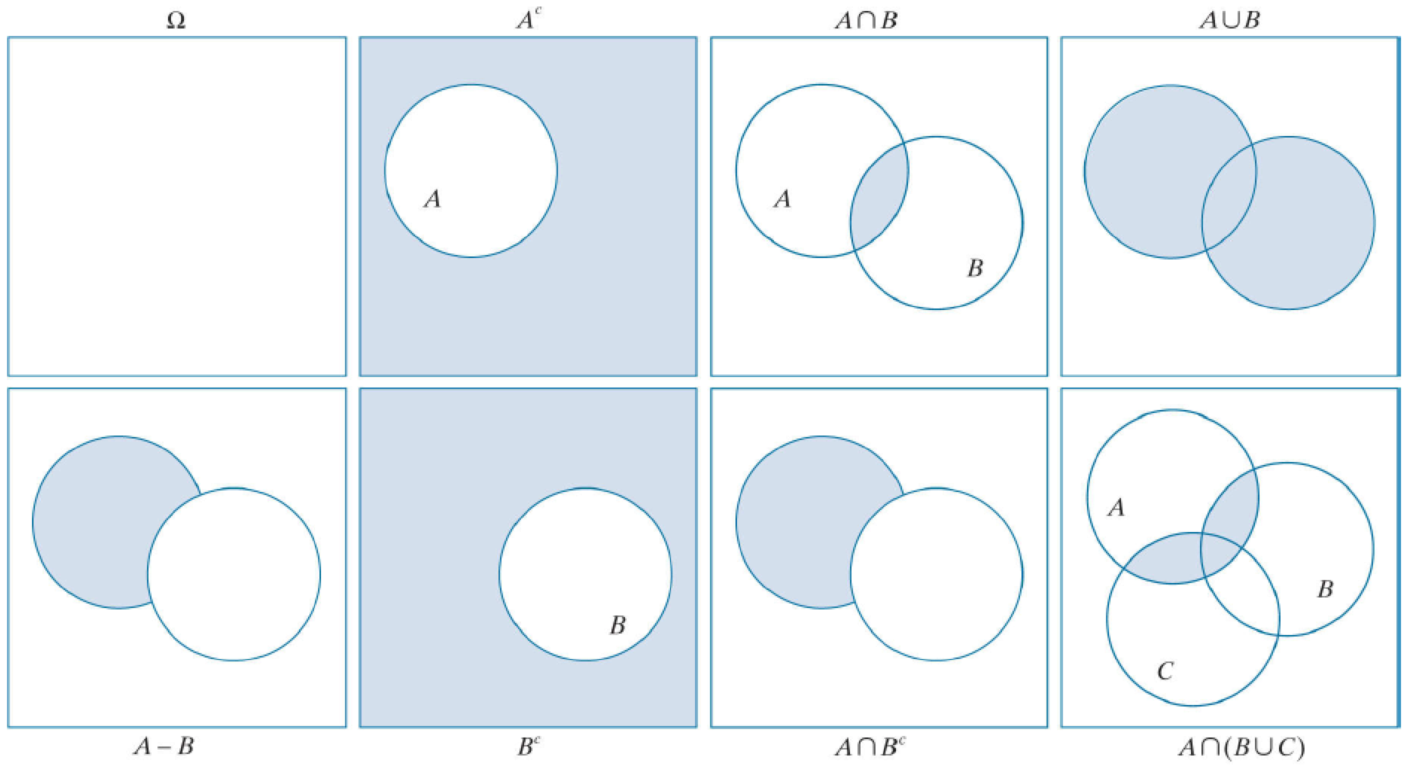
The *complement* of a set *A* is the set of elements that are not in *A.*

$$A^c = \{w \mid w \notin A\} \tag{2-39}$$

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

$$A^c — B = \{ w \mid w \in A, w \notin B] = A \cap B^c \tag{2-40}$$

This is the set of elements that belong to *A*, but not to *B.* We can define A^c in terms of **Ω** and the set difference operation; that is, $A^c = \Omega - A$.



When applying the concepts just discussed to image processing, we let sets represent objects (regions) in a binary image, and the elements of the sets are the (x, y) coordinates of those objects. For example, if we want to know whether two objects, *A* and *B*, of a binary image overlap, all we have to do is compute $A \cap B$. If the result is not the empty set, we know that some of the elements of the two objects overlap.

Logical Operations

Logical operations deal with TRUE (typically denoted by 1) and FALSE (typically denoted by 0) variables and expressions.

We work with set and logical operators on binary images using one of two basic approaches: (1) we can use the *coordinates* of individual regions of foreground pixels in a single image as sets, or (2) we can work with one or more images of the same size and perform logical operations between corresponding pixels in those arrays.

In the first category, a binary image can be viewed as a Venn diagram in which the coordinates of individual regions of 1-valued pixels are treated as sets. The union of these sets with the set composed of 0-valued pixels comprises the set universe, Ω.. For example, given a binary image with two 1-valued regions, R_1 and R_2, we can determine if the regions overlap by performing the set intersection operation $R_1 \cap R_2$. In the second approach, we perform logical operations on the pixels of one binary image, or on the corresponding pixels of two or more binary images of the same size.

Logical operators can be defined in terms of truth tables, as **Table 2.2**

TABLE 2.2
Truth table defining the logical operators AND ($\wedge$), OR ($\vee$), and NOT ($\sim$).

a	*b*	*a* AND *b*	*a* OR *b*	NOT(*a*)
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

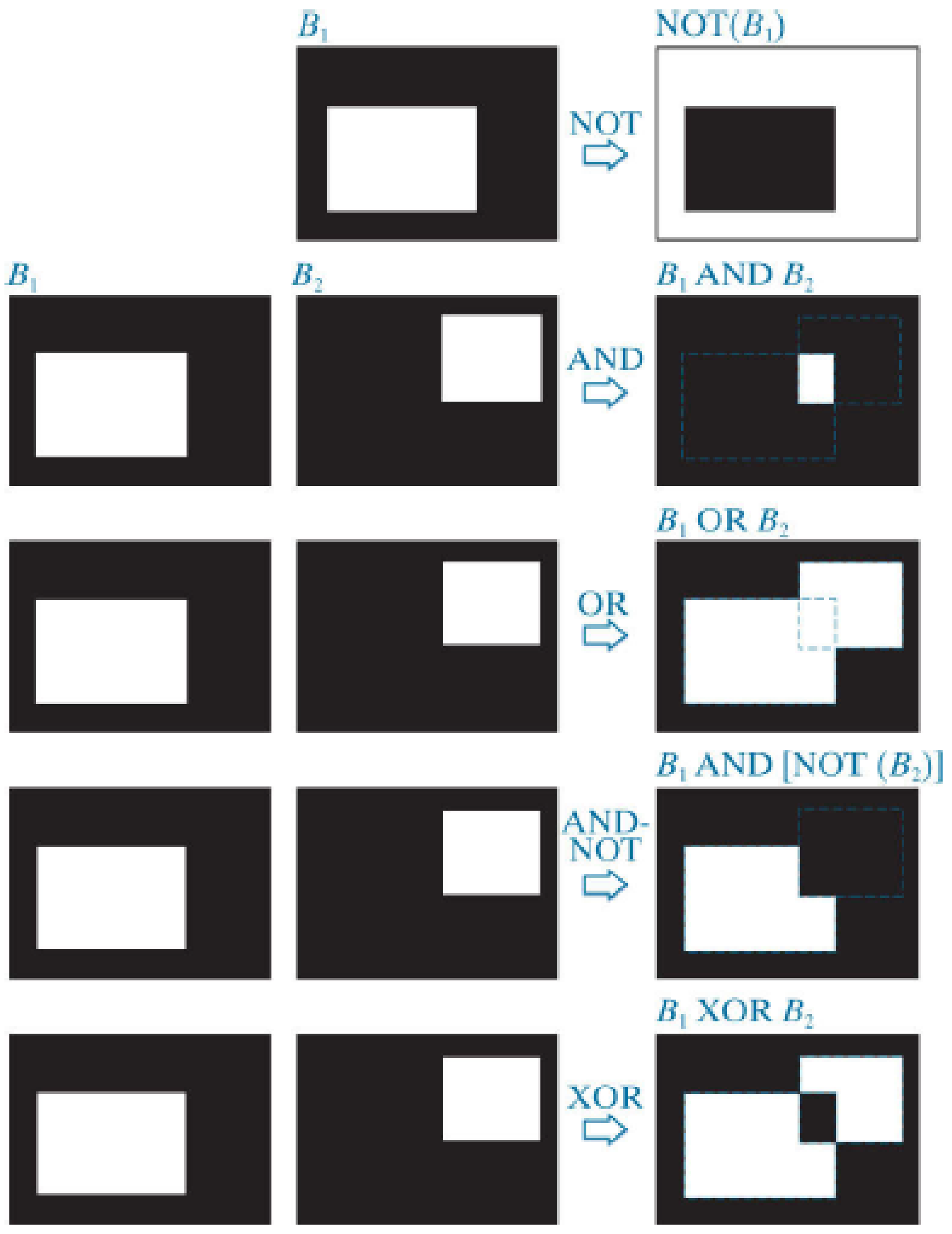


FIGURE 2.37
Illustration of logical operations involving foreground (white) pixels. Black represents binary 0's and white binary 1' are shown for reference only. They are not part of the result.

Fuzzy Sets

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 or not young. We need 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.

Spatial Operations

Spatial operations are performed directly on the pixels of an image. We classify spatial operations into three broad categories:

(1) Single-Pixel Operations,
(2) Neighborhood Operations, And
(3) Geometric Spatial Transformations.

1-Single-Pixel Operations

The simplest operation we perform on a digital image is to alter the intensity of its pixels individually using a transformation function, *T*, of the form:

$$s = T(z) \tag{2-42}$$

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.

2-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 on the neighborhood of pixels in the input image with coordinates in the set 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*** x n centered on (x, ***y).*** The coordinates of pixels in this region are the elements of set S_{xy}**.** Figures 2.39(a) and (b) illustrate the process. We can express this averaging operation as

$$g(x,y) = \frac{1}{mn} \sum_{(r,c) \in S_{xy}} f(r,c) \tag{2-43}$$

where ***r*** and c are the row and column coordinates of the pixels whose coordinates are in 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 then repeating the neighborhood operation at each new location.

