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. Thus, developing an understanding of basic characteristics of human visual perception as a first step in our journey through this book is appropriate. In particular, our interest is in the elementary mechanics of how images are formed and perceived by humans. We are interested in learning the physical limitations of human vision in terms of factors that also are used in our work with digital images. Factors such as how human and electronic imaging devices compare in terms of resolution and ability to adapt to changes in illumination are not only interesting, they are also important from a practical point of view.

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.

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

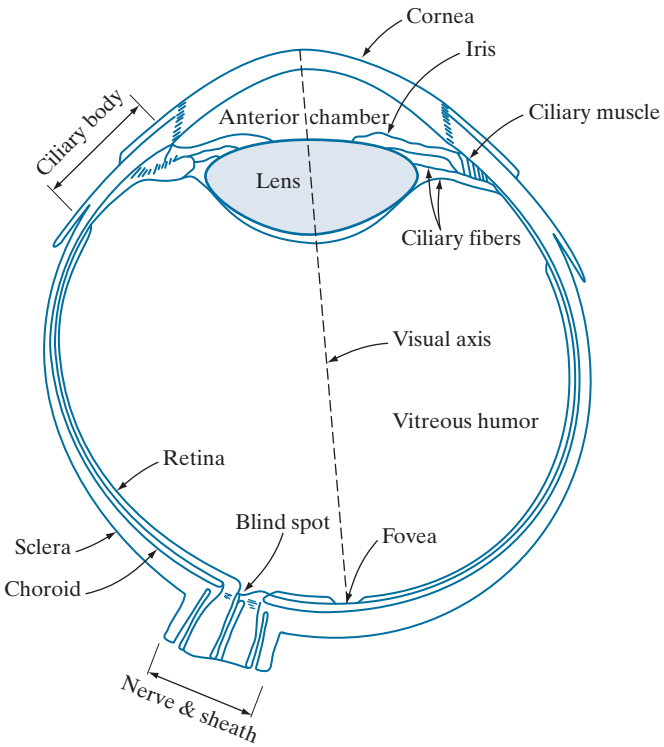


FIGURE 2.1 Simplified diagram of a cross section of the human eye.

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. Note in Fig. 2.2 that cones are most dense in the center area of the fovea, and that rods increase in density from the center out to approximately 20° off axis. Then, their density decreases out to the periphery of the retina.

The fovea itself is a circular indentation in the retina of about 1.5 mm in diameter, so it has an area of approximately 1.77 mm^2. As Fig. 2.2 shows, the density of cones in that area of the retina is on the order of 150,000 elements per mm^2. Based on these figures, the number of cones in the fovea, which is the region of highest acuity

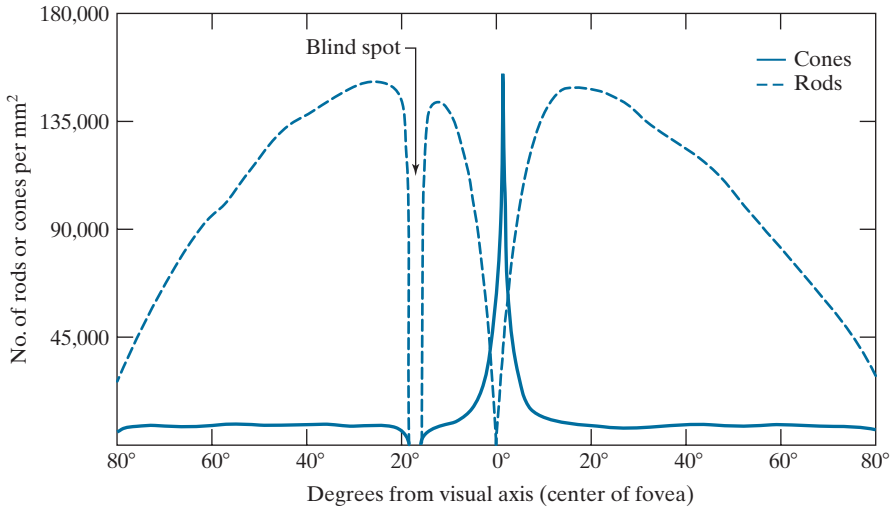


FIGURE 2.2 Distribution of rods and cones in the retina.

in the eye, is about 265,000 elements. Modern electronic imaging chips exceed this number by a large factor. While the ability of humans to integrate intelligence and experience with vision makes purely quantitative comparisons somewhat superficial, keep in mind for future discussions that electronic imaging sensors can easily exceed the capability of the eye in resolving image detail.

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.

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

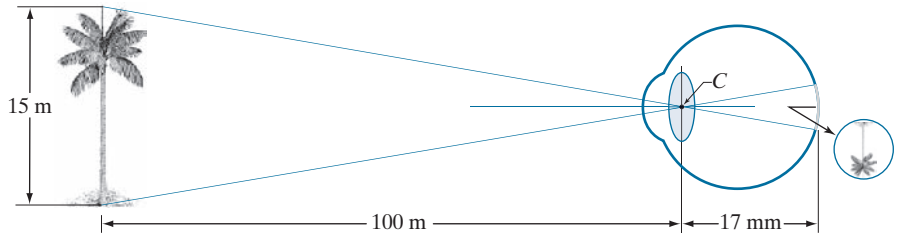


FIGURE 2.3
Graphical representation of the eye looking at a palm tree. Point C is the focal center of the lens.

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 10^6. 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,

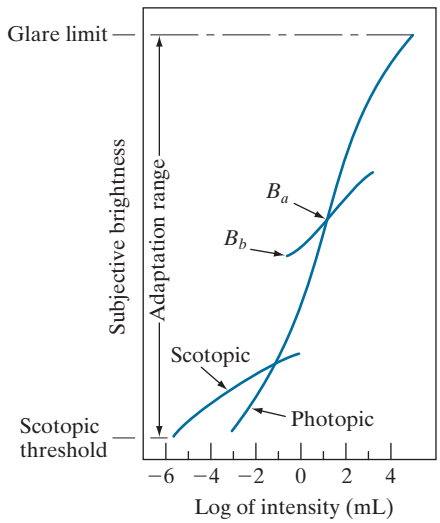


FIGURE 2.4
Range of subjective brightness sensations showing a particular adaptation level, B_a.

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.

The ability of the eye to discriminate between *changes* in light intensity at any specific adaptation level is of considerable interest. A classic experiment used to determine the capability of the human visual system for brightness discrimination consists of having a subject look at a flat, uniformly illuminated area large enough to occupy the entire field of view. This area typically is a diffuser, such as opaque glass, illuminated from behind by a light source, I, with variable intensity. To this field is added an increment of illumination, ΔI, in the form of a short-duration flash that appears as a circle in the center of the uniformly illuminated field, as Fig. 2.5 shows.

If ΔI is not bright enough, the subject says "no," indicating no perceivable change. As ΔI gets stronger, the subject may give a positive response of "yes," indicating a perceived change. Finally, when ΔI is strong enough, the subject will give a response of "yes" all the time. The quantity $\Delta I_c / I$, where ΔI_c is the increment of illumination discriminable 50% of the time with background illumination I, is called the *Weber ratio*. A small value of $\Delta I_c / I$ means that a small percentage change in intensity is discriminable. This represents "good" brightness discrimination. Conversely, a large value of $\Delta I_c / I$ means that a large percentage change in intensity is required for the eye to detect the change. This represents "poor" brightness discrimination.

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

If the background illumination is held constant and the intensity of the other source, instead of flashing, is now allowed to vary incrementally from never being perceived to always being perceived, the typical observer can discern a total of one to two dozen different intensity changes. Roughly, this result is related to the number of different intensities a person can see at any one *point* or *small area* in a monochrome image. This does not mean that an image can be represented by such a small number of intensity values because, as the eye roams about the image, the average

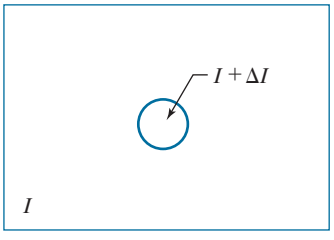


FIGURE 2.5 Basic experimental setup used to characterize brightness discrimination.

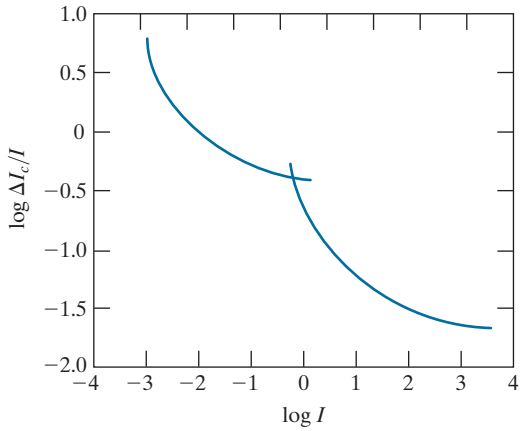


FIGURE 2.6
A typical plot of the Weber ratio as a function of intensity.

background changes, thus allowing a *different* set of incremental changes to be detected at each new adaptation level. The net result is that the eye is capable of a broader range of *overall* intensity discrimination. In fact, as we will show in Section 2.4, the eye is capable of detecting objectionable effects in monochrome images whose overall intensity is represented by fewer than approximately two dozen levels.

Two phenomena demonstrate that perceived brightness is not a simple function of intensity. The first is based on the fact that the visual system tends to undershoot or overshoot around the boundary of regions of different intensities. Figure 2.7(a) shows a striking example of this phenomenon. Although the intensity of the stripes

a
b
c

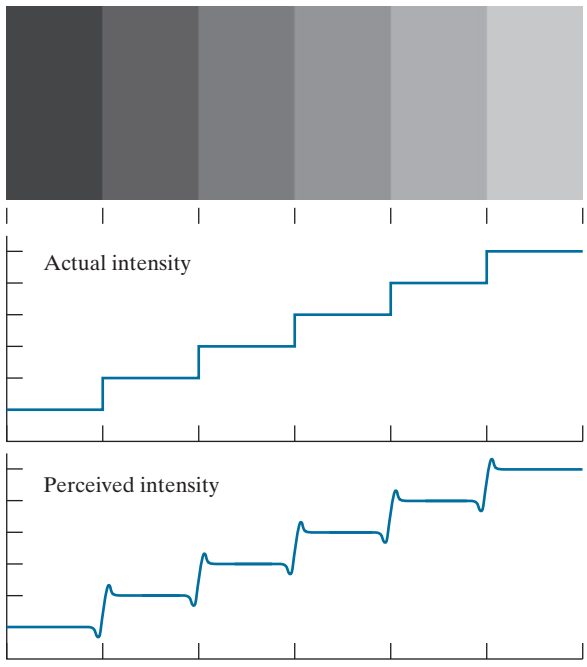


FIGURE 2.7
Illustration of the Mach band effect. Perceived intensity is not a simple function of actual intensity.

a b c

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

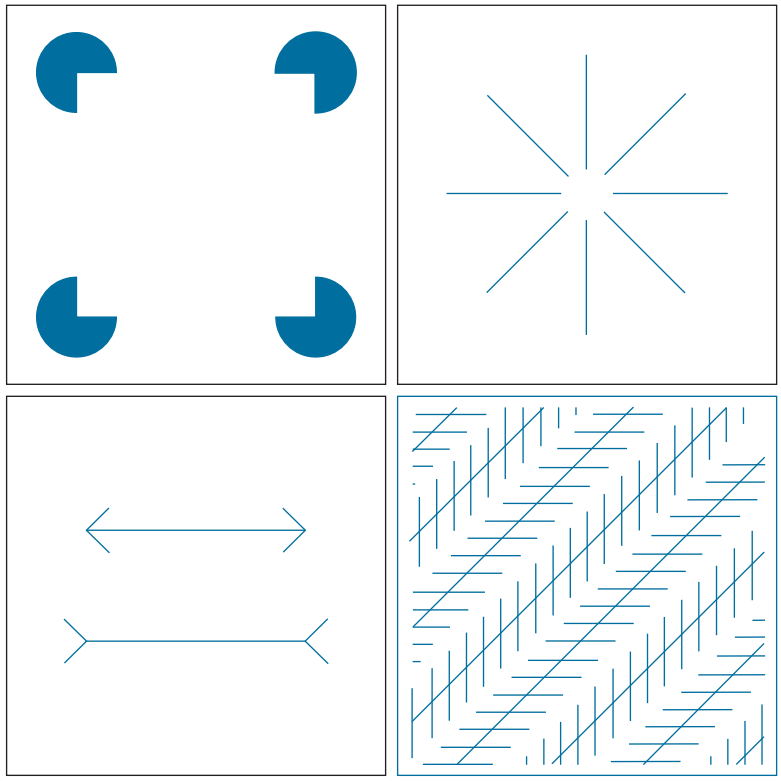
is constant [see Fig. 2.7(b)], we actually perceive a brightness pattern that is strongly scalloped near the boundaries, as Fig. 2.7(c) shows. These perceived scalloped bands are called *Mach bands* after Ernst Mach, who first described the phenomenon in 1865.

The second phenomenon, called *simultaneous contrast*, is that a region's perceived brightness does not depend only on its intensity, as Fig. 2.8 demonstrates. All the center squares have exactly the same intensity, but each appears to the eye to become darker as the background gets lighter. A more familiar example is a piece of paper that looks white when lying on a desk, but can appear totally black when used to shield the eyes while looking directly at a bright sky.

Other examples of human perception phenomena are *optical illusions*, in which the eye fills in nonexisting details or wrongly perceives geometrical properties of objects. Figure 2.9 shows some examples. In Fig. 2.9(a), the outline of a square is seen clearly, despite the fact that no lines defining such a figure are part of the image. The same effect, this time with a circle, can be seen in Fig. 2.9(b); note how just a few lines are sufficient to give the illusion of a complete circle. The two horizontal line segments in Fig. 2.9(c) are of the same length, but one appears shorter than the other. Finally, all long lines in Fig. 2.9(d) are equidistant and parallel. Yet, the crosshatching creates the illusion that those lines are far from being parallel.

2.2 LIGHT AND THE ELECTROMAGNETIC SPECTRUM

The electromagnetic spectrum was introduced in Section 1.3. We now consider this topic in more detail. In 1666, Sir Isaac Newton discovered that when a beam of sunlight 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. We showed examples in Section 1.3 of images in most of the bands in the EM spectrum.



a b
c d

FIGURE 2.9 Some well-known optical illusions.

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.