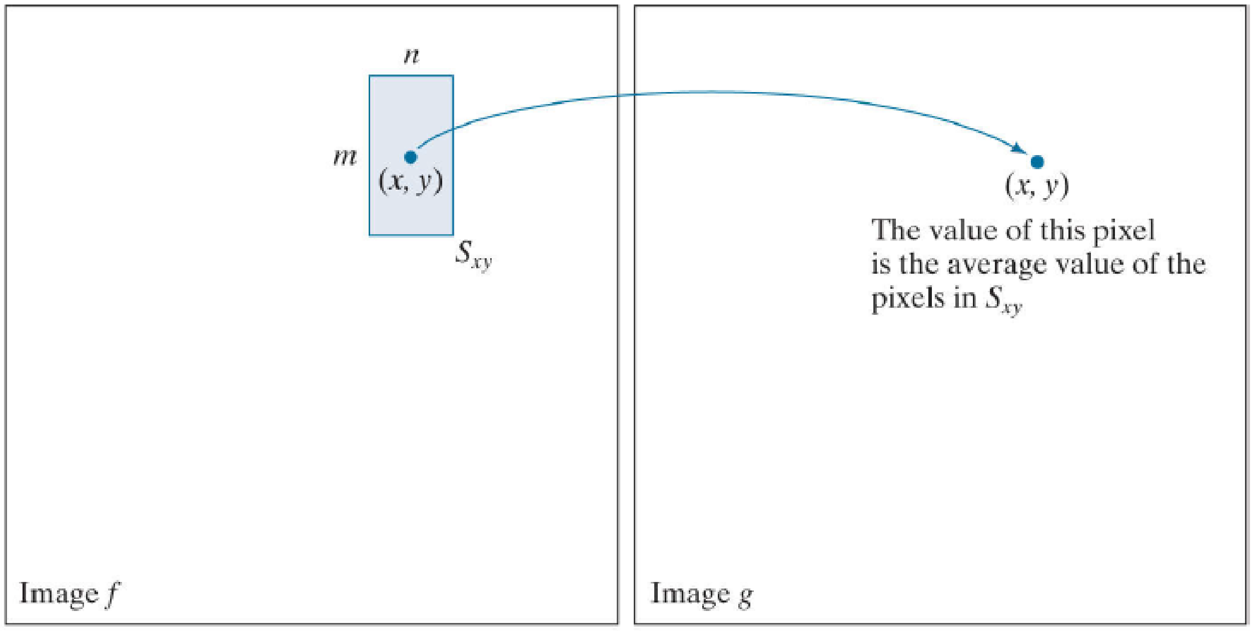

n
m
(x, y)
S_{xy}
(x, y)
The value of this pixel is the average value of the pixels in S_{xy}
Image f
Image g

FIGURE 2.39 **a.** **b.**

Local averaging using neighborhood processing. The procedure is illustrated in (a) and (b) for a rectangular neighborhood.

3-Geometric Transformations

We use geometric transformations modify the spatial arrangement of pixels in an image. These transformations are called ***rubber-sheet transformations*** because they may be viewed as analogous to "printing" an image on a rubber sheet, then stretching or shrinking the sheet according to a predefined set of rules. Geometric transformations of digital images consist of two basic operations:

1. Spatial transformation of coordinates.
2. Intensity interpolation that assigns intensity values to the spatially transformed pixels.

The transformation of coordinates may be expressed as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{T}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} \qquad (2\text{-}44)$$

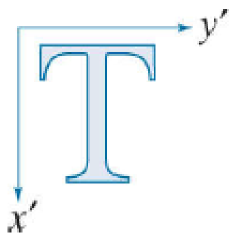
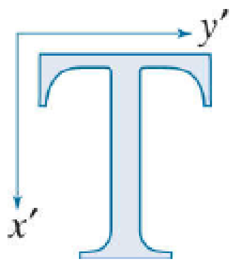
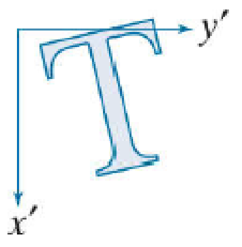
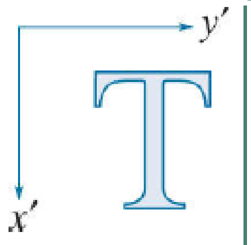
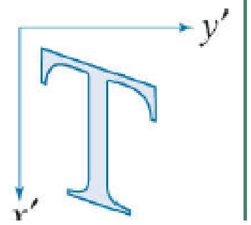
where (x, y) are pixel coordinates in the original image and (***x',y***') are the corresponding pixel coordinates of the transformed image. For example, the transformation (***x',y***') = (x/2,y/2) shrinks the original image to half its size in both spatial directions.

Our interest is in so-called ***affine transformations,*** which include scaling, translation, rotation, and shearing. The key characteristic of an affine transformation in 2-D is that it preserves points, straight lines, and planes Equation (2-44) can be used to express the transformations just mentioned, except translation, which would require that a constant 2-D vector be added to the right side of the equation. However, it is possible to use homogeneous coordinates to express all four affine transformations using a single 3x3 matrix in the following general form:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{A}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

This transformation can *scale*, *rotate*, *translate*, or *sheer* an image, depending on the values chosen for the elements of matrix **A**.

Table 2.3 shows the matrix values used to implement these transformations.

Transformation Name	Affine Matrix, A	Coordinate Equations	Example
Identity	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x$ $y' = y$	
Scaling/Reflection (For reflection, set one scaling factor to −1 and the other to 0)	$\begin{bmatrix} c_x & 0 & 0 \\ 0 & c_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = c_x x$ $y' = c_y y$	
Rotation (about the origin)	$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x\cos\theta - y\sin\theta$ $y' = x\sin\theta + y\cos\theta$	
Translation	$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x + t_x$ $y' = y + t_y$	
Shear (vertical)	$\begin{bmatrix} 1 & s_v & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x + s_v y$ $y' = y$	

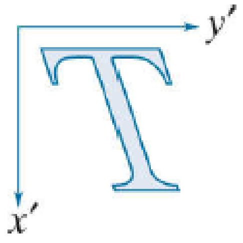
Shear (horizontal)	$\begin{bmatrix} 1 & 0 & 0 \\ s_h & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$x' = x$ $y' = s_h x + y$	

Image Registration

Image registration is an important application of digital image processing used to align two or more images of the same scene. In image registration, we have available an ***input*** image and a ***reference*** image. The objective is to transform the input image geometrically to produce an output image that is aligned (registered) with the reference image. Transformation functions are known, the geometric transformation needed to produce the output, but registered image generally is not known, and must be estimated.

Examples of image registration include aligning 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 instruments, such as satellite images of a given location taken several days, months, or even years apart.

One of the principal approaches for solving the problem just discussed is to use *tie points* (also called *control points).* These are corresponding points whose locations are known precisely in the input and reference images. Approaches for selecting tie points range from selecting them interactively to using algorithms that detect these points automatically. Some 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* or *fiducial marks*) directly on all images captured by the system. These known points can then 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

$$x = c_1 v + c_2 w + c_3 vw + c_4 \qquad (2\text{-}46)$$

and

$$y = c_5 v + c_6 w + c_7 vw + c_8 \qquad (2\text{-}47)$$

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. (2-46) and (2-47) and use them to solve for the eight unknown coefficients, c_1 through c_8.

Once we have the coefficients, Eqs. (2-46) and (2-47) become our vehicle for transforming all the pixels in the input image. The result is the desired registered image. After the coefficients have been computed, we let (v, ***w)*** denote the coordinates of each pixel in the input image, and (x, y) become the corresponding coordinates of the output image. The same set of coefficients, c, through c_8, are used in computing all coordinates (x, y); we just step through all (v, w) in the input image to generate the corresponding (x, y) in the output, registered image. If the tie points were selected correctly, this new image should be registered with the reference image, within the accuracy of the bilinear approximation model.

Vector and Matrix Operations

Multispectral image processing is a typical area in which vector and matrix operations are used routinely. We know that color images are formed in RGB color space by using red, green, and blue component images, as Fig. 2.43 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

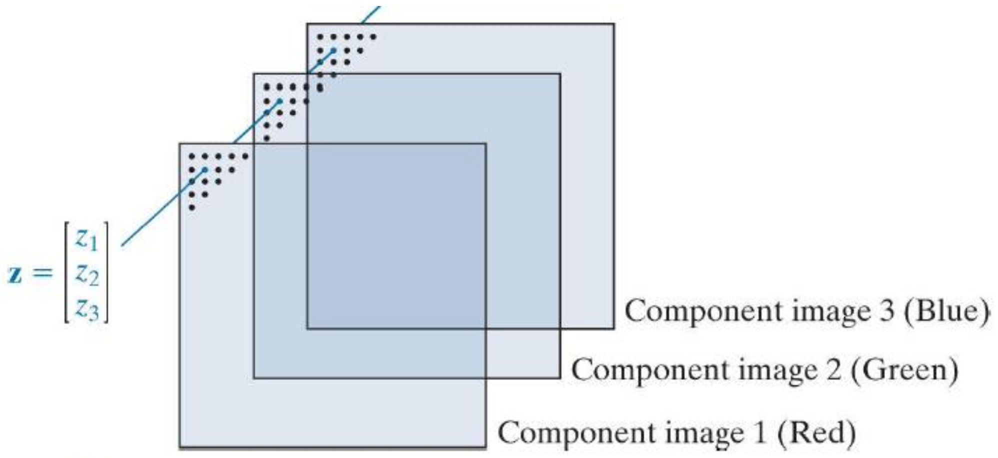

$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}$
Component image 3 (Blue)
Component image 2 (Green)
Component image 1 (Red)

FIGURE 2.43
Forming a vector from corresponding pixel values in three RGB component images.

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \tag{2-48}$$

where z_1 is the intensity of the pixel in the red image, and z_2 and z_3 are the corresponding pixel intensities in the green and blue images, respectively. Thus, an RGB color image of size ***M*** x ***N*** can be represented by three component images of this size, or by a total of ***MN*** vectors of size 3 x 1.