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 μm and equal to 10^{-6} m) and *nanometers* (denoted nm and equal to 10^{-9} m) being used just as frequently. Frequency is measured in *Hertz* (Hz), with one 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 λ (Fig. 2.11), or they can be thought of as a stream of massless particles,

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

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

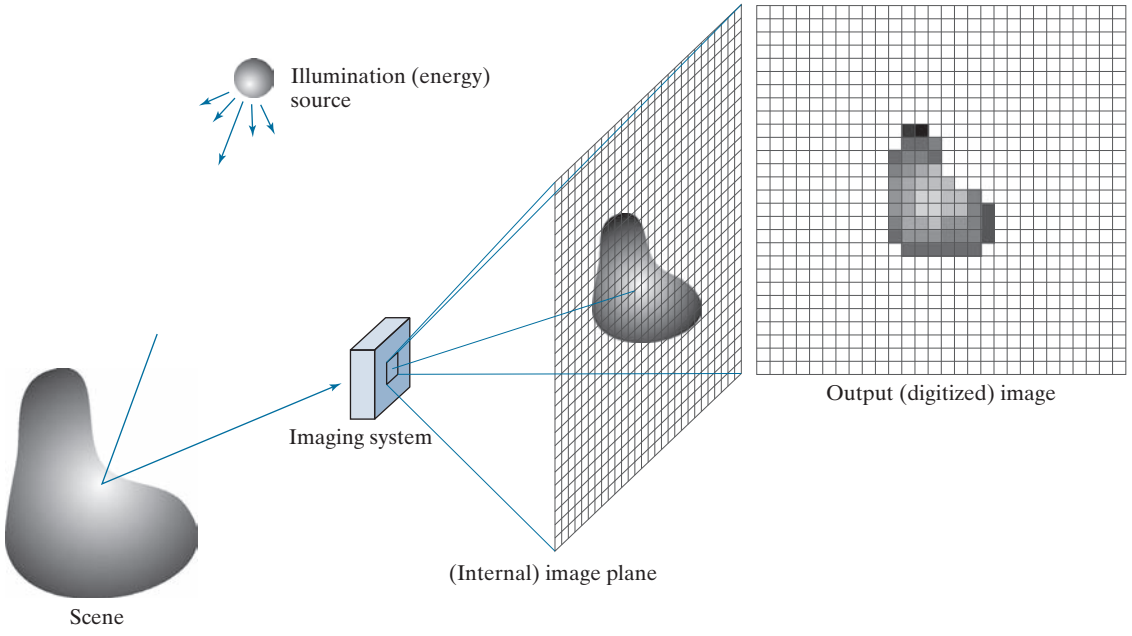
In some cases, the source is imaged directly, as in obtaining images of the sun.

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). Converting images into digital form is the topic of Section 2.4.

A SIMPLE IMAGE FORMATION MODEL

As introduced in Section 1.1, we denote images by two-dimensional functions of the form $f(x, y)$. The value of f at 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 (see Section 2.6 regarding intensity scaling).



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.

$$0 \leq f(x,y) < \infty \tag{2-3}$$

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 $f(x,y)$:

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

where

$$0 \leq i(x,y) < \infty \tag{2-5}$$

and

$$0 \leq r(x,y) \leq 1 \tag{2-6}$$

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. These expressions are applicable also to images formed via transmission of the illumination through a medium, such as a

chest X-ray. In this case, we would deal with a *transmissivity* instead of a *reflectivity* function, but the limits would be the same as in Eq. (2-6), and the image function formed would be modeled as the product in Eq. (2-4).

EXAMPLE 2.1: Some typical values of illumination and reflectance.

The following numerical quantities illustrate some typical values of illumination and reflectance for visible light. On a clear day, the sun may produce in excess of 90,000 lm/m^2 of illumination on the surface of the earth. This value decreases to less than 10,000 lm/m^2 on a cloudy day. On a clear evening, a full moon yields about 0.1 lm/m^2 of illumination. The typical illumination level in a commercial office is about 1,000 lm/m^2. Similarly, the following are typical values of $r(x, y)$: 0.01 for black velvet, 0.65 for stainless steel, 0.80 for flat-white wall paint, 0.90 for silver-plated metal, and 0.93 for snow.

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

$$\ell = f(x, y) \tag{2-7}$$

From Eqs. (2-4) through (2-6) it is evident that ℓ lies in the range

$$L_{\min} \leq \ell \leq L_{\max} \tag{2-8}$$

In theory, the requirement on $L_{\min}$ is that it be nonnegative, and on $L_{\max}$ that it be finite. In practice, $L_{\min} = i_{\min} r_{\min}$ and $L_{\max} = i_{\max} r_{\max}$. From Example 2.1, using average office illumination and reflectance values as guidelines, we may expect $L_{\min} \approx 10$ and $L_{\max} \approx 1000$ to be typical indoor values in the absence of additional illumination. The units of these quantities are lum/m^2. However, actual units seldom are of interest, except in cases where photometric measurements are being performed.

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

2.4 IMAGE SAMPLING AND QUANTIZATION

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

As discussed in the previous section, there are numerous ways to acquire images, but our objective in all is the same: to generate digital images from sensed data. The output of most sensors is a continuous voltage waveform whose amplitude and spatial behavior are related to the physical phenomenon being sensed. To create a digital image, we need to convert the continuous sensed data into 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

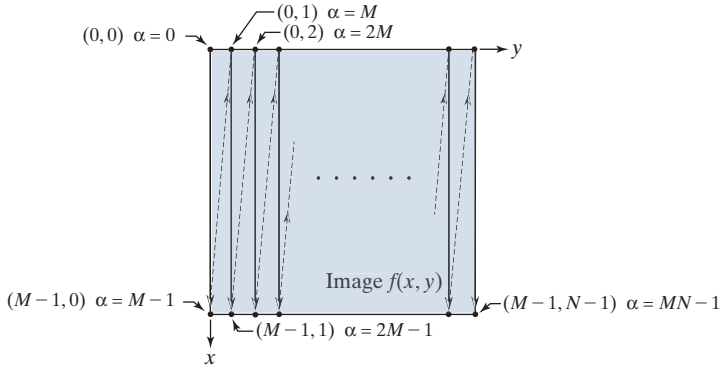


FIGURE 2.22 Illustration of column scanning for generating linear indices. Shown are several 2-D coordinates (in parentheses) and their corresponding linear indices.

Conversely, the coordinate indices for a given linear index value α are given by the equations[†]

$$x = \alpha \bmod M \tag{2-15}$$

and

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

Recall that $\alpha \bmod M$ means "the remainder of the division of α by M." This is a formal way of stating that row numbers repeat themselves at the start of every column. Thus, when $\alpha = 0$, the remainder of the division of 0 by M is 0, so $x = 0$. When $\alpha = 1$, the remainder is 1, and so $x = 1$. You can see that x will continue to be equal to α until $\alpha = M - 1$. When $\alpha = M$ (which is at the beginning of the second column), the remainder is 0, and thus $x = 0$ again, and it increases by 1 until the next column is reached, when the pattern repeats itself. Similar comments apply to Eq. (2-16). See Problem 2.11 for a derivation of the preceding two equations.

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 $2W$, and there are $W/2$ 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

[†]When working with modular number systems, it is more accurate to write $x \equiv \alpha \bmod M$, where the symbol $\equiv$ means *congruence*. However, our interest here is just on converting from linear to coordinate indexing, so we use the more familiar equal sign.

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 1024×1024 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, as we mentioned when discussing Eq. (2-11). The most common number is 8 bits, with 16 bits being used in some applications in which enhancement of specific intensity ranges is necessary. Intensity quantization using 32 bits is rare. Sometimes one finds systems that can digitize the intensity levels of an image using 10 or 12 bits, but these are 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 (see Section 2.1). The following two examples illustrate the effects of spatial and intensity resolution on discernible detail. Later in this section, we will discuss how these two parameters interact in determining perceived image quality.

EXAMPLE 2.2: Effects of reducing the spatial resolution of a digital image.

Figure 2.23 shows the effects of reducing the spatial resolution of an image. The images in Figs. 2.23(a) through (d) have resolutions of 930, 300, 150, and 72 dpi, respectively. Naturally, the lower resolution images are smaller than the original image in (a). For example, the original image is of size 2136×2140 pixels, but the 72 dpi image is an array of only 165×166 pixels. In order to facilitate comparisons, all the smaller images were zoomed back to the original size (the method used for zooming will be discussed later in this section). This is somewhat equivalent to "getting closer" to the smaller images so that we can make comparable statements about visible details.

There are some small visual differences between Figs. 2.23(a) and (b), the most notable being a slight distortion in the seconds marker pointing to 60 on the right side of the chronometer. For the most part, however, Fig. 2.23(b) is quite acceptable. In fact, 300 dpi is the typical minimum image spatial resolution used for book publishing, so one would not expect to see much difference between these two images. Figure 2.23(c) begins to show visible degradation (see, for example, the outer edges of the chronometer

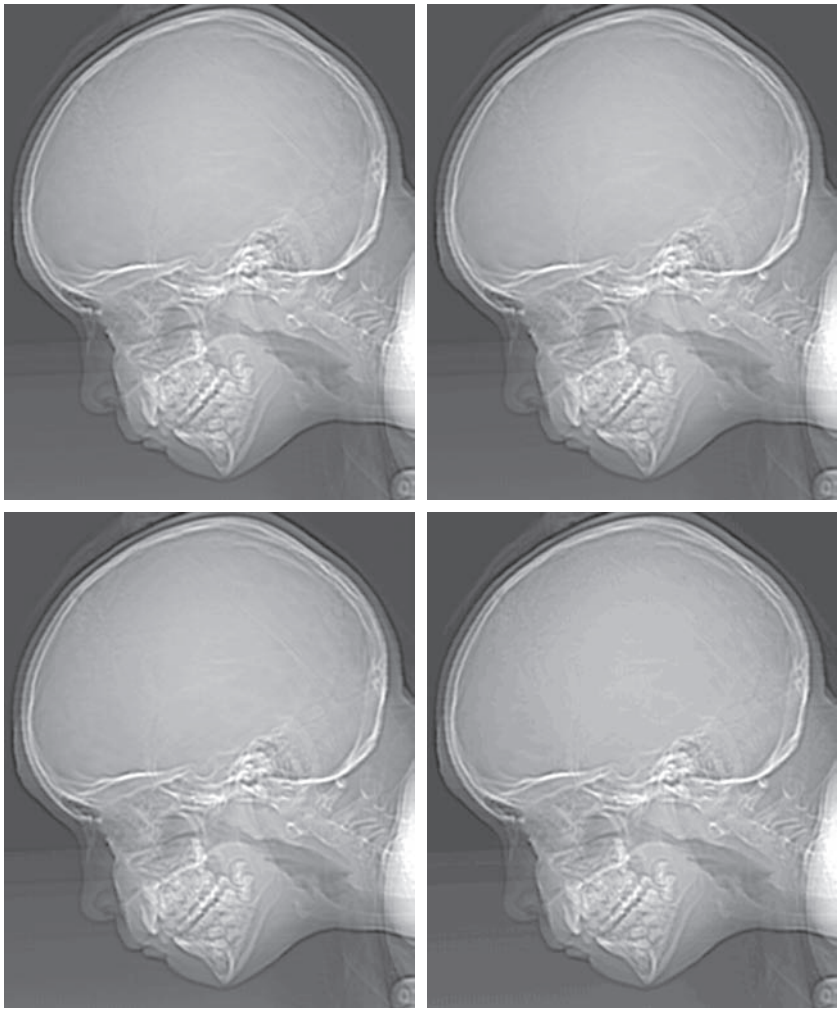
a b
c d

FIGURE 2.23
Effects of reducing spatial resolution. The images shown are at:
(a) 930 dpi,
(b) 300 dpi,
(c) 150 dpi, and
(d) 72 dpi.

case and compare the seconds marker with the previous two images). The numbers also show visible degradation. Figure 2.23(d) shows degradation that is visible in most features of the image. When printing at such low resolutions, the printing and publishing industry uses a number of techniques (such as locally varying the pixel size) to produce much better results than those in Fig. 2.23(d). Also, as we will show later in this section, it is possible to improve on the results of Fig. 2.23 by the choice of interpolation method used.

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

Figure 2.24(a) is a 774×640 CT projection image, displayed using 256 intensity levels (see Chapter 1 regarding CT images). The objective of this example is to reduce the number of intensities of the image from 256 to 2 in integer powers of 2, while keeping the spatial resolution constant. Figures 2.24(b) through (d) were obtained by reducing the number of intensity levels to 128, 64, and 32, respectively (we will discuss in Chapter 3 how to reduce the number of levels).

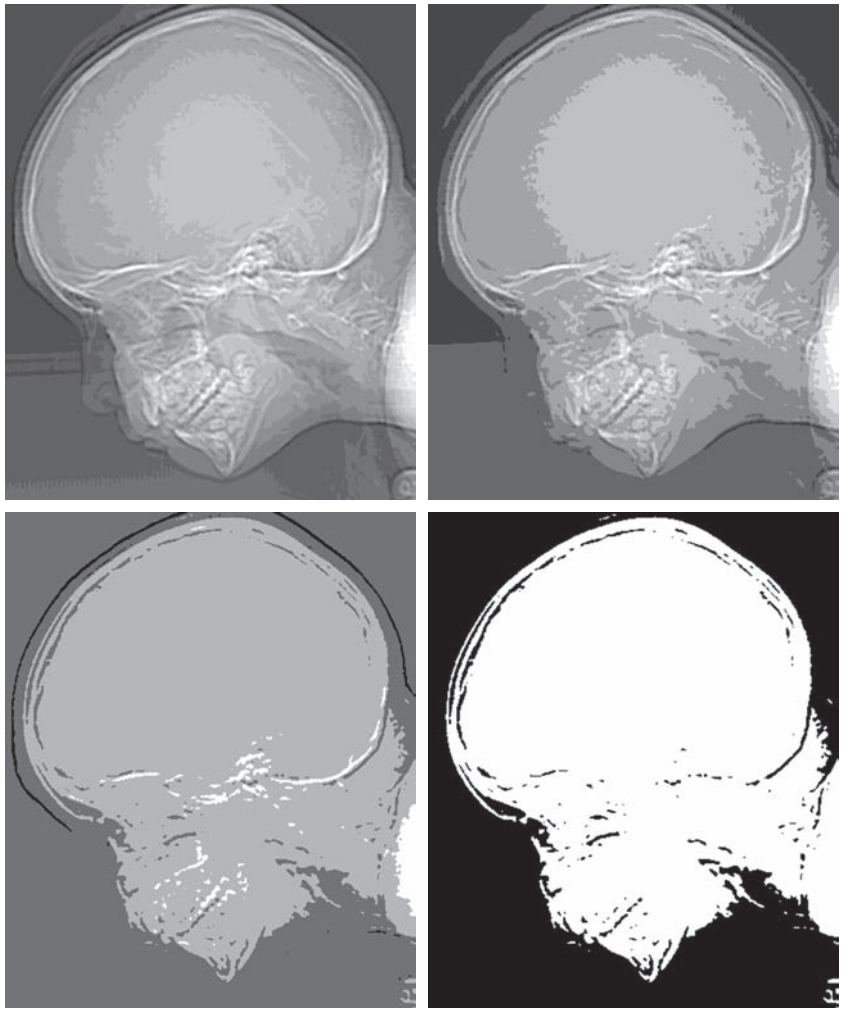


a b
c d

FIGURE 2.24
(a) 774 × 640, 256-level image. (b)-(d) Image displayed in 128, 64, and 32 intensity levels, while keeping the spatial resolution constant. (Original image courtesy of the Dr. David R. Pickens, Department of Radiology & Radiological Sciences, Vanderbilt University Medical Center.)

The 128- and 64-level images are visually identical for all practical purposes. However, the 32-level image in Fig. 2.24(d) has a set of almost imperceptible, very fine ridge-like structures in areas of constant intensity. These structures are clearly visible in the 16-level image in Fig. 2.24(e). This effect, caused by using an insufficient number of intensity levels in smooth areas of a digital image, is called *false contouring*, so named because the ridges resemble topographic contours in a map. False contouring generally is quite objectionable in images displayed using 16 or fewer uniformly spaced intensity levels, as the images in Figs. 2.24(e)-(h) show.

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



e f
g h

FIGURE 2.24
(*Continued*)
(e)-(h) Image displayed in 16, 8, 4, and 2 intensity levels.