The *inner product* (also called the *dot product*) of two *n*-dimensional column vectors **a** and **b** is defined as

$$\begin{aligned} \mathbf{a} \cdot \mathbf{b} &\triangleq \mathbf{a}^T\mathbf{b} \\ &= a_1b_1 + a_2b_2 + \cdots + a_nb_n \\ &= \sum_{i=1}^{n} a_ib_i \end{aligned} \tag{2-50}$$

where *T* indicates the transpose. The *Euclidean vector norm*, denoted by $\|\mathbf{z}\|$, is defined as the square root of the inner product:

$$\|\mathbf{z}\| = (\mathbf{z}^T\mathbf{z})^{\frac{1}{2}} \tag{2-51}$$

We recognize this expression as the length of vector **z**.

Image Transforms

All the Image processing approaches discussed thus far operate directly on the pixels of an 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

$$T(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) r(x,y,u,v) \tag{2-55}$$

where $f(x, y)$ is an input image, $r(x, y, u, v)$ is called a *forward transformation kernel*, and **Eq. (2-55)** 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 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 $T(u, v)$:

$$f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} T(u,v) s(x,y,u,v) \tag{2-56}$$

for $x = 0, 1, 2, \ldots, M-1$ and $y = 0, 1, 2, \ldots, N-1$, where $s(x, y, u, v)$ is called an *inverse transformation kernel*. Together, **Eqs. (2-55)** and **(2-56)** are called a *transform pair*.

Figure 2.44 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

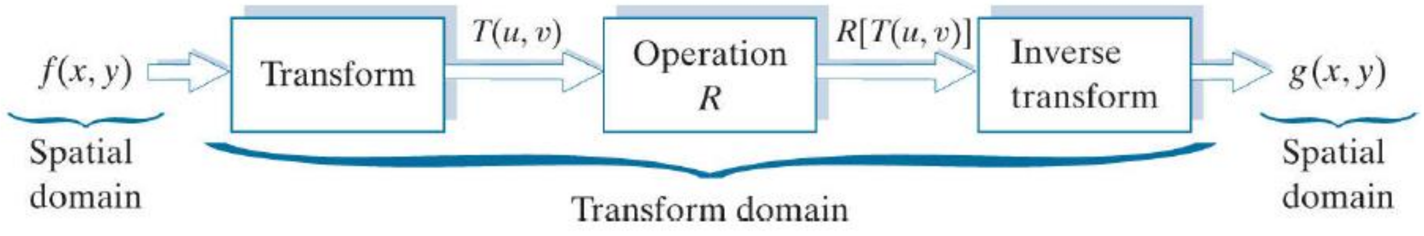


FIGURE 2.44

General approach for working in the linear transform domain.

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

$$r(x, y, u, v) = r_1(x, u) r_2(y, v) \tag{2-57}$$

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

$$r(x, y, u, v) = r_1(x, u') r_1(y, v) \tag{2-58}$$

Identical comments apply to the inverse kernel.

The nature of a transform is determined by its kernel. A transform of particular importance in digital image processing is the ***Fourier transform,*** which has the following forward and inverse kernels:

$$r(x,y,u,v) = e^{-j2\pi(ux/M+vy/N)} \quad (2\text{-}59)$$

and

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

respectively, where $j = \sqrt{-1}$, so these kernels are complex functions. Substituting the preceding kernels into the general transform formulations in Eqs. (2-55) and (2-56) gives us the *discrete Fourier transform pair*.

Probability and Random Variables

Probability is a branch of mathematics that deals with uncertainty.

Sample Spaces, Events, and Probability

A ***random experiment*** is a process whose outcome cannot be predicted with certainty, but whose set of ***all possible*** outcomes can be specified. As noted earlier when discussing sets, the set of all possible outcomes of an experiment is called the ***sample space*** of the experiment, and is denoted by ***Ω.*** A familiar random experiment consists of tossing a single die and observing the numerical value of the face that lands facing up. The sample space of this experiment is the set ***Ω*** = {1,***2,3,4,***5,6}.

An ***event*** is a ***subset*** of the sample space. In the single-die experiment, the event ***A*** = {1,3,5} is the subset of n that correspond to the odd faces of the die. We say that an event ***occurs*** if the outcome of an experiment is ***any*** of the elements of the event set.

To make the notion of a random experiment useful, we need a measure (a probability) that quantifies how likely it is than an event will occur. A ***probability, P,*** is a ***function*** that satisfies the following properties:

To make the notion of a random experiment useful, we need a measure (a probability) that quantifies how likely it is than an event will occur. A ***probability, P,*** is a ***function*** that satisfies the following properties:

1. 0 <= ***P(A)*** <= 1 for every event ***A.***
2. P(Ω) = 1.
3. If $A_1, A_2, \ldots\ldots A_n$ are disjoint events, then

$$P(A_1 \cup A_2 \cup \ldots\ldots \cup A_n) = P(A_1) + P(A_2) + \ldots + P(A_n). \quad (2\text{-}67)$$

These three properties are called the ***axioms of probability*** Axiom 1 says that the probability must be a number in the range [0, 1], with 0 indicating that ***A*** never occurs, and 1 indicating that ***A*** always occurs. Because n is the set containing all possible outcomes, the second axiom indicates that some event from Ω always occurs when the experiment is performed. Axiom 3 states that the probability of the union of a sequence of ***disjoint*** events is equal to the sum of the probabilities of the individual events.

The Sum (Addition) Rule of Probability

Axiom 3 is a special case of the *sum* (or *addition) rule* of probability, which states that the probability of the union of *n* events is equal to the sum of the probabilities of these events taken one at time, minus the sum of the probabilities of the events taken two at a time, plus the sum of the probabilities of the events taken three at a time, and so on, up to the sum of the probabilities of all the *n* events (Ross [2014]).

For two events, the sum rule is

$$P(A \cup P) = P(A) + P(P) - P(A \cap B) \quad (2\text{-}68)$$

For three events, this expression becomes

$$P(A \cup B \cup C) = P(A) + P(B) + P(C) - p(A \cap B) - p(A \cap C) - P(B \cap C) + P(A \cap B \cap C) \quad (2\text{-}69)$$

When the events are disjoint, all terms except the individual probabilities become zero, thus reducing the expression to the one given in Axiom 3. The rightmost term in Eq. (2-69) is a result of applying Eq. (2-68) to combined events.

Conditional Probability

The probability of event *A*, given that event *B* has occurred, is defined as

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)} \quad (2\text{-}70)$$

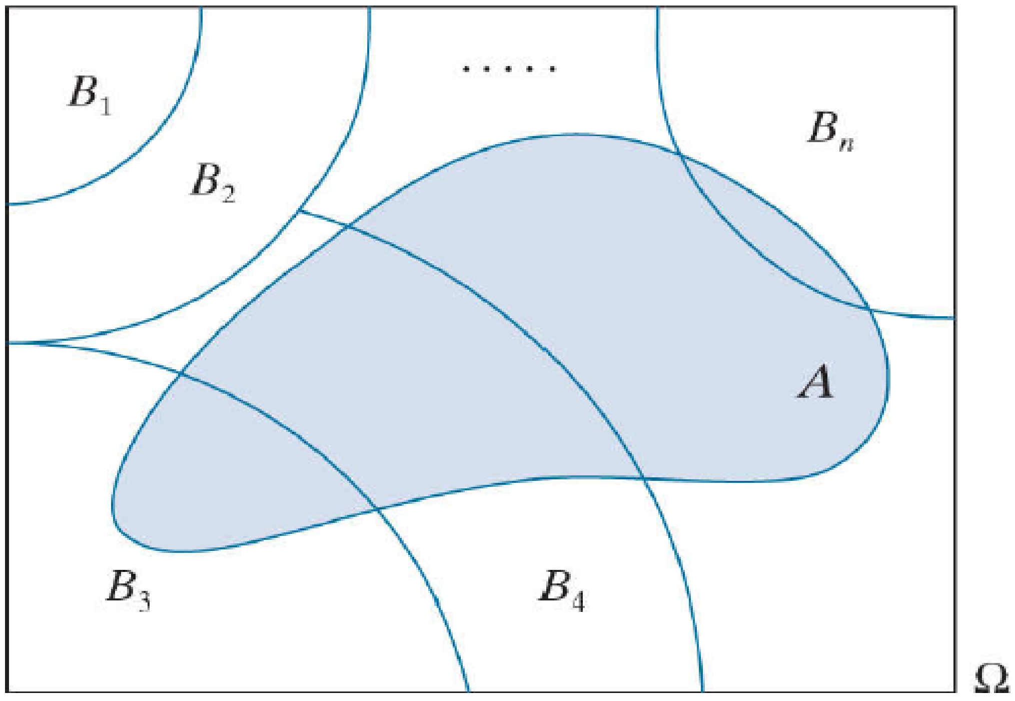
where $P(A \mid B)$ is called the *conditional probability*; it reads "the probability of *A* given *B*." As noted above, $P(A \cap B)$. is the joint probability of *A* and *B*.

The Law of Total Probability

With reference to Fig. 2.47 , suppose that events $B_1, B_2, \ldots, B_n$ form a *partition* of the sample space. That is, the events are disjoint sets, and their union contains all the elements of Ω. From the preceding discussion, it follows that *P*(*A*), the probability of event *A*, is the sum of the contributions made by the intersection of *A* with each of the elements of the partition:

$$P(A) = \sum_{i=1}^{n} P(A \cap B_i) \quad (2\text{-}78)$$

This result is called the *partition theorem*, which is a mathematical way of saying that the whole is equal to the sum of its parts (*A* does not have to intersect every *B* of the partition, as illustrated in Fig. 2.47).



Using Eq. (2-70) we can write Eq. (2-78) as

$$P(A) = \sum_{i=1}^{n} P(A \mid B_i)P(B_i) \tag{2-79}$$

This expression is called the *law of total probability*.

Bayes' Rule

Bayes' theorem is stated mathematically as the following equation:[3]

$$\boxed{P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}}$$

where A and B are events and $P(B) \neq 0$.