The results in Examples 2.2 and 2.3 illustrate the effects produced on image quality by varying spatial and intensity resolution independently. However, these results did not consider any relationships that might exist between these two parameters. An early study by Huang [1965] attempted to quantify experimentally the effects on image quality produced by the interaction of these two variables. The experiment consisted of a set of subjective tests. Images similar to those shown in Fig. 2.25 were used. The woman's face represents an image with relatively little detail; the picture of the cameraman contains an intermediate amount of detail; and the crowd picture contains, by comparison, a large amount of detail.

Sets of these three types of images of various sizes and intensity resolution were generated by varying N and k [see Eq. (2-13)]. Observers were then asked to rank

a b c

FIGURE 2.25 (a) Image with a low level of detail. (b) Image with a medium level of detail. (c) Image with a relatively large amount of detail. (Image (b) courtesy of the Massachusetts Institute of Technology.)

them according to their subjective quality. Results were summarized in the form of so-called *isopreference curves* in the *Nk*-plane. (Figure 2.26 shows average isopreference curves representative of the types of images in Fig. 2.25.) Each point in the *Nk*-plane represents an image having values of N and k equal to the coordinates of that point. Points lying on an isopreference curve correspond to images of equal subjective quality. It was found in the course of the experiments that the isopreference curves tended to shift right and upward, but their shapes in each of the three image categories were similar to those in Fig. 2.26. These results were not unexpected, because a shift up and right in the curves simply means larger values for N and k, which implies better picture quality.

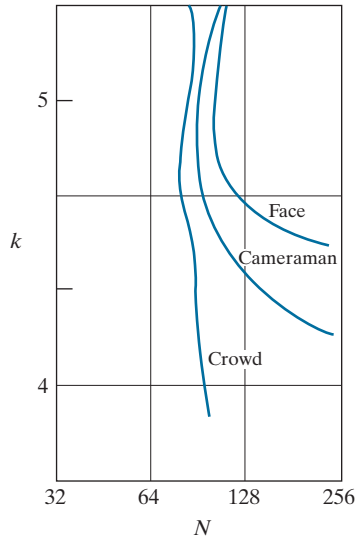


FIGURE 2.26 Representative isopreference curves for the three types of images in Fig. 2.25.

Observe that isopreference curves tend to become more vertical as the detail in the image increases. This result suggests that for images with a large amount of detail only a few intensity levels may be needed. For example, the isopreference curve in Fig. 2.26 corresponding to the crowd is nearly vertical. This indicates that, for a fixed value of N, the perceived quality for this type of image is nearly independent of the number of intensity levels used (for the range of intensity levels shown in Fig. 2.26). The perceived quality in the other two image categories remained the same in some intervals in which the number of samples was increased, but the number of intensity levels actually decreased. The most likely reason for this result is that a decrease in k tends to increase the apparent contrast, a visual effect often perceived as improved image quality.

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. Uses of interpolation in applications such as rotation and geometric corrections will be discussed in Section 2.6.

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 500×500 pixels has to be enlarged 1.5 times to 750×750 pixels. A simple way to visualize zooming is to create an imaginary 750×750 grid with the same pixel spacing as the original image, then shrink it so that it exactly overlays the original image. Obviously, the pixel spacing in the shrunken 750×750 grid will be less than the pixel spacing in the original image. To 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 750×750 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 (see Section 2.5 regarding neighborhoods). 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

Contrary to what the name suggests, bilinear interpolation is *not* a linear operation because it involves multiplication of coordinates (which is not a linear operation). See Eq. (2-17).

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

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.

Although images are displayed with integer coordinates, it is possible during processing to work with *subpixel accuracy* by increasing the size of the image using interpolation to "fill the gaps" between pixels in the original image.

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

Figure 2.27(a) is the same as Fig. 2.23(d), which was obtained by reducing the resolution of the 930 dpi image in Fig. 2.23(a) to 72 dpi (the size shrank from 2136×2140 to 165×166 pixels) and then zooming the reduced image back to its original size. To generate Fig. 2.23(d) we used nearest neighbor interpolation both to shrink and zoom the image. As noted earlier, the result in Fig. 2.27(a) is rather poor. Figures 2.27(b) and (c) are the results of repeating the same procedure but using, respectively, bilinear and bicubic interpolation for both shrinking and zooming. The result obtained by using bilinear interpolation is a significant improvement over nearest neighbor interpolation, but the resulting image is blurred slightly. Much sharper results can be obtained using bicubic interpolation, as Fig. 2.27(c) shows.

a b c

FIGURE 2.27 (a) Image reduced to 72 dpi and zoomed back to its original 930 dpi using nearest neighbor interpolation. This figure is the same as Fig. 2.23(d). (b) Image reduced to 72 dpi and zoomed using bilinear interpolation. (c) Same as (b) but using bicubic interpolation.

On the other hand, suppose that we are working with the max operation, whose function is to find the maximum value of the pixels in an image. For our purposes here, the simplest way to prove that this operator is nonlinear is to find an example that fails the test in Eq. (2-23). Consider the following two images

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

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.

As you will see in the next three chapters, linear operations are exceptionally important because they encompass a large body of theoretical and practical results that are applicable to image processing. The scope of nonlinear operations is considerably more limited. However, you will encounter in the following chapters several nonlinear image processing operations whose performance far exceeds what is achievable by their linear counterparts.

ARITHMETIC OPERATIONS

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

$$\begin{aligned} s(x, y) &= f(x, y) + g(x, y) \\ d(x, y) &= f(x, y) - g(x, y) \\ p(x, y) &= f(x, y) \times g(x, y) \\ v(x, y) &= f(x, y) \div g(x, y) \end{aligned} \tag{2-24}$$

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 $x = 0, 1, 2, \ldots, M - 1$ and $y = 0, 1, 2, \ldots, N - 1$. As usual, M and N are the row and column sizes of the images. Clearly, s, d, p, and v are images of size $M \times N$ also. Note that image arithmetic in the manner just defined involves images of the same size. The following examples illustrate the important role of arithmetic operations in digital image processing.

EXAMPLE 2.5: Using image addition (averaging) for noise reduction.

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

$$g(x, y) = f(x, y) + \eta(x, y) \tag{2-25}$$

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

If the noise satisfies the constraints just stated, it can be shown (Problem 2.26) that if an image $\bar{g}(x, y)$ is formed by averaging K different noisy images,

$$\bar{g}(x, y) = \frac{1}{K}\sum_{i=1}^{K} g_i(x, y) \tag{2-26}$$

then it follows that

$$E\{\bar{g}(x, y)\} = f(x, y) \tag{2-27}$$

and

$$\sigma^2_{\bar{g}(x,y)} = \frac{1}{K}\sigma^2_{\eta(x,y)} \tag{2-28}$$

where $E\{\bar{g}(x, y)\}$ is the expected value of $\bar{g}(x, y)$, and $\sigma^2_{\bar{g}(x,y)}$ and $\sigma^2_{\eta(x,y)}$ are the variances of $\bar{g}(x, y)$ and $\eta(x, y)$, respectively, all at coordinates (x, y). These variances are arrays of the same size as the input image, and there is a scalar variance value for each pixel location.

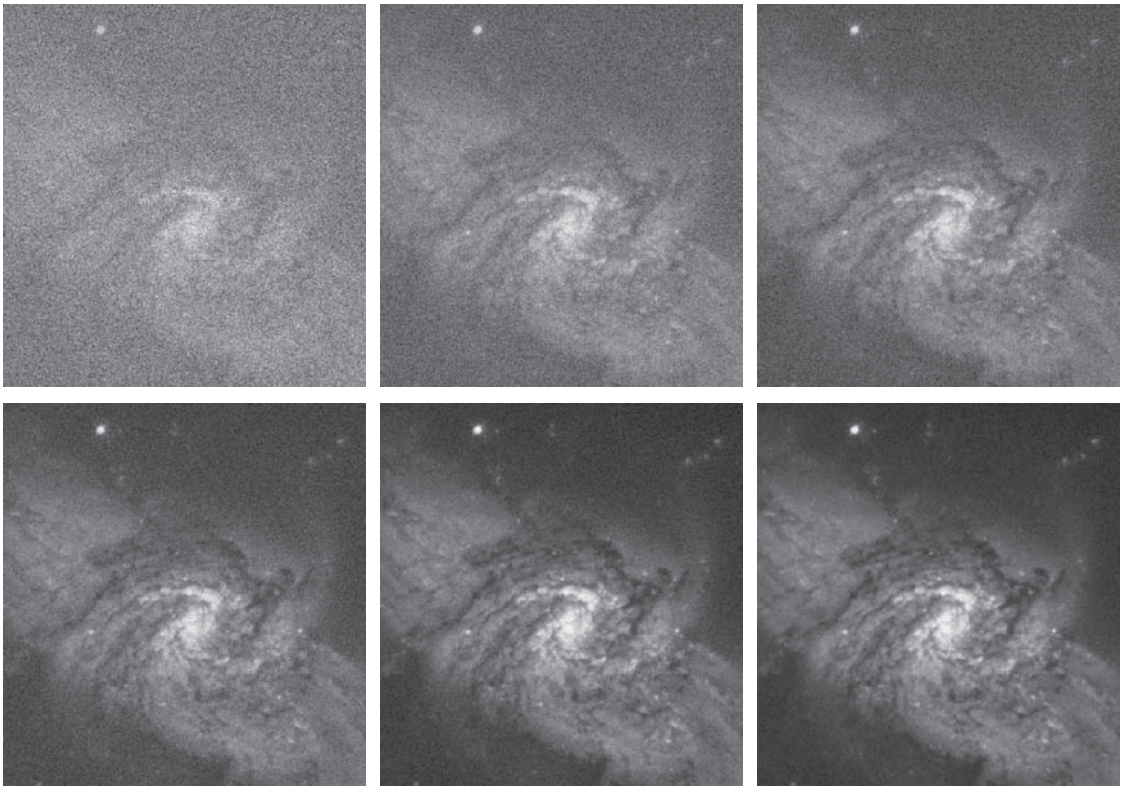
The standard deviation (square root of the variance) at any point (x, y) in the average image is

$$\sigma_{\bar{g}(x,y)} = \frac{1}{\sqrt{K}}\sigma_{\eta(x,y)} \tag{2-29}$$

As K increases, Eqs. (2-28) and (2-29) indicate that the variability (as measured by the variance or the standard deviation) of the pixel values at each location (x, y) decreases. Because $E\{\bar{g}(x, y)\} = f(x, y)$, this means that $\bar{g}(x, y)$ approaches the noiseless image $f(x, y)$ as the number of noisy images used in the averaging process increases. In order to avoid blurring and other artifacts in the output (average) image, it is necessary that the images $g_i(x, y)$ be *registered* (i.e., spatially aligned).

An important application of image averaging is in the field of astronomy, where imaging under very low light levels often cause sensor noise to render individual images virtually useless for analysis (lowering the temperature of the sensor helps reduce noise). Figure 2.29(a) shows an 8-bit image of the Galaxy Pair NGC 3314, in which noise corruption was simulated by adding to it Gaussian noise with zero mean and a standard deviation of 64 intensity levels. This image, which is representative of noisy astronomical images taken under low light conditions, is useless for all practical purposes. Figures 2.29(b) through (f) show the results of averaging 5, 10, 20, 50, and 100 images, respectively. We see from Fig. 2.29(b) that an average of only 10 images resulted in some visible improvement. According to Eq.

[†]The variance of a random variable z with mean $\bar{z}$ is defined as $E\{(z - \bar{z})^2\}$, where $E\{\cdot\}$ is the expected value of the argument. The covariance of two random variables z_i and z_j is defined as $E\{(z_i - \bar{z}_i)(z_j - \bar{z}_j)\}$. If the variables are uncorrelated, their covariance is 0, and vice versa. (Do not confuse correlation and statistical independence. If two random variables are statistically independent, their correlation is zero. However, the converse is not true in general.)



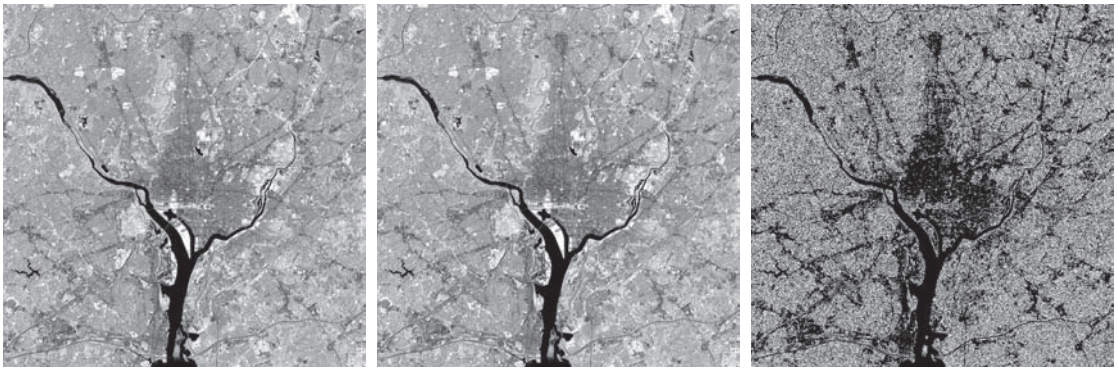
a b c
d e f

FIGURE 2.29 (a) Image of Galaxy Pair NGC 3314 corrupted by additive Gaussian noise. (b)-(f) Result of averaging 5, 10, 20, 50, and 1,00 noisy images, respectively. All images are of size 566×598 pixels, and all were scaled so that their intensities would span the full [0, 255] intensity scale. (Original image courtesy of NASA.)

(2-29), the standard deviation of the noise in Fig. 2.29(b) is less than half ($1/\sqrt{5} = 0.45$) the standard deviation of the noise in Fig. 2.29(a), or $(0.45)(64) \approx 29$ intensity levels. Similarly, the standard deviations of the noise in Figs. 2.29(c) through (f) are 0.32, 0.22, 0.14, and 0.10 of the original, which translates approximately into 20, 14, 9, and 6 intensity levels, respectively. We see in these images a progression of more visible detail as the standard deviation of the noise decreases. The last two images are visually identical for all practical purposes. This is not unexpected, as the difference between the standard deviations of their noise level is only about 3 intensity levels According to the discussion in connection with Fig. 2.5, this difference is below what a human generally is able to detect.

EXAMPLE 2.6: Comparing images using subtraction.

Image subtraction is used routinely for enhancing differences between images. For example, the image in Fig. 2.30(b) was obtained by setting to zero the least-significant bit of every pixel in Fig. 2.30(a). Visually, these images are indistinguishable. However, as Fig. 2.30(c) shows, subtracting one image from



a b c

FIGURE 2.30 (a) Infrared image of the Washington, D.C. area. (b) Image resulting from setting to zero the least significant bit of every pixel in (a). (c) Difference of the two images, scaled to the range [0, 255] for clarity. (Original image courtesy of NASA.)

the other clearly shows their differences. Black (0) values in the difference image indicate locations where there is no difference between the images in Figs. 2.30(a) and (b).

We saw in Fig. 2.23 that detail was lost as the resolution was reduced in the chronometer image shown in Fig. 2.23(a). A vivid indication of image change as a function of resolution can be obtained by displaying the differences between the original image and its various lower-resolution counterparts. Figure 2.31(a) shows the difference between the 930 dpi and 72 dpi images. As you can see, the differences are quite noticeable. The intensity at any point in the difference image is proportional to the magnitude of the numerical difference between the two images at that point. Therefore, we can analyze which areas of the original image are affected the most when resolution is reduced. The next two images in Fig. 2.31 show proportionally less overall intensities, indicating smaller differences between the 930 dpi image and 150 dpi and 300 dpi images, as expected.

a b c

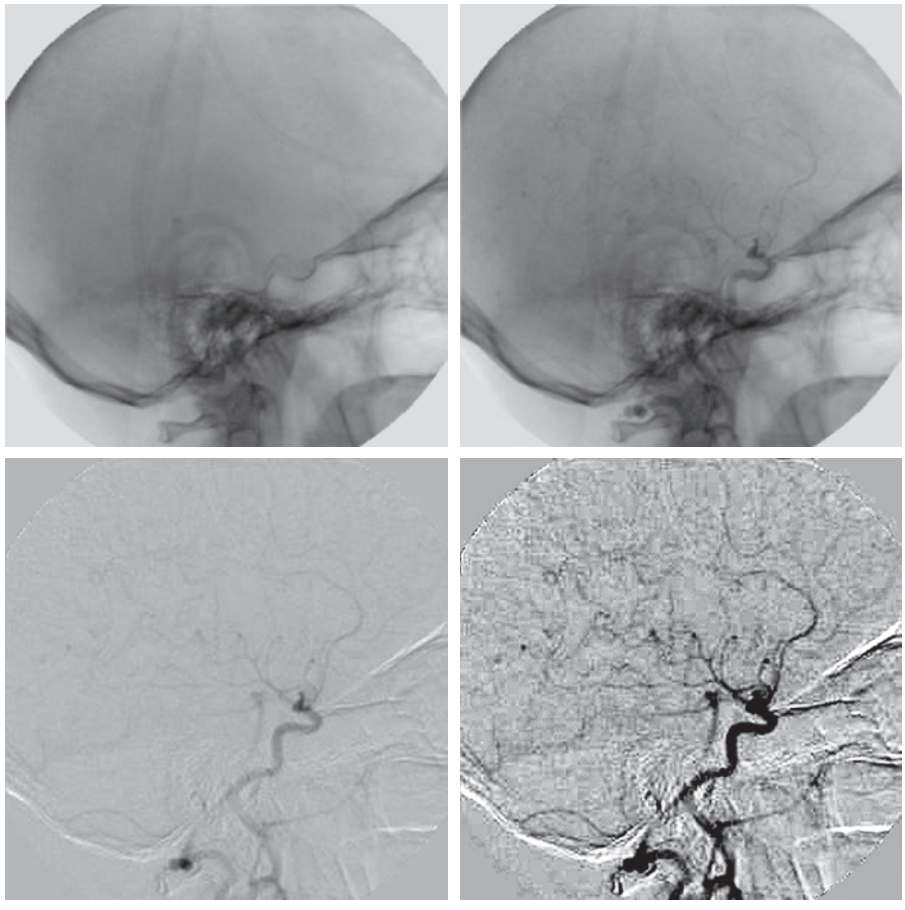
FIGURE 2.31 (a) Difference between the 930 dpi and 72 dpi images in Fig. 2.23. (b) Difference between the 930 dpi and 150 dpi images. (c) Difference between the 930 dpi and 300 dpi images.

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

$$g(x, y) = f(x, y) - h(x, y) \tag{2-30}$$

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

Figure 2.32(a) shows a mask X-ray image of the top of a patient's head prior to injection of an iodine medium into the bloodstream, and Fig. 2.32(b) is a sample of a live image taken after the medium was



a b
c d

FIGURE 2.32 Digital subtraction angiography. (a) Mask image. (b) A live image. (c) Difference between (a) and (b). (d) Enhanced difference image. (Figures (a) and (b) courtesy of the Image Sciences Institute, University Medical Center, Utrecht, The Netherlands.)

injected. Figure 2.32(c) is the difference between (a) and (b). Some fine blood vessel structures are visible in this image. The difference is clear in Fig. 2.32(d), which was obtained by sharpening the image and enhancing its contrast (we will discuss these techniques in the next chapter). Figure 2.32(d) is a "snapshot" of how the medium is propagating through the blood vessels in the subject's brain.

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

An important application of image multiplication (and division) is *shading correction*. Suppose that an imaging sensor produces images that can be modeled as the product of a "perfect image," denoted by $f(x, y)$, times a shading function, $h(x, y)$; that is, $g(x, y) = f(x, y)h(x, y)$. If $h(x, y)$ is known or can be estimated, we can obtain $f(x, y)$ (or an estimate of it) by multiplying the sensed image by the inverse of $h(x, y)$ (i.e., dividing g by h using elementwise division). If access to the imaging system is possible, we can obtain a good approximation to the shading function by imaging a target of constant intensity. When the sensor is not available, we often can estimate the shading pattern directly from a shaded image using the approaches discussed in Sections 3.5 and 9.8. Figure 2.33 shows an example of shading correction using an estimate of the shading pattern. The corrected image is not perfect because of errors in the shading pattern (this is typical), but the result definitely is an improvement over the shaded image in Fig. 2.33 (a). See Section 3.5 for a discussion of how we estimated Fig. 2.33 (b). Another use of image multiplication is in *masking*, also called *region of interest* (ROI), operations. As Fig. 2.34 shows, the process consists of multiplying a given image by a mask image that has 1's in the ROI and 0's elsewhere. There can be more than one ROI in the mask image, and the shape of the ROI can be arbitrary.

A few comments about implementing image arithmetic operations are in order before we leave this section. In practice, most images are displayed using 8 bits (even 24-bit color images consist of three separate 8-bit channels). Thus, we expect image values to be in the range from 0 to 255. When images are saved in a standard image format, such as TIFF or JPEG, conversion to this range is automatic. When image values exceed the allowed range, clipping or scaling becomes necessary. For example, the values in the difference of two 8-bit images can range from a minimum of –255

a b c

FIGURE 2.33 Shading correction. (a) Shaded test pattern. (b) Estimated shading pattern. (c) Product of (a) by the reciprocal of (b). (See Section 3.5 for a discussion of how (b) was estimated.)

A *relation* (or, more precisely, a *binary relation*) on a set A is a collection of ordered pairs of elements from A. That is, a binary relation is a subset of the Cartesian product $A \times A$. A binary relation between *two* sets, A and B, is a subset of $A \times B$.

A *partial order* on a set S is a relation $\mathcal{R}$ on S such that $\mathcal{R}$ is:

(a) *reflexive:* for any $a \in S$, $a\mathcal{R}a$;

(b) *transitive:* for any $a, b, c \in S$, $a\mathcal{R}b$ and $b\mathcal{R}c$ implies that $a\mathcal{R}c$;

(c) *antisymmetric:* for any $a, b \in S$, $a\mathcal{R}b$ and $b\mathcal{R}a$ implies that $a = b$.

where, for example, $a\mathcal{R}b$ reads "a is related to b." This means that a and b are in set $\mathcal{R}$, which itself is a subset of $S \times S$ according to the preceding definition of a relation. A set with a partial order is called a *partially ordered set*.

Let the symbol $\preceq$ denote an ordering relation. An expression of the form

$$a_1 \preceq a_2 \preceq a_3 \preceq \cdots \preceq a_n$$

reads: a_1 precedes a_2 or is the same as a_2, a_2 precedes a_3 or is the same as a_3, and so on. When working with numbers, the symbol $\preceq$ typically is replaced by more traditional symbols. For example, the set of real numbers ordered by the relation "less than or equal to" (denoted by $\leq$) is a partially ordered set (see Problem 2.33). Similarly, the set of natural numbers, paired with the relation "divisible by" (denoted by $\div$), is a partially ordered set.

Of more interest to us later in the book are strict orderings. A *strict ordering* on a set S is a relation $\mathcal{R}$ on S, such that $\mathcal{R}$ is:

(a) *antireflexive:* for any $a \in S$, $\neg a\mathcal{R}a$;

(b) *transitive:* for any $a, b, c \in S$, $a\mathcal{R}b$ and $b\mathcal{R}c$ implies that $a\mathcal{R}c$.

where $\neg a\mathcal{R}a$ means that a is *not related* to a. Let the symbol $\prec$ denote a strict ordering relation. An expression of the form

$$a_1 \prec a_2 \prec a_3 \prec \cdots \prec a_n$$

reads a_1 precedes a_2, a_2 precedes a_3, and so on. A set with a strict ordering is called a *strict-ordered set*.

As an example, consider the set composed of the English alphabet of lowercase letters, $S = \{a, b, c, \cdots, z\}$. Based on the preceding definition, the ordering

$$a \prec b \prec c \prec \cdots \prec z$$

is strict because no member of the set can precede itself (antireflexivity) and, for any three letters in S, if the first precedes the second, and the second precedes the third, then the first precedes the third (transitivity). Similarly, the set of integers paired with the relation "less than ($<$)" is a strict-ordered set.

Logical Operations

Logical operations deal with TRUE (typically denoted by 1) and FALSE (typically denoted by 0) variables and expressions. For our purposes, this means binary images

composed of *foreground* (1-valued) pixels, and a *background* composed of 0-valued pixels.

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. In this representation, we work with single images using all the set operations defined in the previous section. For example, given a binary image with two 1-valued regions, R_1 and R_2, we can determine if the regions overlap (i.e., if they have at least one pair of coordinates in common) by performing the set intersection operation $R_1 \cap R_2$ (see Fig. 2.35). 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 shows for two logical variables a and b. The logical AND operation (also denoted $\wedge$) yields a 1 (TRUE) only when both *a and b* are 1. Otherwise, it yields 0 (FALSE). Similarly, the logical OR ($\vee$) yields 1 when both *a or b* or *both* are 1, and 0 otherwise. The NOT ($\sim$) operator is self explanatory. When applied to two binary images, AND and OR operate on pairs of corresponding pixels between the images. That is, they are elementwise operators (see the definition of elementwise operators given earlier in this chapter) in this context. The operators AND, OR, and NOT are *functionally complete*, in the sense that they can be used as the basis for constructing any other logical operator.

Figure 2.37 illustrates the logical operations defined in Table 2.2 using the second approach discussed above. The NOT of binary image B_1 is an array obtained by changing all 1-valued pixels to 0, and vice versa. The AND of B_1 and B_2 contains a 1 at all spatial locations where the corresponding elements of B_1 and B_2 are 1; the operation yields 0's elsewhere. Similarly, the OR of these two images is an array that contains a 1 in locations where the corresponding elements of B_1, *or* B_2, *or both*, are 1. The array contains 0's elsewhere. The result in the fourth row of Fig. 2.37 corresponds to the set of 1-valued pixels in B_1 but not in B_2. The last row in the figure is the XOR (exclusive OR) operation, which yields 1 in the locations where the corresponding elements of B_1 *or* B_2, (but *not both*) are 1. Note that the logical

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

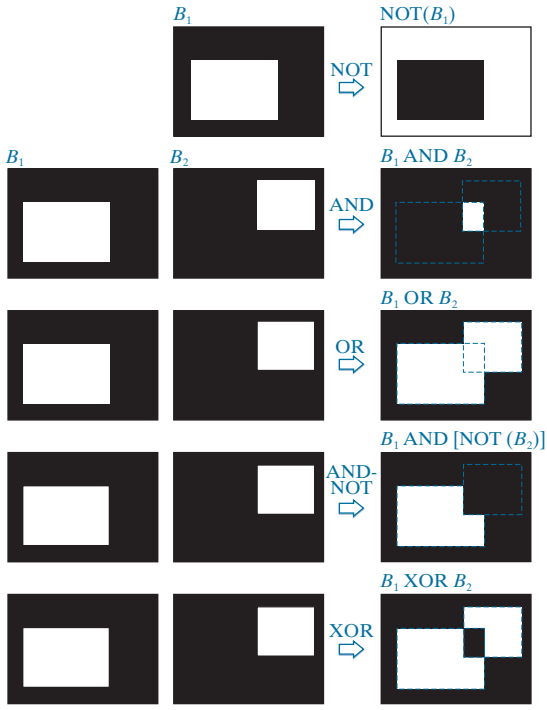


FIGURE 2.37 Illustration of logical operations involving foreground (white) pixels. Black represents binary 0's and white binary 1's. The dashed lines are shown for reference only. They are not part of the result.

expressions in the last two rows of Fig. 2.37 were constructed using operators from Table 2.2; these are examples of the functionally complete nature of these operators.

We can arrive at the same results in Fig. 2.37 using the first approach discussed above. To do this, we begin by labeling the individual 1-valued regions in each of the two images (in this case there is only one such region in each image). Let A and B denote the *set of coordinates* of all the 1-valued pixels in images B_1 and B_2, respectively. Then we form a *single* array by ORing the two images, while keeping the labels A and B. The result would look like the array B_1 OR B_2 in Fig. 2.37, but with the two white regions labeled A and B. In other words, the resulting array would look like a Venn diagram. With reference to the Venn diagrams and set operations defined in the previous section, we obtain the results in the rightmost column of Fig. 2.37 using set operations as follows: $A^c = \text{NOT}(B_1)$, $A \cap B = B_1$ AND B_2, $A \cup B = B_1$ OR B_2, and similarly for the other results in Fig. 2.37. We will make extensive use in Chapter 9 of the concepts developed in this section.

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

The transformation of coordinates may be expressed as

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

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 3×3 matrix in the following general form:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{A} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{2-45}$$

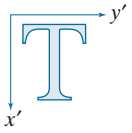
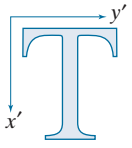
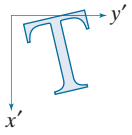
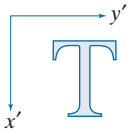
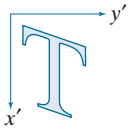
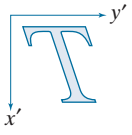
This transformation can *scale*, *rotate*, *translate*, or *sheer* an image, depending on the values chosen for the elements of matrix **A**. Table 2.3 shows the matrix values used to implement these transformations. A significant advantage of being able to perform all transformations using the unified representation in Eq. (2-45) is that it provides the framework for concatenating a sequence of operations. For example, if we want to resize an image, rotate it, and move the result to some location, we simply form a 3×3 matrix equal to the product of the scaling, rotation, and translation matrices from Table 2.3 (see Problems 2.36 and 2.37).

The preceding transformation moves the coordinates of pixels in an image to new locations. To complete the process, we have to assign intensity values to those locations. This task is accomplished using *intensity interpolation*. We already discussed this topic in Section 2.4. We began that discussion with an example of zooming an image and discussed the issue of intensity assignment to new pixel locations. Zooming is simply scaling, as detailed in the second row of Table 2.3, and an analysis similar to the one we developed for zooming is applicable to the problem of assigning intensity values to the relocated pixels resulting from the other transformations in Table 2.3. As in Section 2.4, we consider nearest neighbor, bilinear, and bicubic interpolation techniques when working with these transformations.

We can use Eq. (2-45) in two basic ways. The first, is a *forward mapping*, which consists of scanning the pixels of the input image and, at each location (x, y), com-

puting the spatial location (x', y') of the corresponding pixel in the output image using Eq. (2-45) directly. A problem with the forward mapping approach is that two or more pixels in the input image can be transformed to the same location in the output image, raising the question of how to combine multiple output values into a single output pixel value. In addition, it is possible that some output locations may not be assigned a pixel at all. The second approach, called *inverse mapping*, scans the output pixel locations and, at each location (x', y'), computes the corresponding location in the input image using $(x, y) = A^{-1}(x', y')$. It then interpolates (using one of the techniques discussed in Section 2.4) among the nearest input pixels to determine the intensity of the output pixel value. Inverse mappings are more efficient to implement than forward mappings, and are used in numerous commercial implementations of spatial transformations (for example, MATLAB uses this approach).