- $P(A \mid B)$ is a conditional probability: the likelihood of event A occurring given that B is true.
- $P(B \mid A)$ is also a conditional probability: the likelihood of event B occurring given that A is true.
- $P(A)$ and $P(B)$ are the probabilities of observing A and B respectively; they are known as the marginal probability.

Random Variables

However, suppose that instead of being interested in just the elements of the sample space, n, we are interested in a ***function*** of those elements. For example, in the experiment of throwing a single die, we might be interested in the sum of the up faces in the first ***n*** throws. The outcome of the experiment is now a random number (called a ***random variable)*** that can take on values much different from just the six faces of a die. When dealing with such outcomes, we can ask questions such as: the average value of ***n*** throws; the probability that the sum of the first ***n*** throws exceeds a given quantity; and so on. The list is endless.

A ***random variable*** is a ***function*** that maps outcomes of a random experiment to real numbers. The set of all possible values of a random variable is called its ***range, R.*** It is customary to denote random variables by upper case letters, such as Z, and to use corresponding lower case letters, z, to denote specific values of Z from ***R.*** Recalling that n is the set of all outcomes of a random experiment, we can write the definition just given as

$$Z : \Omega \rightarrow \mathbb{R} \tag{2-82}$$

where $\mathbb{R}$ is the set of all real numbers (i.e., the real line). Figure 2.48 illustrates these concepts using the sum of the throw of two dice as an example. In this case Z is the summation operator, and z is an integer value from the set of possible values $R = \{2, 3, 4, \ldots, 12\}$. Note that multiple outcomes from Ω are mapped to single values in the range of Z so, in this case, Ω has 36 elements while R only has 11.

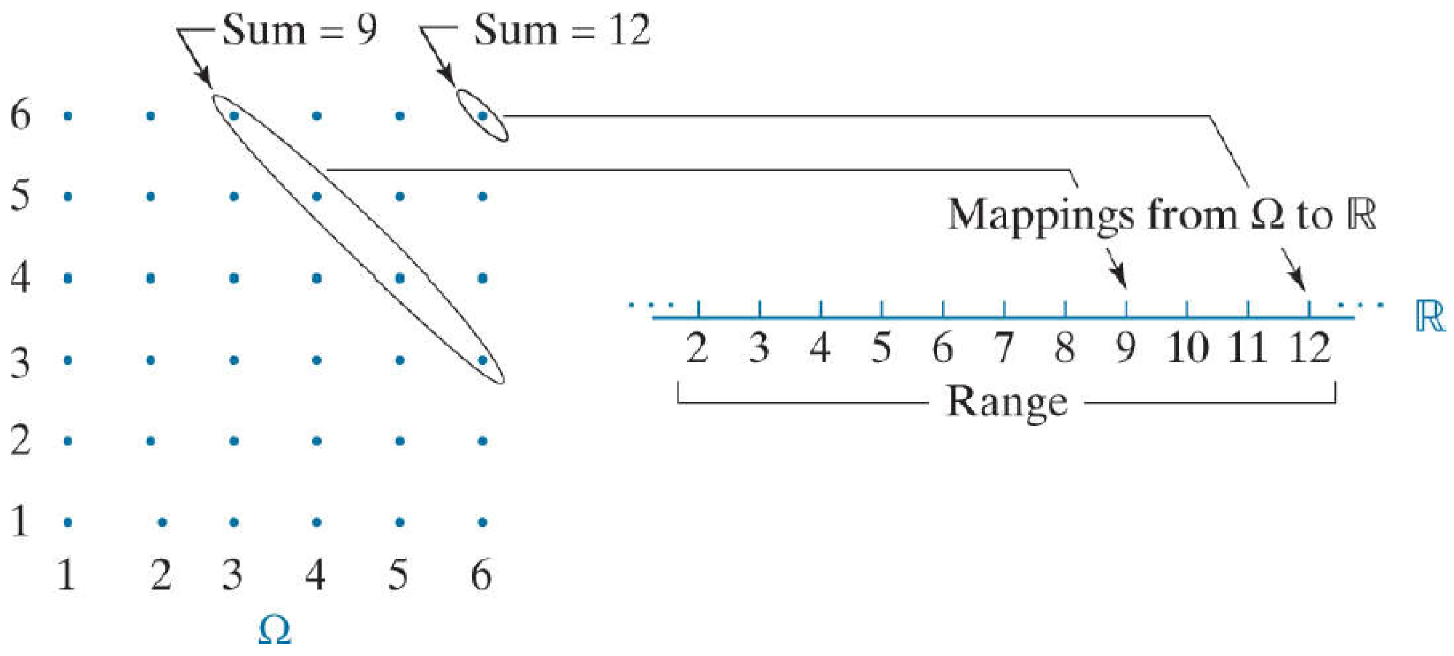


FIGURE 2.48

Probability Functions for Discrete Random Variables

One way to specify a probability function for a random experiment with discrete outcomes is to list the probability value of each possible outcome. However, as mentioned above, the number of possible discrete outcomes of an experiment can be countably infinite. In such situations, we are interested in probability functions that have a closed functional form and thus describe entire ***families*** of random variables.

Let Z denote a random variable with values z and range ***R.*** A ***probability function, p(z),*** lists (literally or functionally) the probabilities of all possible values of z, and satisfies the following conditions:

$$1.\ p(z) \geq 0 \text{ for all } z \in R, \text{ and} \tag{2-83}$$

$$2.\ \sum_{z \in R} p(z) = 1$$

It is understood that p(z) means ***P(Z = z)***; that is, this is the probability that random variable Z has the specific value z from ***R.*** Often, you will see p(z) written as ***$p_z(z)$*** to emphasize that we are dealing with probabilities of values of random variable Z. We use the shorter notation, except when working with more than one random variable simultaneously, in which case we use subscripts for clarity, such as $p_z(z)$ and ***R_z,*** on both the random variable and its range.

For discrete random variables, p(z) is often referred to as a ***probability mass function*** (PMF). For continuous random variables, p(z) is called a ***probability density function*** (PDF). When working with discrete quantities that are approximations of continuous quantities (image intensity is an example), we typically use the term PDF to emphasize the fact that the underlying random variables are continuous.

Some Important Probability Mass Functions

The simplest PMF is the *discrete uniform* PMF, which has the functional form

$$p(z) = \begin{cases} 1/n & \text{if } z \in \{z_1, z_2, \ldots, z_n\} \\ 0 & \text{otherwise} \end{cases} \tag{2-85}$$

where it is assumed that the values of *z* are distributed uniformly over the interval from z_1 to z_n. For instance, a random variable that assigns the value 1/6 to the numbers {1, 2, 3, 4, 5, 6} and zero to all other numbers has a discrete uniform PMF. We used this PMF earlier to model the tossing of a single fair die.

Another important PMF is the *Bernoulli* PMF, which has the form

$$p(z) = \begin{cases} Q & \text{if } z = 1 \\ 1 - Q & \text{if } z = 0 \\ 0 & \text{if } z \neq 1 \text{ and } z \neq 0 \end{cases} \tag{2-86}$$

This PMF can be used to model the flipping of a fair coin by selecting $Q = 0.5$.

The probability of getting exactly *z* successes in *n* trials of an experiment with two possible outcomes can be modeled by the *Binomial* PMF, which has the form

$$p(z) = C(n,z)S^z(1-S)^{n-z} \tag{2-87}$$

where *S* is the probability of success in a single trial, and *C*(*n*, *z*) is the binomial coefficient (thus the name of this PMF):

$$C(n,z) = \frac{n!}{z!(n-z)!} \tag{2-88}$$

The binomial coefficient is the number of combinations of *n* things taken *z* at a time, where $n! = (1)(2)(3)\cdots(n)$. This PMF is used, for example, to model the probability of obtaining exactly *z* heads (or tails) in *n* flips of a fair coin. For instance, the probability of obtaining 7 heads out 10 tosses is

$$p(7) = \frac{10!}{7!(10-7)!}(0.5)^7(1-0.5)^{10-7} = 0.1172$$

where $S = 0.5$ is the probability of a head coming up in a single trial.

Estimating Discrete Probability Functions from Sample Data

Given a set of ***K*** discrete observations of a random variable Z with range ***R,*** an estimate of its PMF is obtained by dividing the number of occurrences of each value by the total number of observations (this is called the ***relative frequency approach*** for estimating probabilities):

$$p(z) = \frac{h(z)}{K} \qquad z \in R \tag{2-89}$$

where ***h(z)*** is the number of times that value z occurs in the ***K*** observations. We do this for each value z in ***R,*** keeping in mind that the number of observations of some occurrences of a particular value may be zero. Thus, Eq. (2-89) is an estimate of the ***probability of occurrence*** of any value z from ***R.***

Listing the probabilities of all possible values of a discrete random variable is of interest in digital image processing because the number of possible intensity levels in an image is usually small (e.g., 256 possible discrete intensity levels for 8-bit images). Intensity can be viewed as a random variable, and the probability of each intensity value occurring in a given image can be estimated using Eq. (2-89).

Let z denote the values of a discrete random variable representing pixel intensities in a digital image of size ***M x N*** pixels. We know that values of z are ***integers*** in the interval [0, L -1]; that is, ***R*** = {0,1,2, ***...,L* -1}.** From this, we can estimate the probability mass function using Eq. (2-89)

$$p(z) = \frac{h(z)}{MN} \qquad z \in \{0, 1, 2, \dots, L-1\} \tag{2-90}$$

where the product ***MN*** is the number of pixels in the image. Note that ***MN*** may be very large, but ***R*** only has ***L*** elements.

Function ***h(z)*** in both Eqs. (2-89) and (2-90) often is called an ***unnormalized histogram*** and ***p(z)*** is called a ***normalized histogram.*** Clearly, p(z) is a PMF that satisfies Eq. (2-83) . When dealing with images, we typically use normalized histograms and refer to them simply as ***histograms, intensity histograms,*** or ***image histograms.***

Expected Value and Moments of Discrete Random Variables

Let $g(z)$ denote a function defined over the values, z, of discrete random variable Z. As before, $z \in R$, where R is the range of Z. The *expected value* of $g(z)$ is defined as

$$E[g(z)] = \sum_{z \in R} g(z)p(z) \tag{2-91}$$

The simplest form of this equation is when $g(z) = z$, which yields the *mean* (*average*) of the values of random variable Z:

$$\bar{z} = E[z] = \sum_{z \in R} zp(z) \tag{2-92}$$

We also use the symbol μ in a different context to denote the mean.