TABLE 2.3 Affine transformations based on Eq. (2-45).

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

EXAMPLE 2.9: Image rotation and intensity interpolation.

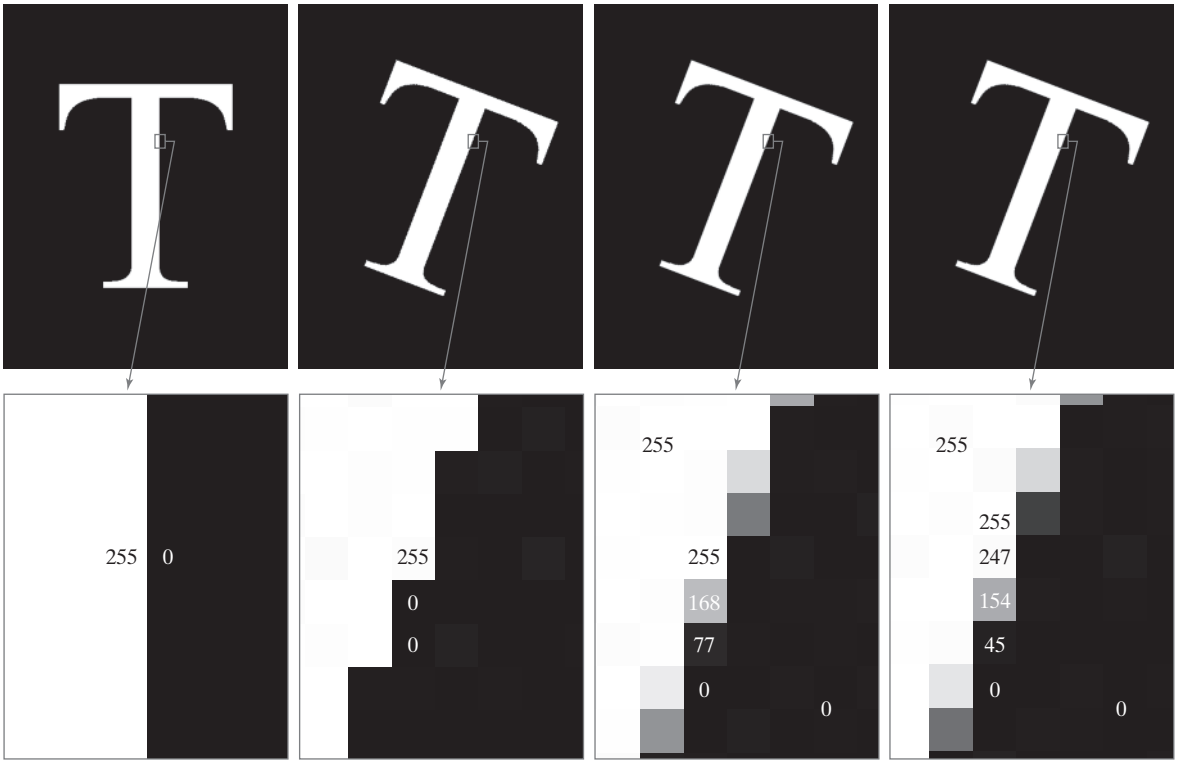
The objective of this example is to illustrate image rotation using an affine transform. Figure 2.40(a) shows a simple image and Figs. 2.40(b)–(d) are the results (using inverse mapping) of rotating the original image by –21° (in Table 2.3, clockwise angles of rotation are negative). Intensity assignments were computed using nearest neighbor, bilinear, and bicubic interpolation, respectively. A key issue in image rotation is the preservation of straight-line features. As you can see in the enlarged edge sections in Figs. 2.40(f) through (h), nearest neighbor interpolation produced the most jagged edges and, as in Section 2.4, bilinear interpolation yielded significantly improved results. As before, using bicubic interpolation produced slightly better results. In fact, if you compare the progression of enlarged detail in Figs. 2.40(f) to (h), you can see that the transition from white (255) to black (0) is smoother in the last figure because the edge region has more values, and the distribution of those values is better balanced. Although the small intensity differences resulting from bilinear and bicubic interpolation are not always noticeable in human visual analysis, they can be important in processing image data, such as in automated edge following in rotated images.

The size of the spatial rectangle needed to contain a rotated image is larger than the rectangle of the original image, as Figs. 2.41(a) and (b) illustrate. We have two options for dealing with this: (1) we can crop the rotated image so that its size is equal to the size of the original image, as in Fig. 2.41(c), or we can keep the larger image containing the full rotated original, an Fig. 2.41(d). We used the first option in Fig. 2.40 because the rotation did not cause the object of interest to lie outside the bounds of the original rectangle. The areas in the rotated image that do not contain image data must be filled with some value, 0 (black) being the most common. Note that counterclockwise angles of rotation are considered positive. This is a result of the way in which our image coordinate system is set up (see Fig. 2.19), and the way in which rotation is defined in Table 2.3.

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. Unlike the discussion in the previous section where transformation functions are known, the geometric transformation needed to produce the output, 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. In either case, combining the images or performing quantitative analysis and comparisons between them requires compensating for geometric distortions caused by differences in viewing angle, distance, orientation, sensor resolution, shifts in object location, and other factors.



a b c d
e f g h

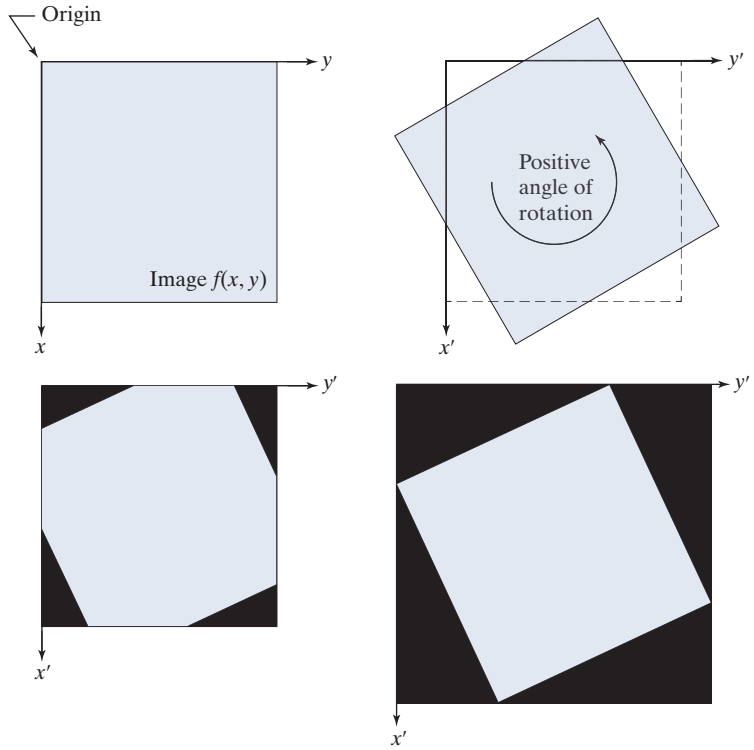
FIGURE 2.40 (a) A 541×421 image of the letter T. (b) Image rotated –21° using nearest-neighbor interpolation for intensity assignments. (c) Image rotated –21° using bilinear interpolation. (d) Image rotated –21° using bicubic interpolation. (e)-(h) Zoomed sections (each square is one pixel, and the numbers shown are intensity values).

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

$$x = c_1 v + c_2 w + c_3 v w + c_4 \tag{2-46}$$

and



a b
c d

FIGURE 2.41
(a) A digital image.
(b) Rotated image (note the counterclockwise direction for a positive angle of rotation).
(c) Rotated image cropped to fit the same area as the original image.
(d) Image enlarged to accommodate the entire rotated image.

$$y = c_5 v + c_6 w + c_7 v w + c_8 \tag{2-47}$$

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_1 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.

In situations where four tie points are insufficient to obtain satisfactory registration, an approach used frequently is to select a larger number of tie points and then treat the quadrilaterals formed by groups of four tie points as subimages. The subimages are processed as above, with all the pixels within a quadrilateral being transformed using the coefficients determined from the tie points corresponding to that quadrilateral. Then we move to another set of four tie points and repeat the

6.1 COLOR FUNDAMENTALS

Although the process employed by the human brain in perceiving and interpreting color is a physiopsychological phenomenon that is not fully understood, the physical nature of color can be expressed on a formal basis supported by experimental and theoretical results.

In 1666, Sir Isaac Newton discovered that when a beam of sunlight passes through a glass prism, the emerging 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. 6.1 shows, the color spectrum may be divided into six broad regions: violet, blue, green, yellow, orange, and red. When viewed in full color (see Fig. 6.2), no color in the spectrum ends abruptly; rather, each color blends smoothly into the next.

Basically, the colors that humans and some other animals perceive in an object are determined by the nature of the light reflected from the object. As illustrated in Fig. 6.2, visible light is composed of a relatively narrow band of frequencies in the electromagnetic spectrum. A body that reflects light that is balanced in all visible wavelengths appears white to the observer. However, a body that favors reflectance in a limited range of the visible spectrum exhibits some shades of color. For example, green objects reflect light with wavelengths primarily in the 500 to 570 nm range, while absorbing most of the energy at other wavelengths.

Characterization of light is central to the science of color. If the light is *achromatic* (void of color), its only attribute is its *intensity*, or amount. Achromatic light is what you see on movie films made before the 1930s. As defined in Chapter 2, and used numerous times since, the term *gray (*or *intensity) level* refers to a scalar measure of intensity that ranges from black, to grays, and finally to white.

Chromatic light spans the electromagnetic spectrum from approximately 400 to 700 nm. Three basic quantities used to describe the quality of a chromatic light source are: radiance, luminance, and brightness. *Radiance* is the total amount of energy that flows from the light source, and it is usually measured in watts (W). *Luminance*, measured in lumens (lm), is a measure of the amount of energy that an observer *perceives* from a light source. For example, light emitted from a source operating in the far infrared region of the spectrum could have significant energy (radiance), but an observer would hardly perceive it; its luminance would be almost zero. Finally, *brightness* is a subjective descriptor that is practically impossible to measure. It embodies the achromatic notion of intensity, and is one of the key factors in describing color sensation.

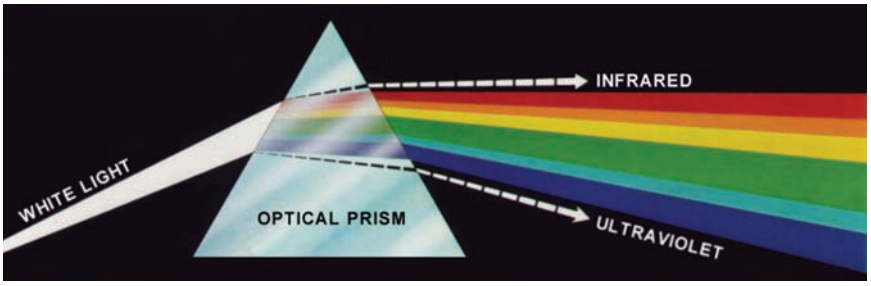


FIGURE 6.1 Color spectrum seen by passing white light through a prism. (Courtesy of the General Electric Co., Lighting Division.)

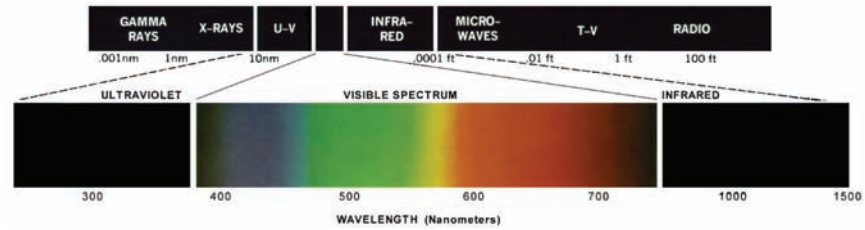


FIGURE 6.2
Wavelengths comprising the visible range of the electromagnetic spectrum. (Courtesy of the General Electric Co., Lighting Division.)

As noted in Section 2.1, cones are the sensors in the eye responsible for color vision. Detailed experimental evidence has established that the 6 to 7 million cones in the human eye can be divided into three principal sensing categories, corresponding roughly to red, green, and blue. Approximately 65% of all cones are sensitive to red light, 33% are sensitive to green light, and only about 2% are sensitive to blue. However, the blue cones are the most sensitive. Figure 6.3 shows average experimental curves detailing the absorption of light by the red, green, and blue cones in the eye. Because of these absorption characteristics, the human eye sees colors as variable combinations of the so-called *primary colors*: *red* (R), *green* (G), and *blue* (B).

For the purpose of standardization, the CIE (Commission Internationale de l'Eclairage—the International Commission on Illumination) designated in 1931 the following specific wavelength values to the three primary colors: blue = 435.8 nm, green = 546.1 nm, and red = 700 nm. This standard was set before results such as those in Fig. 6.3 became available in 1965. Thus, the CIE standards correspond only approximately with experimental data. It is important to keep in mind that defining three specific primary color wavelengths for the purpose of standardization does

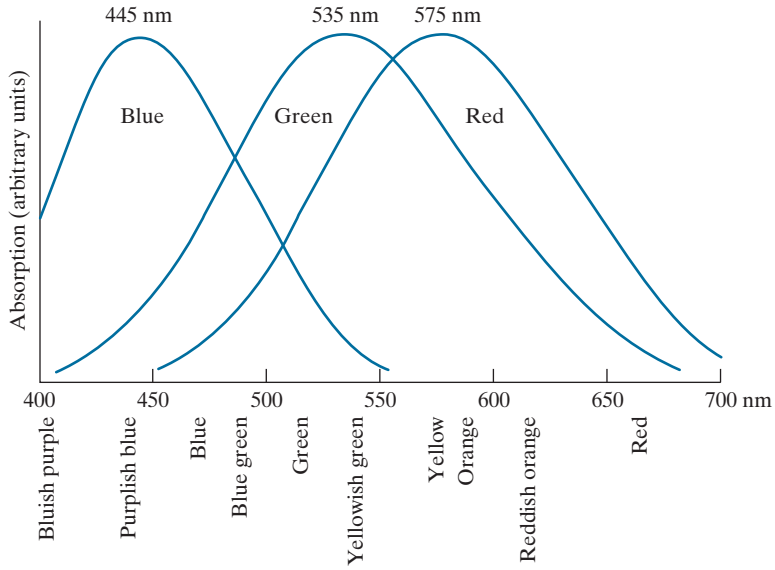


FIGURE 6.3
Absorption of light by the red, green, and blue cones in the human eye as a function of wavelength.