As an example of this equation, consider the histogram in Fig. 2.49(b), which lists all the values of $p(z)$ for the image in Fig. 2.49(a). The average value of z is

$$\bar{z} = E[z] = \sum_{z \in \{0,1,2,3\}} zp(z) = (0)p(0) + (1)p(1) + (2)p(2) + 3p(3)$$
$$= (0)(0.125) + (1)(0.250) + (2)(0.500) + (3)(0.125) = 1.625$$

Because z represents intensities in this case, $E[z]$ is the average intensity of the image. Note that the values of $p(z)$ act as *weighting factors* on the values of z. In this case, the relatively high value of $p(2)$ caused the average value to be higher than the midpoint of the intensity interval [0, 3]. If $p(z)$ had been uniform instead, all weighting factors would have been equal (1/4), and the average intensity would have been 1.5, exactly midway between the lowest and highest intensities in the image.

Continuous Random Variables

As noted earlier, a random variable Z is said to be *continuous* if its values, z, are continuous. A continuous random variable is characterized by a *probability density function* (PDF), $p(z)$, such that

$$\text{1. } p(z) \geq 0 \text{ for all } z. \tag{2-97}$$
$$\text{2. } \int_{-\infty}^{\infty} p(z)dz = 1.$$
$$\text{3. } P(a \leq Z \leq b) = \int_{a}^{b} p(z)dz.$$

Unlike discrete random variables, the probability that a continuous random variable has a specific value is 0, as you can see by letting $b = a$ in the preceding definition:

$$P(Z = a) = \int_{b}^{a} p(z)dz = 0$$

The *cumulative distribution function* (CDF) for a continuous random variable Z is defined as

$$F(z) = \int_{-\infty}^{z} p(v)dv \tag{2-98}$$

where v is a dummy variable of integration.

The Uniform and Gaussian Probability Density Functions

The *uniform* PDF is defined as

$$p(z) = \begin{cases} \frac{1}{b-a} & \text{for } a \leq z \leq b \\ 0 & \text{otherwi se} \end{cases} \tag{2-99}$$

The corresponding CDF is given by

$$F(z) = \begin{cases} 0 & z < a \\ \int_a^z \frac{1}{b-a} dv = \frac{z-a}{b-a} & a \leq z \leq b \\ 1 & z > b \end{cases} \tag{2-100}$$

Figure 2.52 shows plots of the uniform PDF and CDF.

a
b

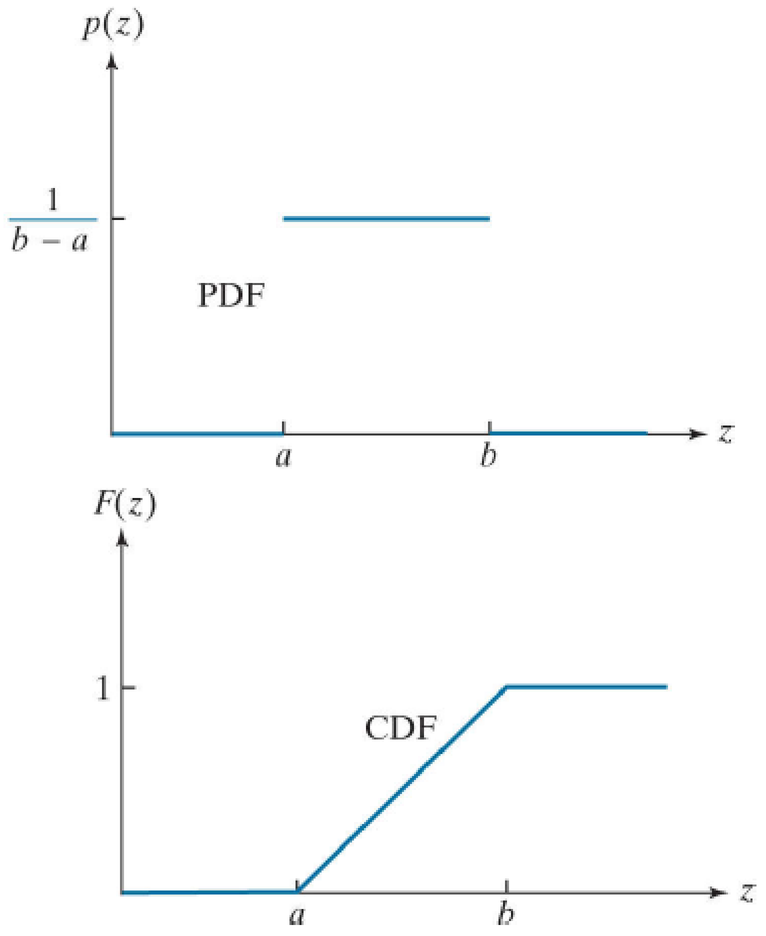


FIGURE 2.52
The uniform PDF and CDF.

The mean and variance of the uniform density are

$$\bar{z} = E[z] = \frac{a+b}{2} \tag{2-101}$$

$$\sigma^2 = E[z^2] - \overline{z}^2 = \frac{1}{12}(b-a)^2 \qquad (2\text{-}102)$$

Expected Values and Moments of Continuous Random Variables

The expressions for expected values and moments of continuous random variables have the same form as their discrete counterparts in Eqs. (2-91) through(2-96) , but with the summations replaced by integrals. Thus, the *expected value* of a function *g*(*z*) defined over the values, *z*, of continuous random variable *Z* is defined by the expression

$$E[g(z)] = \int_{-\infty}^{\infty} g(z)p(z)dz \qquad (2\text{-}105)$$

Similarly, the *mean* and *variance* are defined as:

$$\overline{z} = E[z] = \int_{-\infty}^{\infty} zp(z)dz \qquad (2\text{-}106)$$

and

$$\sigma^2 = E[(z-\overline{z})^2] = \int_{-\infty}^{\infty} (z-\overline{z})^2 p(z)dz \qquad (2\text{-}107)$$

As before, the variance can be expressed in the following equivalent forms:

$$\begin{aligned}\sigma^2 &= E[(z-\overline{z})^2] \\ &= E[z^2] - \overline{z}^2 \\ &= \int_{-\infty}^{\infty} z^2 p(z)dz - \overline{z}^2\end{aligned} \qquad (2\text{-}108)$$

Finally, the *central moment of order n* for continuous random variables is defined as:

$$\mu_n = E[(z-\overline{z})^n] = \int_{-\infty}^{\infty} (z-\overline{z})^n p(z)dz \qquad (2\text{-}109)$$

Intensity Transformations and Spatial Filtering

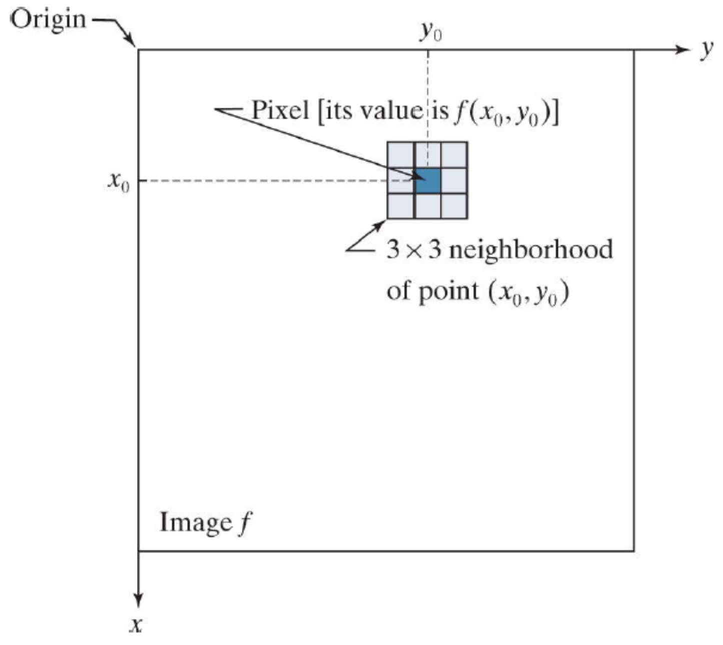
3.1-The Basics of Intensity Transformations and Spatial Filtering

The spatial domain processes we discuss in this chapter are based on the expression

$$g(x,y) = T[f(x,y)] \qquad \textbf{(3-1)}$$

where **f(x, y)** *is an 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 be applied to the pixels of a single image or to the pixels of a set of images, such as performing the elementwise sum of a sequence of images for noise reduction.*

Figure 3.1 shows the basic implementation of Eq. (3-1) on a single image. The point (x_0, y_0) shown is an arbitrary location in the image, and the small region shown is a ***neighborhood*** of (x_0,y_0).



Origin
y_0
y
Pixel [its value is $f(x_0, y_0)$]
x_0
3×3 neighborhood of point (x_0, y_0)
Image f
x

FIGURE 3.1

The process that Fig. 3.1 illustrates consists of moving the center of the neighborhood from pixel to pixel, and applying the operator T to the pixels in the neighborhood to yield an output value at that location. Thus, for any specific location (x_0,y_0), 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_0, y_0) in f. For example, suppose that the neighborhood is a square of size 3x3 and that operator ***T*** is defined as "compute the average intensity of the pixels in the neighborhood." Consider an arbitrary location in an image, say (100,150). The result at that location in the output image, g(100,150), is the sum of f(100,150) and its 8-neighbors, divided by 9. The center of the neighborhood is then moved to the next adjacent 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 (vertical) scan, one row (column) at a time.

3.2-Some Basic Intensity Transformation Functions

Intensity transformations are among the simplest of all image processing techniques. we denote the values of pixels, before and after processing, by *r* and *s*, respectively. Because we deal with digital quantities, values of an intensity transformation function typically are stored in a table, and the mappings from *r* to *s* are implemented via table lookups. For an 8-bit image, 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 in image processing: 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 the input and output intensities are identical.

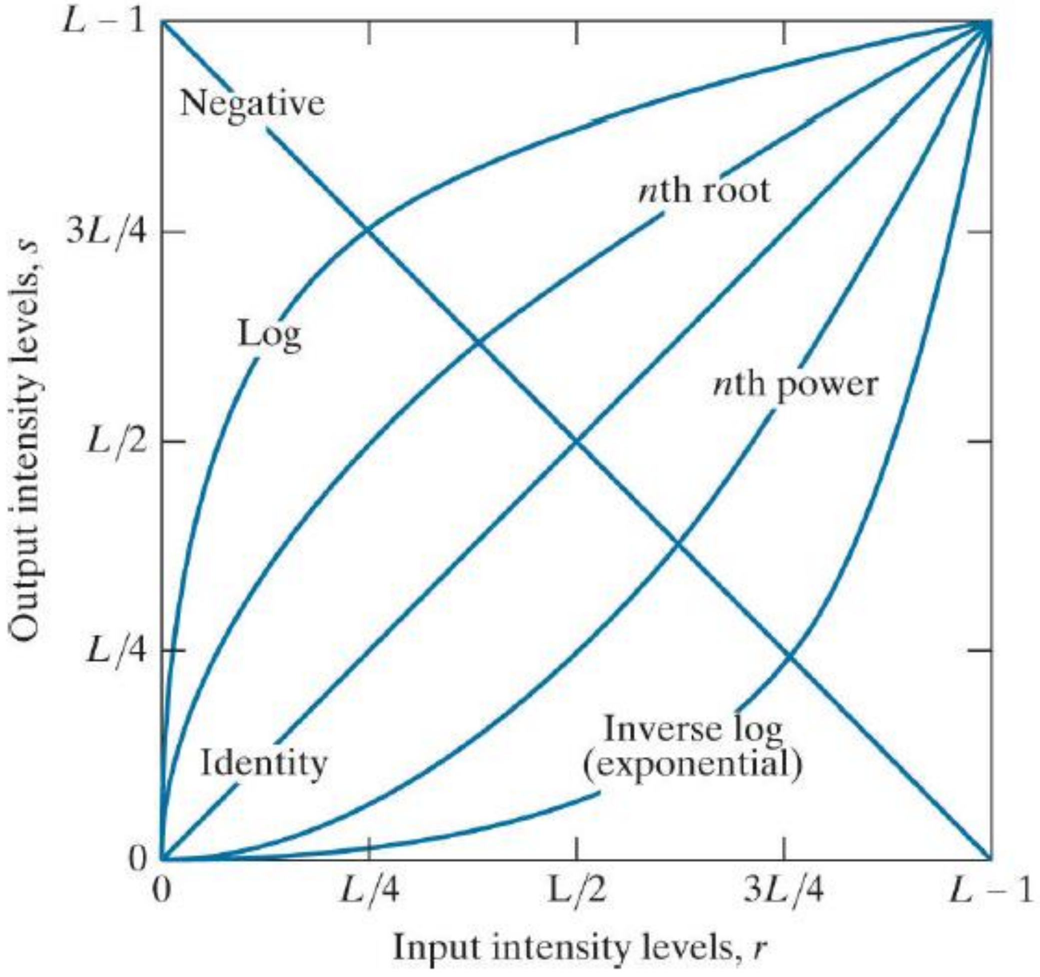


FIGURE 3.3

Image Negatives

The negative of an image with intensity levels in the range [0, *L* - 1] is obtained by using the negative transformation function shown in Fig. 3.3, which has the form:

$$s = L - 1 - r \qquad (3\text{-}3)$$

Reversing the intensity levels of a digital image in this manner produces the equivalent of a photographic negative. This type of processing is used, for example, in enhancing white or gray detail embedded in dark regions of an image, especially when the black areas are dominant in size.

Log Transformations

The general form of the log transformation in **Fig. 3.3** is where *c* is a constant and it is assumed that 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. For example, note how input levels in the range [0,*L*/4] map to output levels to the range [0, 3*L*/4]. Conversely, higher values of input levels are mapped to a narrower range in the output. 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 (exponential) transformation.

Any curve having the general shape of the log function shown in Fig. 3.3 would accomplish this spreading/compressing of intensity levels in an image. The log function has the important characteristic that it compresses the dynamic range of pixel values. It is not unusual to encounter spectrum values that range from 0 to 10^6 or higher. Processing numbers such as these presents no problems for a computer, but image displays cannot reproduce faithfully such a wide range of values. The net effect is that intensity detail can be lost in the display of a typical Fourier spectrum.

Power-Law (Gamma) Transformations

Power-law transformations have the form

$$s = cr^y \qquad (3\text{-}5)$$

where c and *y* are positive constants. Sometimes Eq. (3-5) is written as $s = c(r + \mathcal{E})^y$ to account for offsets (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. (3-5). Figure 3.6 shows plots of s as a function of *r* for various values of *y*. As with log transformations, power-law curves with fractional values of *y* 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.

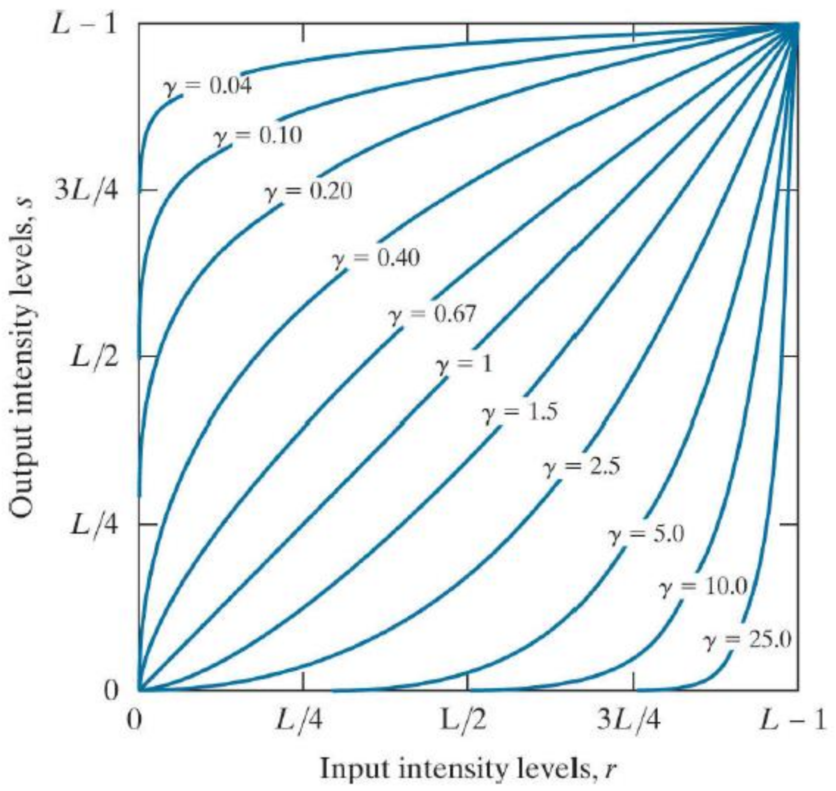


FIGURE 3.6

Note also in Fig. 3.6 that a family of transformations can be obtained simply by varying ***y*.** Curves generated with values of ***y*** > 1 have exactly the opposite effect as those generated with values of ***y*** < 1. When ***c*** = ***y*** = 1 Eq. (3-5) reduces to the identity transformation.

The response of many devices used for image capture, printing, and display obey a power law. By convention, the exponent in a power law equation is referred to as *gamma* [hence our use of this symbol in **Eq. (3-5)**]. The process used to correct these power-law response phenomena is called *gamma correction* or *gamma encoding*. 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.

Piecewise Linear Transformation Functions

Contrast Stretching

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*** expands the range of intensity levels in an image so that it spans the ideal full intensity range of the recording medium or display device.

a b
c d

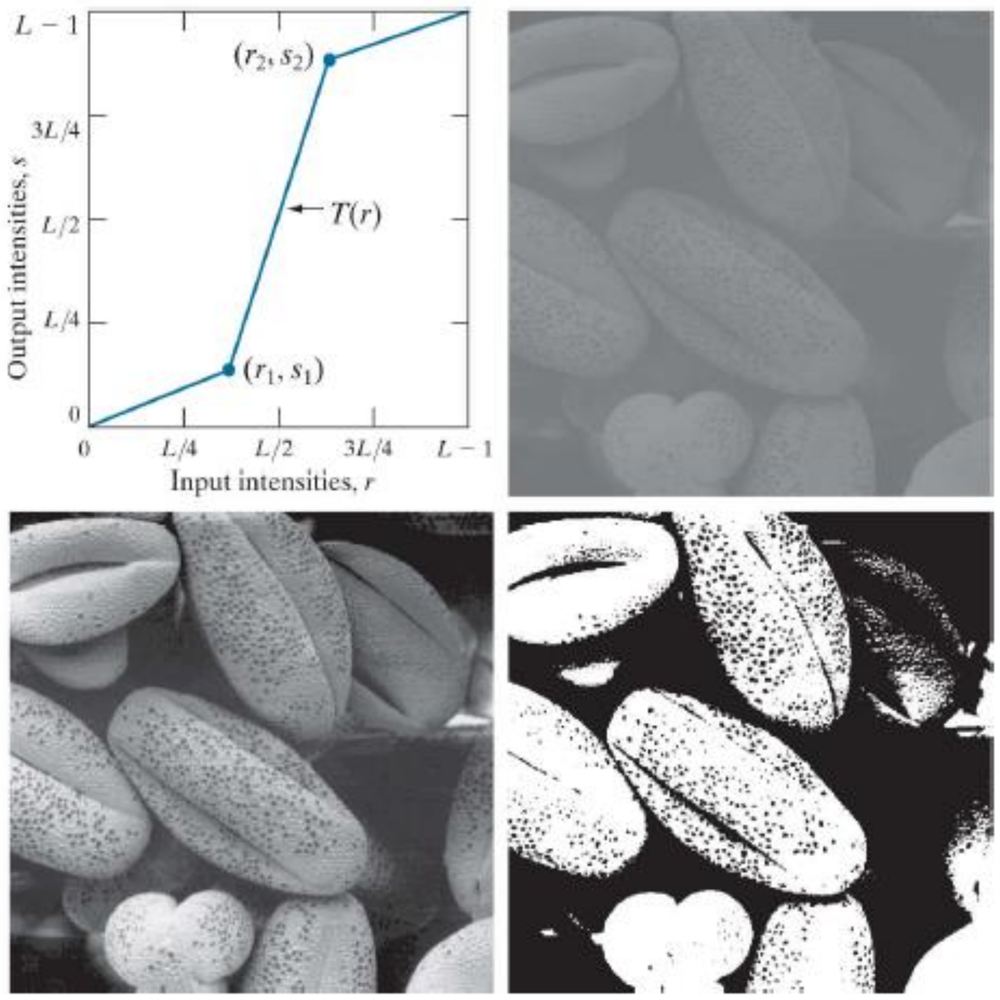


FIGURE 3.10
Contrast stretching. (a) Piecewise linear transformation function. (b) A low-contrast electron microscope image of pollen, magnified 700 times. (c) Result of contrast stretching. (d) Result of thresholding. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra, Australia.)