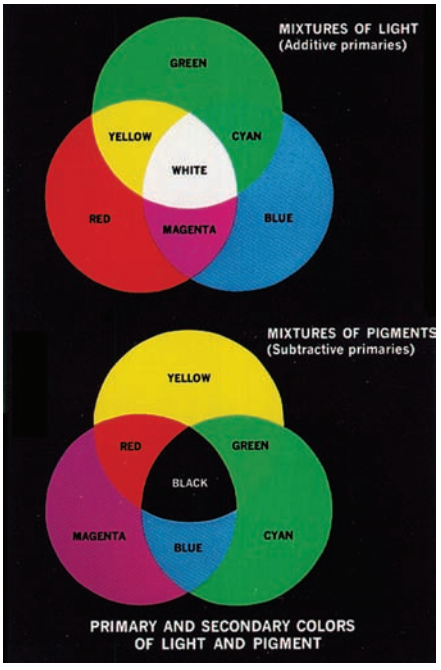
not mean that these three fixed RGB components acting alone can generate all spectrum colors. Use of the word *primary* has been widely misinterpreted to mean that the three standard primaries, when mixed in various intensity proportions, can produce all visible colors. As you will see shortly, this interpretation is not correct unless the wavelength also is allowed to vary, in which case we would no longer have three fixed primary colors.

The primary colors can be added together to produce the *secondary* colors of light—*magenta* (red plus blue), *cyan* (green plus blue), and *yellow* (red plus green). Mixing the three primaries, or a secondary with its opposite primary color, in the right intensities produces white light. This result is illustrated in Fig. 6.4(a), which shows also the three primary colors and their combinations to produce the secondary colors of light.

Differentiating between the primary colors of light and the primary colors of pigments or colorants is important. In the latter, a primary color is defined as one that subtracts or absorbs a primary color of light, and reflects or transmits the other two. Therefore, the primary colors of pigments are magenta, cyan, and yellow, and the secondary colors are red, green, and blue. These colors are shown in Fig. 6.4(b). A proper combination of the three pigment primaries, or a secondary with its opposite primary, produces black.

In practice, pigments seldom are pure. This results in a muddy brown instead of black when primaries, or primaries and secondaries, are combined. We will discuss this issue in Section 6.2

Color television reception is an example of the additive nature of light colors. The interior of CRT (cathode ray tube) color TV screens used well into the 1990s is composed of a large array of triangular dot patterns of electron-sensitive phosphor. When excited, each dot in a triad produces light in one of the primary colors. The



a
b

FIGURE 6.4
Primary and secondary colors of light and pigments. (Courtesy of the General Electric Co., Lighting Division.)

intensity of the red-emitting phosphor dots is modulated by an electron gun inside the tube, which generates pulses corresponding to the "red energy" seen by the TV camera. The green and blue phosphor dots in each triad are modulated in the same manner. The effect, viewed on the television receiver, is that the three primary colors from each phosphor triad are received and "added" together by the color-sensitive cones in the eye and perceived as a full-color image. Thirty successive image changes per second in all three colors complete the illusion of a continuous image display on the screen.

CRT displays started being replaced in the late 1990s by flat-panel digital technologies, such as liquid crystal displays (LCDs) and plasma devices. Although they are fundamentally different from CRTs, these and similar technologies use the same principle in the sense that they all require three subpixels (red, green, and blue) to generate a single color pixel. LCDs use properties of polarized light to block or pass light through the LCD screen and, in the case of active matrix display technologies, thin film transistors (TFTs) are used to provide the proper signals to address each pixel on the screen. Light filters are used to produce the three primary colors of light at each pixel triad location. In plasma units, pixels are tiny gas cells coated with phosphor to produce one of the three primary colors. The individual cells are addressed in a manner analogous to LCDs. This individual pixel triad coordinate addressing capability is the foundation of digital displays.

The characteristics generally used to distinguish one color from another are brightness, hue, and saturation. As indicated earlier in this section, brightness embodies the achromatic notion of intensity. *Hue* is an attribute associated with the dominant wavelength in a mixture of light waves. Hue represents dominant color as perceived by an observer. Thus, when we call an object red, orange, or yellow, we are referring to its hue. *Saturation* refers to the relative purity or the amount of white light mixed with a hue. The pure spectrum colors are fully saturated. Colors such as pink (red and white) and lavender (violet and white) are less saturated, with the degree of saturation being inversely proportional to the amount of white light added.

Hue and saturation taken together are called *chromaticity* and, therefore, a color may be characterized by its brightness and chromaticity. The amounts of red, green, and blue needed to form any particular color are called the *tristimulus* values, and are denoted, X, Y, and Z, respectively. A color is then specified by its *trichromatic coefficients*, defined as

$$x = \frac{X}{X + Y + Z} \tag{6-1}$$

$$y = \frac{Y}{X + Y + Z} \tag{6-2}$$

and

$$z = \frac{Z}{X + Y + Z} \tag{6-3}$$

Our use of x, y, and z in this context follows convention. These should not be confused with our use of (x, y) throughout the book to denote spatial coordinates.

We see from these equations that

$$x + y + z = 1 \tag{6-4}$$

For any wavelength of light in the visible spectrum, the tristimulus values needed to produce the color corresponding to that wavelength can be obtained directly from curves or tables that have been compiled from extensive experimental results (Poynton [1996, 2012]).

Another approach for specifying colors is to use the CIE *chromaticity diagram* (see Fig. 6.5), which shows color composition as a function of x (red) and y (green). For any value of x and y, the corresponding value of z (blue) is obtained from Eq. (6-4) by noting that $z = 1 - (x + y)$. The point marked green in Fig. 6.5, for example, has approximately 62% green and 25% red content. It follows from Eq. (6-4) that the composition of blue is approximately 13%.

The positions of the various spectrum colors—from violet at 380 nm to red at 780 nm—are indicated around the boundary of the tongue-shaped chromaticity diagram. These are the pure colors shown in the spectrum of Fig. 6.2. Any point not actually on the boundary, but within the diagram, represents some mixture of the pure spectrum colors. The *point of equal energy* shown in Fig. 6.5 corresponds to equal fractions of the three primary colors; it represents the CIE standard for white light. Any point located on the boundary of the chromaticity chart is fully saturated. As a point leaves the boundary and approaches the point of equal energy, more white light is added to the color, and it becomes less saturated. The saturation at the point of equal energy is zero.

The chromaticity diagram is useful for color mixing because a straight-line segment joining any two points in the diagram defines all the different color variations that can be obtained by combining these two colors additively. Consider, for example, a straight line drawn from the red to the green points shown in Fig. 6.5. If there is more red than green light, the exact point representing the new color will be on the line segment, but it will be closer to the red point than to the green point. Similarly, a line drawn from the point of equal energy to any point on the boundary of the chart will define all the shades of that particular spectrum color.

Extending this procedure to three colors is straightforward. To determine the range of colors that can be obtained from any three given colors in the chromaticity diagram, we simply draw connecting lines to each of the three color points. The result is a triangle, and any color inside the triangle, or on its boundary, can be produced by various combinations of the three vertex colors. A triangle with vertices at any three fixed colors cannot enclose the entire color region in Fig. 6.5. This observation supports graphically the remark made earlier that not all colors can be obtained with three single, *fixed* primaries, because three colors form a triangle.

The triangle in Fig. 6.6 shows a representative range of colors (called the *color gamut*) produced by RGB monitors. The shaded region inside the triangle illustrates the color gamut of today's high-quality color printing devices. The boundary of the color printing gamut is irregular because color printing is a combination of additive and subtractive color mixing, a process that is much more difficult to control than

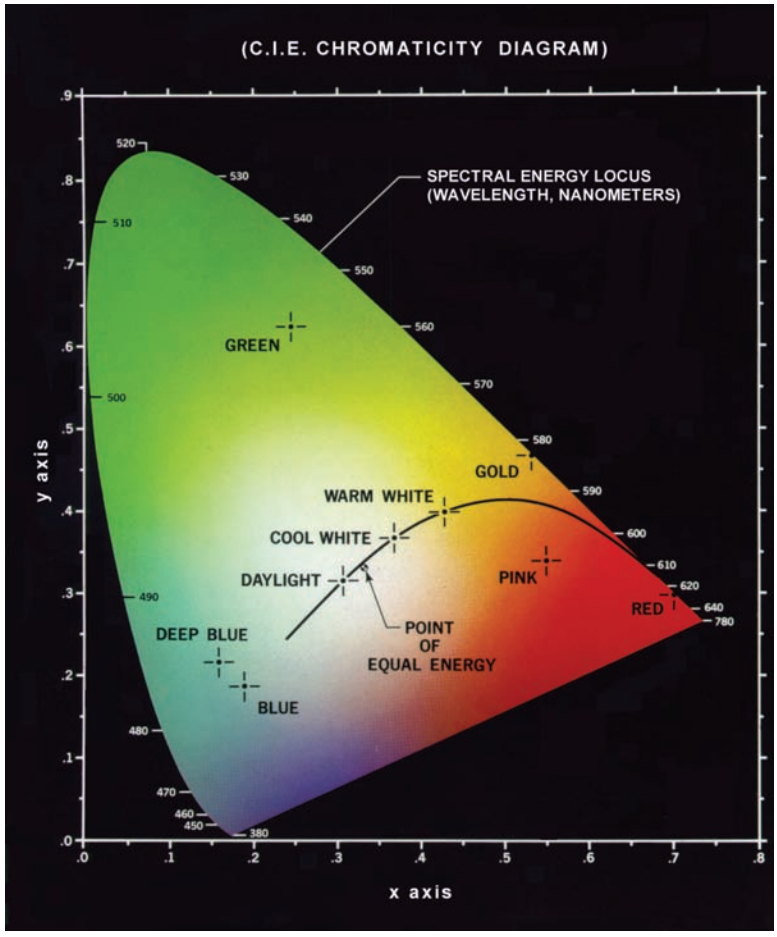


FIGURE 6.5
The CIE chromaticity diagram. (Courtesy of the General Electric Co., Lighting Division.)

that of displaying colors on a monitor, which is based on the addition of three highly controllable light primaries.

6.2 COLOR MODELS

The purpose of a *color model* (also called a *color space* or *color system*) is to facilitate the specification of colors in some standard way. In essence, a color model is a specification of (1) a coordinate system, and (2) a subspace within that system, such that each color in the model is represented by a single point contained in that subspace.

Most color models in use today are oriented either toward hardware (such as for color monitors and printers) or toward applications, where color manipulation is a goal (the creation of color graphics for animation is an example of the latter). In terms of digital image processing, the hardware-oriented models most commonly used in practice are the RGB (red, green, blue) model for color monitors and a

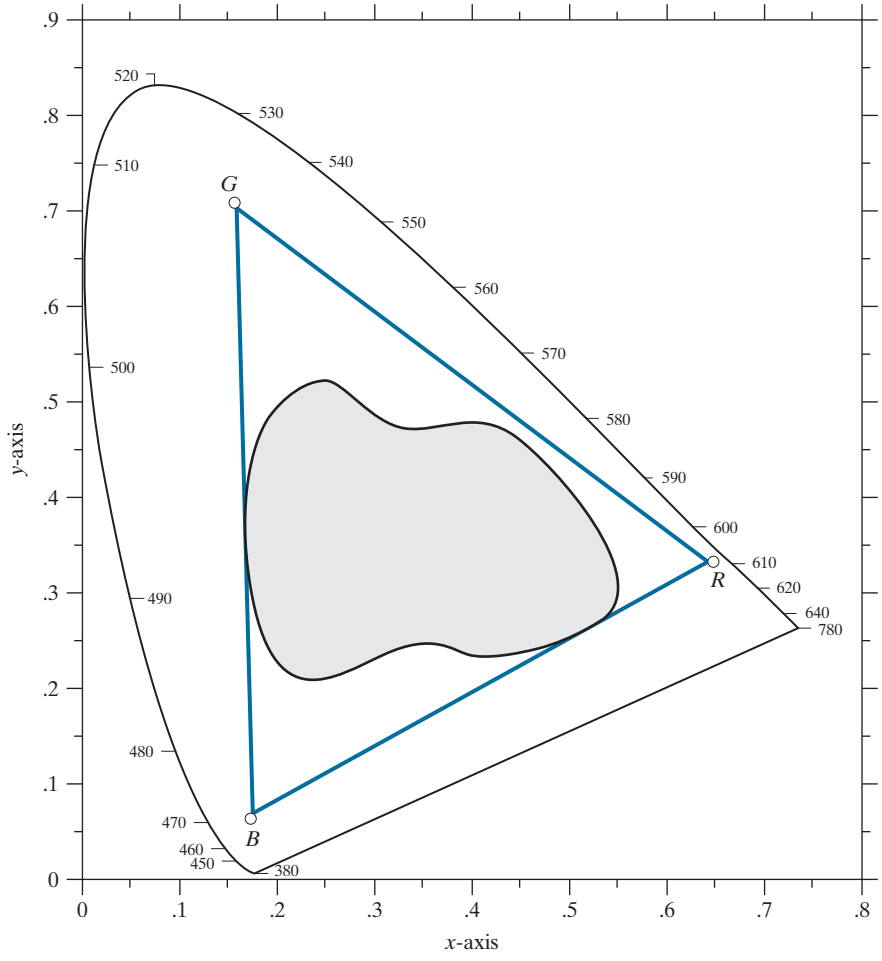


FIGURE 6.6
Illustrative color gamut of color monitors (triangle) and color printing devices (shaded region).

broad class of color video cameras; the CMY (cyan, magenta, yellow) and CMYK (cyan, magenta, yellow, black) models for color printing; and the HSI (hue, saturation, intensity) model, which corresponds closely with the way humans describe and interpret color. The HSI model also has the advantage that it decouples the color and gray-scale information in an image, making it suitable for many of the gray-scale techniques developed in this book. There are numerous color models in use today. This is a reflection of the fact that color science is a broad field that encompasses many areas of application. It is tempting to dwell on some of these models here, simply because they are interesting and useful. However, keeping to the task at hand, we focus attention on a few models that are representative of those used in image processing. Having mastered the material in this chapter, you will have no difficulty in understanding additional color models in use today.

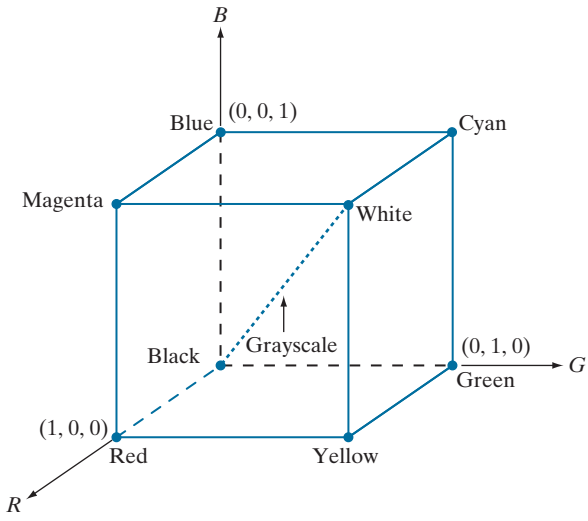


FIGURE 6.7 Schematic of the RGB color cube. Points along the main diagonal have gray values, from black at the origin to white at point (1, 1, 1).

THE RGB COLOR MODEL

In the RGB model, each color appears in its primary spectral components of red, green, and blue. This model is based on a Cartesian coordinate system. The color subspace of interest is the cube shown in Fig. 6.7, in which RGB primary values are at three corners; the secondary colors cyan, magenta, and yellow are at three other corners; black is at the origin; and white is at the corner farthest from the origin. In this model, the grayscale (points of equal RGB values) extends from black to white along the line joining these two points. The different colors in this model are points on or inside the cube, and are defined by vectors extending from the origin. For convenience, the assumption is that all color values have been normalized so the cube in Fig. 6.7 is the unit cube. That is, all values of R, G, and B in this representation are assumed to be in the range [0, 1]. Note that the RGB primaries can be interpreted as unit vectors emanating from the origin of the cube.