Figure 3.10(a) shows a typical transformation used for contrast stretching. The locations of points (r_t, s_a) and (r_2, s_2) control the shape of the transformation function. If $r_t = s_1$ and $r_2 = s_2$ the transformation is a linear function that produces no changes in intensity. If $r_1 = r_2$, $Sj = 0$, and $s_2 = L$ — 1 the transformation becomes a ***thresholding function*** that creates a binary image [see Fig. 3.2(b)].

Intermediate values of $(r_u\ s_t)$ and (s_2, r_2) produce various degrees of spread in the intensity levels of the output image, thus affecting its contrast. In general, $r_1 < r_2$ and s-^ < s_2 is assumed so that the function is single valued and monotonically increasing. This preserves the order of intensity levels, thus preventing the creation of intensity artifacts. 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 $(r_t, s_x) = (r_{min<}\ 0)$ and $(r_2, s_2) = (r_{max}, L$ — 1), where r_{min} and r_{max} denote the minimum and maximum intensity levels in the input image, respectively. The transformation stretched the intensity levels linearly to the full intensity range, $[0, L$ -1]. Finally, Fig. 3.10(d) shows the result of using the thresholding function, 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.

Intensity-Level Slicing

There are applications in which it is of interest to highlight a specific range of intensities in an image. Some of these applications include enhancing features in satellite imagery, such as masses of water, and enhancing flaws in X-ray images. The method, called ***intensity- level 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.

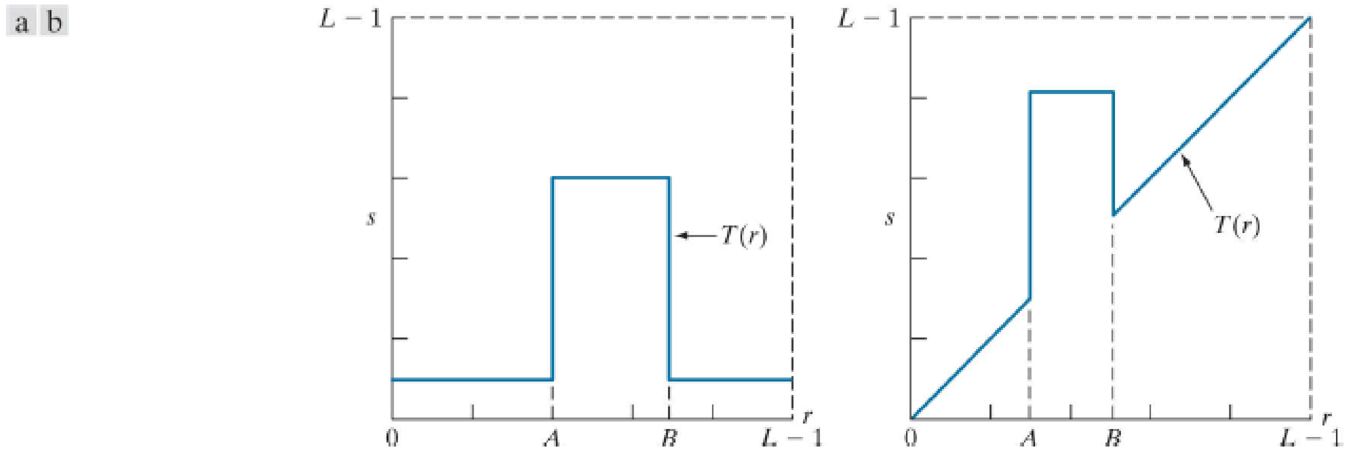


FIGURE 3.11
(a) This transformation function highlights range [A, B] and reduces all other intensities to a lower level. (b) This function highlights range [A, B] and leaves other intensities unchanged.

Bit-Plane Slicing

Pixel values are integers composed of bits. For example, values in a 256-level gray-scale image are composed of 8 bits (one byte). Instead of highlighting intensity-level ranges, we could highlight the contribution 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 one-bit planes, with plane 1 containing the lowest-order bit of all pixels in the image, and plane 8 all the highest-order bits.

The binary image for the 8th bit plane of an 8-bit image can be obtained by thresholding the input image with a transformation function that maps to 0 intensity values between 0 and 127, and maps to 1 values between 128 and 255.

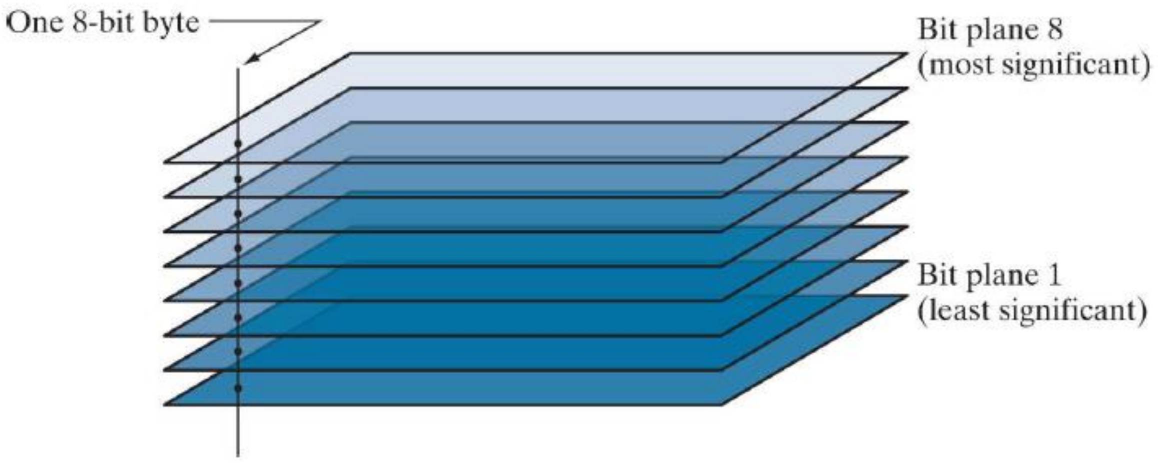


FIGURE 3.13
Bit-planes of an 8-bit image.

3.3- Histogram Processing

When dealing with histograms, let r_k, for $k = 0,1,2,\ldots\ldots L-1$, denote the intensities of an L-level digital image, f(x, y). The *unnormalized histogram* of f is defined as

$$h(r_k) = n_k \text{ for } k = 0,1,2,\ldots.. L-1 \qquad (3\text{-}6)$$

where n_k is the number of pixels in f with intensity r_k, and the subdivisions of the intensity scale are called *histogram bins.* Similarly, the *normalized histogram* of *f* is defined

$$p(r_k) = \frac{h(r_k)}{MN} = \frac{n_k}{MN} \qquad (3\text{-}7)$$

where, as usual, ***M*** and ***N*** are the number of image rows and columns, respectively. Mostly, we work with normalized histograms, which we refer to simply as ***histograms*** or ***image histograms.*** The sum of $p(r_k)$ for all values of ***k*** is always 1. The components of $p(r_k)$ are estimates of the probabilities of intensity levels occurring in an image. As you will learn in this section, histogram manipulation is a fundamental tool in image processing. Histograms are simple to compute and are also suitable for fast hardware implementations, thus making histogram-based techniques a popular tool for real-time image processing.

Histogram Equalization

Assuming initially continuous intensity values, 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

$$s = T(r) \quad 0 <= r <= L - 1 \qquad (3\text{-}8)$$

that produce an output intensity value, s, for a given intensity value *r* in the input image. We assume that

a. *T(r)* is a monotonic' increasing function in the interval $0 <= r <= L — 1$; and
b. $0 <= T(r) <= L - 1$ for $0 <= r <= L – 1$

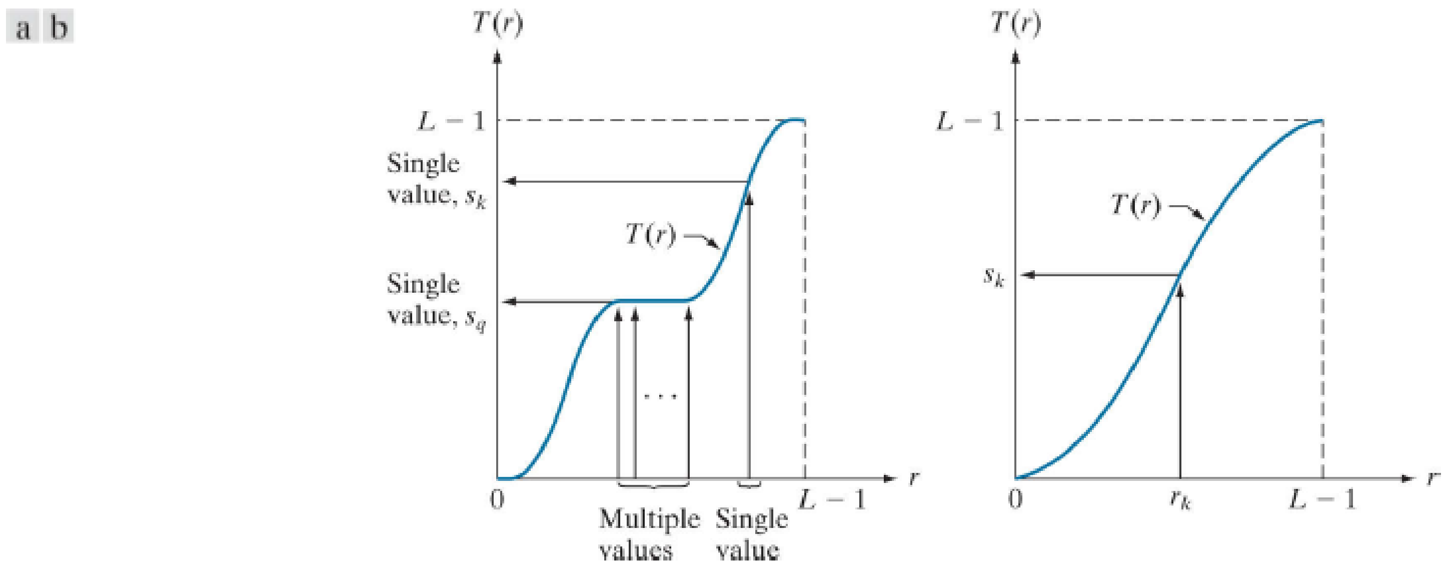


FIGURE 3.17
(a) Monotonic increasing function, showing how multiple values can map to a single value. (b) Strictly monotonic increasing function. This is a one-to-one mapping, both ways.

The condition in (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.

The intensity of an image may be viewed as a random variable in the interval [0, ***L*** -1]. Let $p_r(r)$ and $p_s(s)$ denote the PDFs of intensity values rand s in two different images. The subscripts on p indicate that p_r and p_s are different functions. A fundamental result from 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 as

$$P_s(s) = p_r(r)\left|\frac{dr}{ds}\right| \qquad (3\text{-}10)$$

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 is

$$s = T(r) = (L-1)\int_0^r p_r(w)dw \qquad (3\text{-}11)$$

where w is a dummy variable of integration. The integral on the right side is the ***cumulative distribution function*** (CDF) of random variable r. Because PDFs always are positive, and the integral of a function is the area under the function, it follows that the transformation function of Eq. (3-11) satisfies condition (a). This is 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, as it must for a PDF. Thus, the maximum value of s is ***L*** — 1, and condition (b) is satisfied also.

We use Eq. (3-10) to find the $p_s(s)$ corresponding to the transformation just discussed. We know from Leibniz's rule in calculus that the derivative of a definite integral with respect to its upper limit is the integrand evaluated at the limit. That is,

$$\begin{aligned}\frac{ds}{dr} &= \frac{dT(r)}{dr} \\ &= (L-1)\frac{d}{dr}\left[\int_0^r p_r(w)\,dw\right] \\ &= (L-1)p_r(r)\end{aligned} \qquad (3\text{-}12)$$

Substituting this result for *dr*/*ds* in Eq. (3-10) , and noting that all probability values are positive, gives the result

$$\begin{aligned}p_s(s) &= p_r(r)\left|\frac{dr}{ds}\right| \\ &= p_r(r)\left|\frac{1}{(L-1)p_r(r)}\right| \\ &= \frac{1}{L-1} \quad 0 \le s \le L-1\end{aligned} \qquad (3\text{-}13)$$

We recognize the form of $p_s(s)$ in the last line of this equation as a *uniform* probability density function. Thus, performing the intensity transformation in Eq. (3-11) yields a random variable, *s*, characterized by a uniform PDF. What is important is that $p_s(s)$ in Eq. (3-13) will *always* be uniform, *independently* of the form of $p_r(r)$. Figure 3.18 and the following example illustrate these concepts.

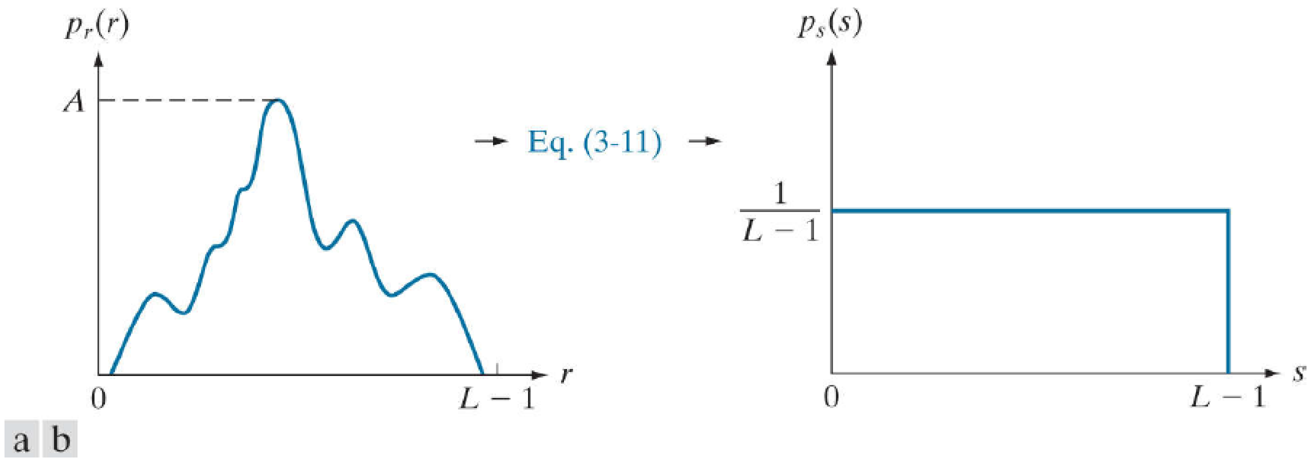


FIGURE 3.18

Histogram Matching (Specification)

Histogram equalization produces a transformation function that seeks to generate an output image with a uniform histogram. When automatic enhancement is desired, this is a good approach to consider because the results from this technique are predictable and the method is simple to implement. However, there are applications in which histogram equalization is not suitable. 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 images that have a specified histogram is called ***histogram matching*** or ***histogram specification.***

Consider for a moment continuous intensities rand z which, as before, we treat as random variables with PDFs $p_r(r)$ and $p_z(z)$, respectively. Here, rand z denote the intensity levels of the input and output (processed) images, respectively. We can estimate $p_r(r)$ from the given input image, and $p_z(z)$ is the ***specified*** PDF that we wish the output image to have.

Let s be a random variable with the property

$$s = T(r) = (L-1)\int_0^r p_r(w)dw \qquad (3\text{-}17)$$

where w is dummy variable of integration. This is the same as **Eq. (3-11)** , which we repeat here for convenience.

Define a function G on variable z with the property

$$G(z) = (L-1)\int_0^z p_z(v)dv = s \qquad (3\text{-}18)$$

where v is a dummy variable of integration. It follows from the preceding two equations that $G(z) = s = T(r)$ and, therefore, that z must satisfy the condition

$$z = G^{-1}(s) = G^{-1}[T(r)] \qquad (3\text{-}19)$$

The transformation function $T(r)$ can be obtained using **Eq. (3-17)** after $p_r(r)$ has been estimated using the input image. Similarly, function $G(z)$ can be obtained from **Eq. (3-18)** because $p_z(z)$ is given.

Equations (3-17) through **(3-19)** imply that an image whose intensity levels have a specified PDF can be obtained using the following procedure:

1. Obtain $p_r(r)$ from the input image to use in **Eq. (3-17)** .
2. Use the specified PDF, $p_z(z)$, in **Eq. (3-18)** to obtain the function $G(z)$.
3. Compute the inverse transformation $z = G^{-1}(s)$; this is a mapping from s to z, the latter being the values that have the specified PDF.

4. Obtain the output image by first equalizing the input image using Eq. (3-17) ; 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 processed with this transformation, the PDF of the output image, $p_z(z)$, will be equal to the specified PDF.

Because s is related to r by $T(r)$, it is possible for the mapping that yields z from s to be expressed directly in terms of r. In general, however, finding analytical expressions for G^{-1} is not a trivial task.

0) , we may summarize the procedure for discrete histogram specification as follows:

1. Compute the histogram, $p_r(r)$, of the input image, and use it in Eq. (3-20) to map the intensities in the input image to the intensities in the histogram-equalized image. Round the resulting values, s_k, to the integer range $[0, L-1]$.
2. Compute all values of function $G(z_q)$ using the Eq. (3-21) 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 rounded values of G in a lookup 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. Store these mappings from s to z. When more than one value of z_q gives the same match (i.e., the mapping is not unique), choose the smallest value by convention.
4. Form the histogram-specified image by mapping every equalized pixel with value s_k to the corresponding pixel with value z_q in the histogram-specified image, using the mappings found in Step 3.

Exact Histogram Matching (Specification)

The discrete histogram equalization and specification methods generate images with histograms whose shapes generally do not resemble the shape of the specified histograms. These methods can produce effective results. However, there are applications that can benefit from a histogram processing technique capable of generating images whose histograms truly match specified shapes. Examples include normalizing large image data sets used for testing and validation in the pharmaceutical industry, establishing a set of "golden images" for calibrating imaging systems, and establishing a norm for consistent medical image analysis and interpretation by humans. Also, as you will see later, being able to generate images with specified histograms simplifies experimentation when seeking histogram shapes that will produce a desired result.

Foundation

Consider a specified histogram that we wish an image to have:

$$H = \{h(0), h(1), h(2) \ldots\ldots\ldots h(L-1)\} \qquad (3\text{-}24)$$

where ***L*** is the number of discrete intensity levels, and ***h(j)*** is the number of pixels with intensity level ***j.*** This histogram is assumed to be both ***unnormalized*** and ***valid,*** in the sense that the sum of its components equals the total number of pixels in the image (which is always an integer)

$$\sum_{j=0}^{L-1} h(j) = MN \qquad (3\text{-}25)$$

As usual, *M* and *N* are the number of rows and columns in the image, respectively.

Given a digital image and a histogram satisfying the preceding conditions, the procedure used for exact histogram specification consists of three

basic steps:

a. Order the image pixels according to a predefined criterion.
b. Split the ordered pixels into ***L*** groups, such that group ***j*** has ***h(j)*** pixels.
c. Assign intensity value ***j*** to all pixels in group ***j.***