Images represented in the RGB color model consist of three component images, one for each primary color. When fed into an RGB monitor, these three images combine on the screen to produce a composite color image, as explained in Section 6.1. The number of bits used to represent each pixel in RGB space is called the *pixel depth*. Consider an RGB image in which each of the red, green, and blue images is an 8-bit image. Under these conditions, each RGB *color pixel* [that is, a triplet of values (R, G, B)] has a depth of 24 bits (3 image planes times the number of bits per plane). The term *full-color* image is used often to denote a 24-bit RGB color image. The total number of possible colors in a 24-bit RGB image is $(2^8)^3 = 16,777,216$. Figure 6.8 shows the 24-bit RGB color cube corresponding to the diagram in Fig. 6.7. Note also that for digital images, the range of values in the cube are scaled to the

FIGURE 6.8
A 24-bit RGB color cube.

numbers representable by the number bits in the images. If, as above, the primary images are 8-bit images, the limits of the cube along each axis becomes [0, 255]. Then, for example, white would be at point [255, 255, 255] in the cube.

EXAMPLE 6.1: Generating a cross-section of the RGB color cube and its thee hidden planes.

The cube in Fig. 6.8 is a solid, composed of the $(2^8)^3$ colors mentioned in the preceding paragraph. A useful way to view these colors is to generate color planes (faces or cross sections of the cube). This is done by fixing one of the three colors and allowing the other two to vary. For instance, a cross-sectional plane through the center of the cube and parallel to the GB-plane in Fig. 6.8 is the plane (127, G, B) for $G, B = 0, 1, 2, \ldots, 255$. Figure 6.9(a) shows that an image of this cross-sectional plane is generated by feeding the three individual component images into a color monitor. In the component images, 0 represents black and 255 represents white. Observe that each component image into the monitor is a grayscale image. The monitor does the job of combining the intensities of these images to generate an RGB image. Figure 6.9(b) shows the three hidden surface planes of the cube in Fig. 6.8, generated in a similar manner.

Acquiring a color image is the process shown in Fig. 6.9(a) in reverse. A color image can be acquired by using three filters, sensitive to red, green, and blue, respectively. When we view a color scene with a monochrome camera equipped with one of these filters, the result is a monochrome image whose intensity is proportional to the response of that filter. Repeating this process with each filter produces three monochrome images that are the RGB component images of the color scene. In practice, RGB color image sensors usually integrate this process into a single device. Clearly, displaying these three RGB component images as in Fig. 6.9(a) would yield an RGB color rendition of the original color scene.

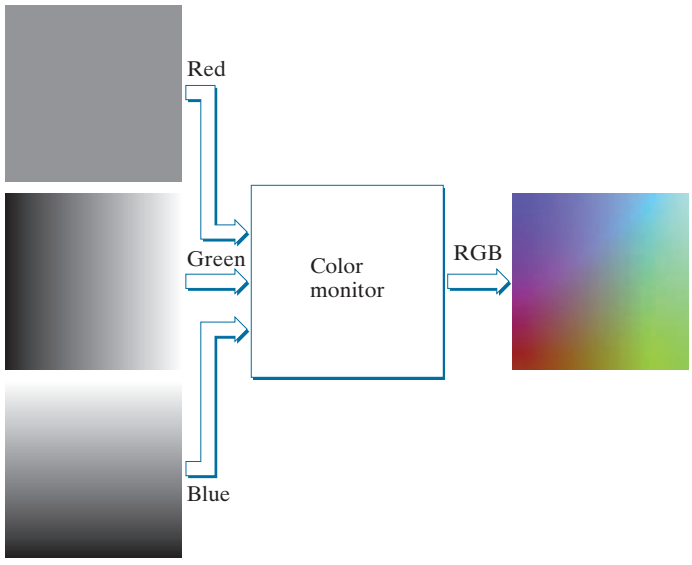
THE CMY AND CMYK COLOR MODELS

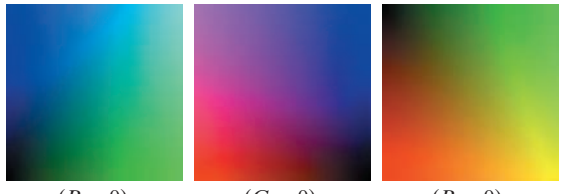
As indicated in Section 6.1, cyan, magenta, and yellow are the secondary colors of light or, alternatively, they are the primary colors of pigments. For example, when a surface coated with cyan pigment is illuminated with white light, no red light is reflected from the surface. That is, cyan subtracts red light from reflected white light, which itself is composed of equal amounts of red, green, and blue light.

Most devices that deposit colored pigments on paper, such as color printers and copiers, require CMY data input or perform an RGB to CMY conversion internally. This conversion is performed using the simple operation

a
b

FIGURE 6.9
(a) Generating the RGB image of the cross-sectional color plane (127, G, B). (b) The three hidden surface planes in the color cube of Fig. 6.8.





$(R = 0)$ $(G = 0)$ $(B = 0)$

Equation (6-5), as well as all other equations in this section, are applied on a pixel-by-pixel basis.

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{6-5}$$

where the assumption is that all RGB color values have been normalized to the range [0, 1]. Equation (6-5) demonstrates that light reflected from a surface coated with pure cyan does not contain red (that is, $C = 1 - R$ in the equation). Similarly, pure magenta does not reflect green, and pure yellow does not reflect blue. Equation (6-5) also reveals that RGB values can be obtained easily from a set of CMY values by subtracting the individual CMY values from 1.

According to Fig. 6.4, equal amounts of the pigment primaries, cyan, magenta, and yellow, should produce black. In practice, because C, M, and Y inks seldom are pure colors, combining these colors for printing black produces instead a muddy-looking brown. So, in order to produce true black (which is the predominant color in printing), a fourth color, *black*, denoted by K, is added, giving rise to the CMYK color model. The black is added in just the proportions needed to produce true black. Thus,

when publishers talk about "four-color printing," they are referring to the three CMY colors, plus a portion of black.

The conversion from CMY to CMYK begins by letting

$$K = \min(C, M, Y) \tag{6-6}$$

If $K = 1$, then we have pure black, with no color contributions, from which it follows that

$$C = 0 \tag{6-7}$$

$$M = 0 \tag{6-8}$$

$$Y = 0 \tag{6-9}$$

The C, M, and Y on the right side of Eqs. (6-6)-(6-12) are in the CMY color system. The C, M, and Y on the left of Eqs. (6-7)-(6-12) are in the CMYK system.

Otherwise,

$$C = (C - K)/(1 - K) \tag{6-10}$$

$$M = (M - K)/(1 - K) \tag{6-11}$$

$$Y = (Y - K)/(1 - K) \tag{6-12}$$

where all values are assumed to be in the range [0, 1]. The conversions from CMYK back to CMY are:

The C, M, Y, and K on the right side of Eqs. (6-13)-(6-15) are in the CMYK color system. The C, M, and Y on the left of these equations are in the CMY system.

$$C = C * (1 - K) + K \tag{6-13}$$

$$M = M * (1 - K) + K \tag{6-14}$$

$$Y = Y * (1 - Y) + K \tag{6-15}$$

As noted at the beginning of this section, all operations in the preceding equations are performed on a pixel-by-pixel basis. Because we can use Eq. (6-5) to convert both ways between CMY and RGB, we can use that equation as a "bridge" to convert between RGB and CMYK, and vice versa.

It is important to keep in mind that all the conversions just presented to go between RGB, CMY, and CMYK are based on the preceding relationships as a group. There are many other ways to convert between these color models, so you cannot mix approaches and expect to get meaningful results. Also, colors seen on monitors generally appear much different when printed, unless these devices are calibrated (see the discussion of a device-independent color model later in this section). The same holds true in general for colors converted from one model to another. However, our interest in this chapter is not on color fidelity; rather, we are interested in using the properties of color models to facilitate image processing tasks, such as region detection.

THE HSI COLOR MODEL

As we have seen, creating colors in the RGB, CMY, and CMYK models, and changing from one model to the other, is straightforward. These color systems are ideally suited for hardware implementations. In addition, the RGB system matches nicely with the fact that the human eye is strongly perceptive to red, green, and blue primaries. Unfortunately, the RGB, CMY, and other similar color models are not well suited for describing colors in terms that are practical for human interpretation. For example, one does not refer to the color of an automobile by giving the percentage of each of the primaries composing its color. Furthermore, we do not think of color images as being composed of three primary images that combine to form a single image.

When humans view a color object, we describe it by its hue, saturation, and brightness. Recall from the discussion in Section 6.1 that hue is a color attribute that describes a pure color (pure yellow, orange, or red), whereas saturation gives a measure of the degree to which a pure color is diluted by white light. Brightness is a subjective descriptor that is practically impossible to measure. It embodies the achromatic notion of *intensity* and is one of the key factors in describing color sensation. We do know that intensity (gray level) is a most useful descriptor of achromatic images. This quantity definitely is measurable and easily interpretable. The model we are about to present, called the *HSI* (hue, saturation, intensity) *color model*, decouples the intensity component from the color-carrying information (hue and saturation) in a color image. As a result, the HSI model is a useful tool for developing image processing algorithms based on color descriptions that are natural and intuitive to humans, who, after all, are the developers and users of these algorithms. We can summarize by saying that RGB is ideal for image color generation (as in image capture by a color camera or image display on a monitor screen), but its use for color description is much more limited. The material that follows provides an effective way to do this.

We know from Example 6.1 that an RGB color image is composed three grayscale intensity images (representing red, green, and blue), so it should come as no surprise that we can to extract intensity from an RGB image. This becomes clear if we take the color cube from Fig. 6.7 and stand it on the black, (0, 0, 0), vertex, with the white, (1, 1, 1), vertex directly above it [see Fig. 6.10(a)]. As noted in our discussion of Fig. 6.7, the intensity (gray) scale is along the line joining these two vertices. In Figs. 6.10(a) and (b), the line (intensity axis) joining the black and white vertices is vertical. Thus, if we wanted to determine the intensity component of any color point in Fig. 6.10, we would simply define a plane that contains the color point and, at the same time, is perpendicular to the intensity axis. The intersection of the plane with the intensity axis would give us a point with intensity value in the range $[0, 1]$. A little thought would reveal that the saturation (purity) of a color increases as a function of distance from the intensity axis. In fact, the saturation of points on the intensity axis is zero, as evidenced by the fact that all points along this axis are gray.

Hue can be determined from an RGB value also. To see how, consider Fig. 6.10(b), which shows a plane defined by three points (black, white, and cyan). The fact that

a b

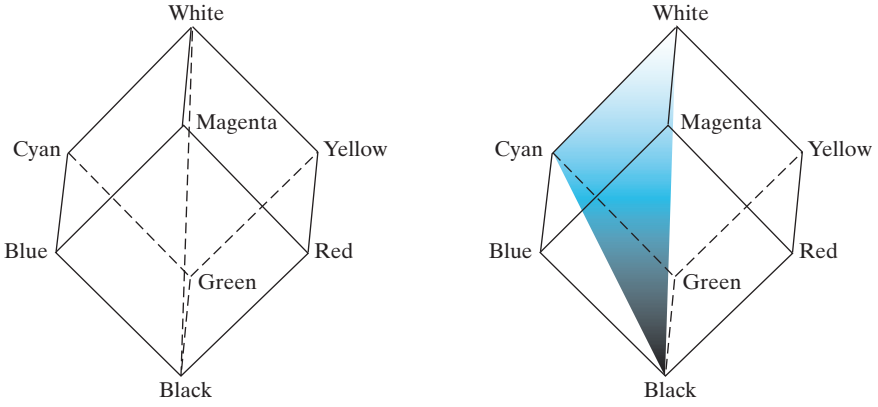
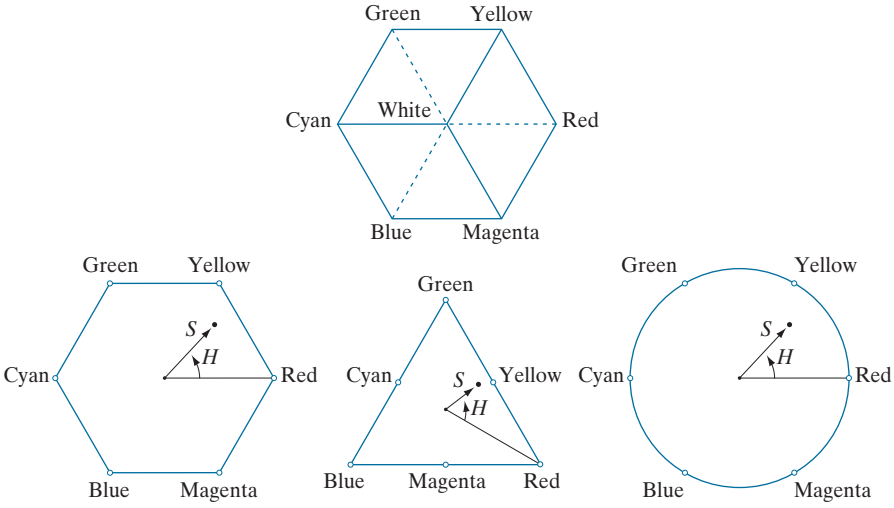


FIGURE 6.10 Conceptual relationships between the RGB and HSI color models.

the black and white points are contained in the plane tells us that the intensity axis also is contained in the plane. Furthermore, we see that *all* points contained in the plane segment defined by the intensity axis and the boundaries of the cube have the *same* hue (cyan in this case). We could arrive at the same conclusion by recalling from Section 6.1 that all colors generated by three colors lie in the triangle defined by those colors. If two of those points are black and white, and the third is a color point, all points on the triangle would have the same hue, because the black and white components cannot change the hue (of course, the intensity and saturation of points in this triangle would be different). By rotating the shaded plane about the vertical intensity axis, we would obtain different hues. From these concepts, we arrive at the conclusion that the hue, saturation, and intensity values required to form the HSI space can be obtained from the RGB color cube. That is, we can convert any RGB point to a corresponding point in the HSI color space by working out the formulas that describe the reasoning outlined in the preceding discussion.

The key point regarding the cube arrangement in Fig. 6.10, and its corresponding HSI color space, is that the HSI space is represented by a vertical intensity axis, and the locus of color points that lie on planes perpendicular to that axis. As the planes move up and down the intensity axis, the boundaries defined by the intersection of each plane with the faces of the cube have either a triangular or a hexagonal shape. This can be visualized much more readily by looking at the cube straight down its grayscale axis, as shown in Fig. 6.11(a). We see that the primary colors are separated by 120°. The secondary colors are 60° from the primaries, which means that the angle between secondaries is 120° also. Figure 6.11(b) shows the same hexagonal shape and an arbitrary color point (shown as a dot). The hue of the point is determined by an angle from some reference point. Usually (but not always) an angle of 0° from the red axis designates 0 hue, and the hue increases counterclockwise from there. The saturation (distance from the vertical axis) is the length of the vector from the origin to the point. Note that the origin is defined by the intersection of the color plane with the vertical intensity axis. The important components of the HSI color space are the vertical intensity axis, the length of the vector to a color point, and the



a
b c d

FIGURE 6.11
Hue and saturation in the HSI color model. The dot is any color point. The angle from the red axis gives the hue. The length of the vector is the saturation. The intensity of all colors in any of these planes is given by the position of the plane on the vertical intensity axis.

angle this vector makes with the red axis. Therefore, it is not unusual to see the HSI planes defined in terms of the hexagon just discussed, a triangle, or even a circle, as Figs. 6.11(c) and (d) show. The shape chosen does not matter because any one of these shapes can be warped into one of the other two by a geometric transformation. Figure 6.12 shows the HSI model based on color triangles, and on circles.

Converting Colors from RGB to HSI

Given an image in RGB color format, the H component of each RGB pixel is obtained using the equation

Computations from RGB to HSI and back are carried out on a pixel-by-pixel basis. We omitted the dependence of the conversion equations on (x,y) for notational clarity.

$$H = \begin{cases} \theta & \text{if } B \le G \\ 360 - \theta & \text{if } B > G \end{cases} \tag{6-16}$$

with[†]

$$\theta = \cos^{-1}\left\{\frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\left[(R-G)^2+(R-B)(G-B)\right]^{1/2}}\right\} \tag{6-17}$$

The saturation component is given by

$$S = 1 - \frac{3}{(R+G+B)}\left[\min(R,G,B)\right] \tag{6-18}$$

[†] It is good practice to add a small number in the denominator of this expression to avoid dividing by 0 when $R = G = B$, in which case θ will be 90°. Note that when all RGB components are equal, Eq. (6-18) gives $S = 0$. In addition, the conversion from HSI back to RGB in Eqs. (6-20) through (6-30) will give $R = G = B = I$, as expected, because, when $R = G = B$, we are dealing with a grayscale image.

a
b

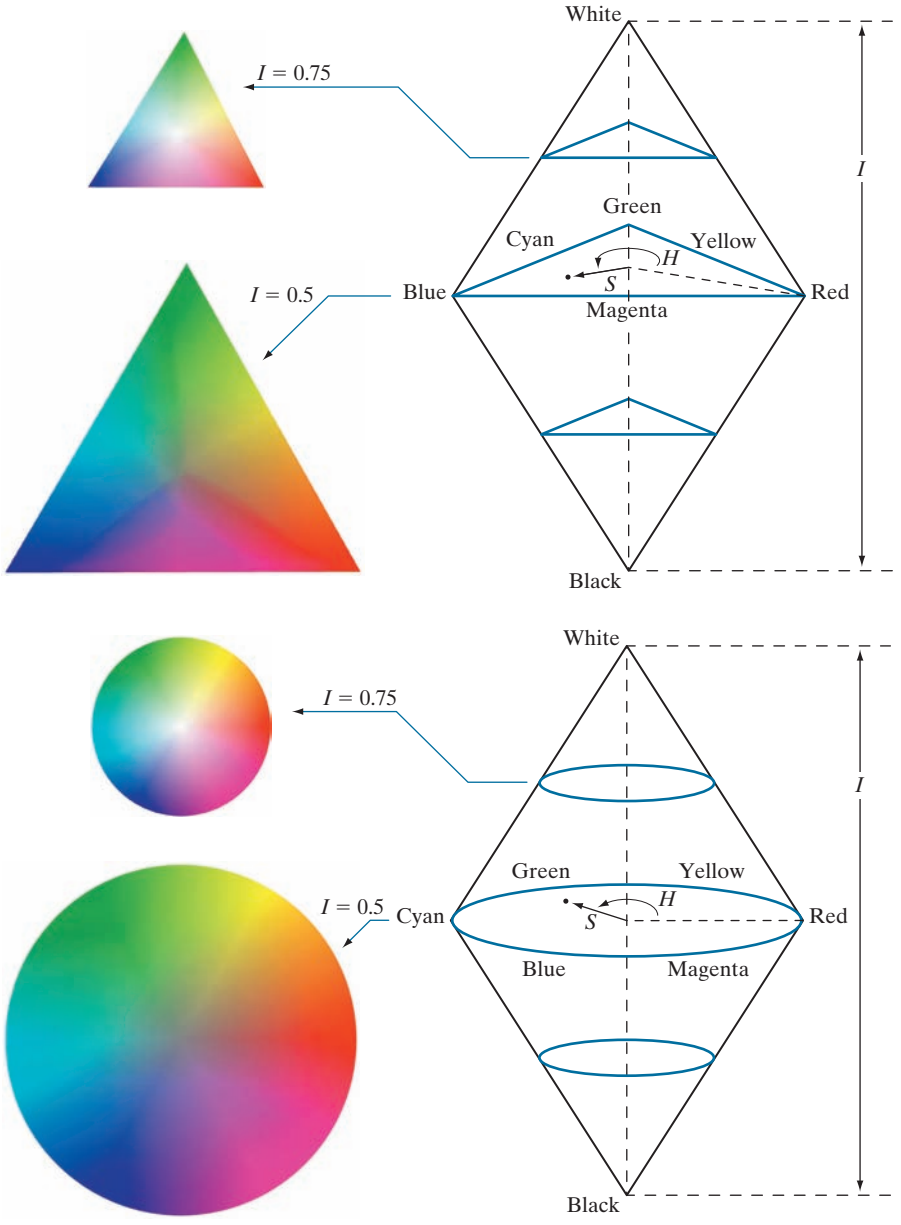


FIGURE 6.12
The HSI color model based on (a) triangular, and (b) circular color planes. The triangles and circles are perpendicular to the vertical intensity axis.

Finally, the intensity component is obtained from the equation

$$I = \frac{1}{3}(R + G + B) \tag{6-19}$$

These equations assume that the RGB values have been normalized to the range $[0, 1]$, and that angle θ is measured with respect to the red axis of the HSI space, as in Fig. 6.11. Hue can be normalized to the range $[0, 1]$ by dividing by $360°$ all values resulting from Eq. (6-16). The other two HSI components already are in this range if the given RGB values are in the interval $[0, 1]$.

The results in Eqs. (6-16) through (6-19) can be derived from the geometry in Figs. 6.10 and 6.11. The derivation is tedious and would not add significantly to the present discussion. You can find the proof for these equations (and for the equations that follow for HSI to RGB conversion) in the *Tutorials* section of the book website.

Converting Colors from HSI to RGB

Given values of HSI in the interval $[0, 1]$, we now want to find the corresponding RGB values in the same range. The applicable equations depend on the values of H. There are three sectors of interest, corresponding to the $120°$ intervals in the separation of primaries (see Fig. 6.11). We begin by multiplying H by $360°$, which returns the hue to its original range of $[0°, 360°]$.

RG sector $(0° \le H < 120°)$: When H is in this sector, the RGB components are given by the equations

$$B = I(1 - S) \tag{6-20}$$

$$R = I\left[1 + \frac{S\cos H}{\cos(60° - H)}\right] \tag{6-21}$$

and

$$G = 3I - (R + B) \tag{6-22}$$

GB sector $(120° \le H < 240°)$: If the given value of H is in this sector, we first subtract $120°$ from it:

$$H = H - 120° \tag{6-23}$$

Then, the RGB components are

$$R = I(1 - S) \tag{6-24}$$

$$G = I\left[1 + \frac{S\cos H}{\cos(60° - H)}\right] \tag{6-25}$$

and

$$B = 3I - (R + G) \tag{6-26}$$

BR sector $(240° \le H \le 360°)$: Finally, if H is in this range, we subtract $240°$ from it:

$$H = H - 240° \tag{6-27}$$

Then, the RGB components are

$$G = I(1 - S) \tag{6-28}$$

$$B = I\left[1 + \frac{S\cos H}{\cos(60° - H)}\right] \tag{6-29}$$

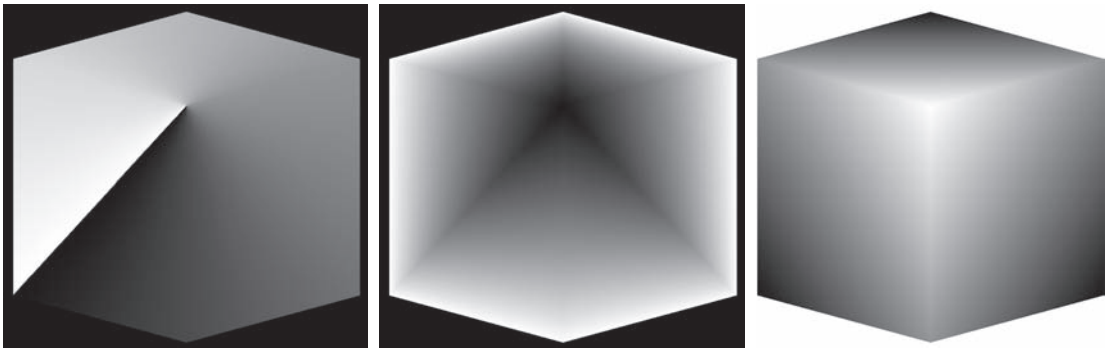
and

$$R = 3I - (G + B) \tag{6-30}$$

We discuss several uses of these equations in the following sections.

EXAMPLE 6.2: The HSI values corresponding to the image of the RGB color cube.

Figure 6.13 shows the hue, saturation, and intensity images for the RGB values in Fig. 6.8. Figure 6.13(a) is the hue image. Its most distinguishing feature is the discontinuity in value along a 45° line in the front (red) plane of the cube. To understand the reason for this discontinuity, refer to Fig. 6.8, draw a line from the red to the white vertices of the cube, and select a point in the middle of this line. Starting at that point, draw a path to the right, following the cube around until you return to the starting point. The major colors encountered in this path are yellow, green, cyan, blue, magenta, and back to red. According to Fig. 6.11, the values of hue along this path should increase from 0° to 360° (i.e., from the lowest to highest



a b c

FIGURE 6.13 HSI components of the image in Fig. 6.8: (a) hue, (b) saturation, and (c) intensity images.

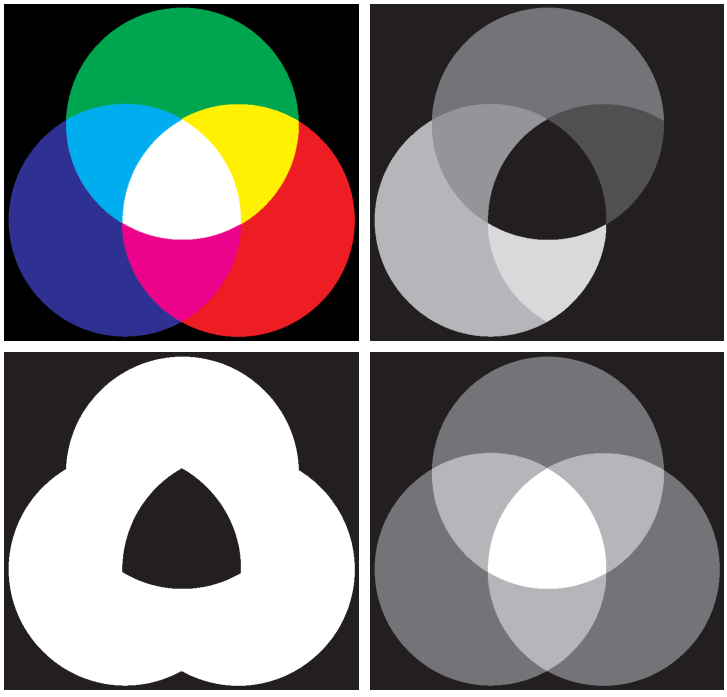
possible values of hue). This is precisely what Fig. 6.13(a) shows, because the lowest value is represented as black and the highest value as white in the grayscale. In fact, the hue image was originally normalized to the range [0, 1] and then scaled to 8 bits; that is, we converted it to the range [0, 255], for display.

The saturation image in Fig. 6.13(b) shows progressively darker values toward the white vertex of the RGB cube, indicating that colors become less and less saturated as they approach white. Finally, every pixel in the intensity image shown in Fig. 6.13(c) is the average of the RGB values at the corresponding pixel in Fig. 6.8.

Manipulating HSI Component Images

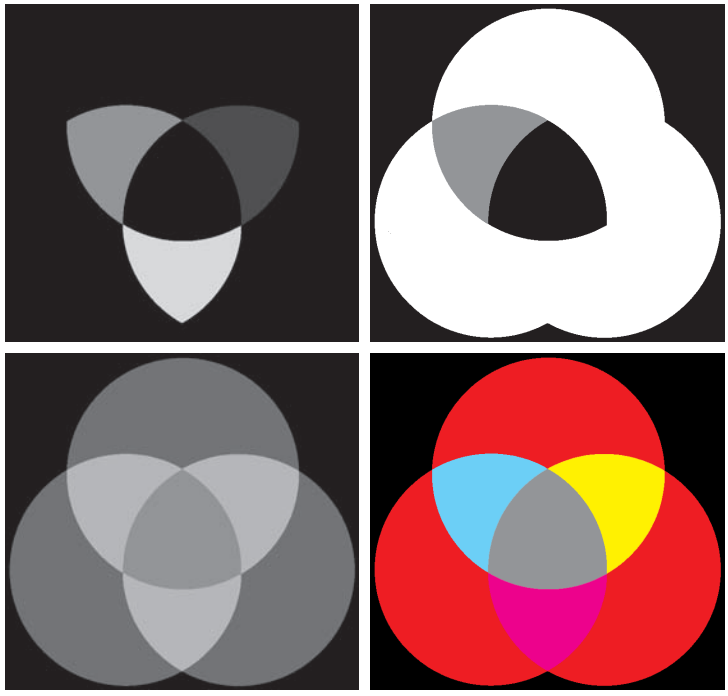
In the following discussion, we take a look at some simple techniques for manipulating HSI component images. This will help you develop familiarity with these comonents, and deepen your understanding of the HSI color model. Figure 6.14(a) shows an image composed of the primary and secondary RGB colors. Figures 6.14(b) through (d) show the H, S, and I components of this image, generated using Eqs. (6-16) through (6-19). Recall from the discussion earlier in this section that the gray-level values in Fig. 6.14(b) correspond to angles; thus, for example, because red corresponds to 0°, the red region in Fig. 6.14(a) is mapped to a black region in the hue image. Similarly, the gray levels in Fig. 6.14(c) correspond to saturation (they were scaled to [0, 255] for display), and the gray levels in Fig. 6.14(d) are average intensities.

To change the individual color of any region in the RGB image, we change the values of the corresponding region in the hue image of Fig. 6.14(b). Then we convert



a b
c d

FIGURE 6.14
(a) RGB image and the components of its corresponding HSI image: (b) hue, (c) saturation, and (d) intensity.



a b
c d

FIGURE 6.15
(a)-(c) Modified HSI component images.
(d) Resulting RGB image. (See Fig. 6.14 for the original HSI images.)

the new H image, along with the unchanged S and I images, back to RGB using the procedure explained in Eqs. (6-20) through (6-30). To change the saturation (purity) of the color in any region, we follow the same procedure, except that we make the changes in the saturation image in HSI space. Similar comments apply to changing the average intensity of any region. Of course, these changes can be made simultaneously. For example, the image in Fig. 6.15(a) was obtained by changing to 0 the pixels corresponding to the blue and green regions in Fig. 6.14(b). In Fig. 6.15(b), we reduced by half the saturation of the cyan region in component image S from Fig. 6.14(c). In Fig. 6.15(c), we reduced by half the intensity of the central white region in the intensity image of Fig. 6.14(d). The result of converting this modified HSI image back to RGB is shown in Fig. 6.15(d). As expected, we see in this figure that the outer portions of all circles are now red; the purity of the cyan region was diminished, and the central region became gray rather than white. Although these results are simple, they clearly illustrate the power of the HSI color model in allowing independent control over hue, saturation, and intensity. These are quantities with which humans are quite familiar when describing colors.

A DEVICE INDEPENDENT COLOR MODEL

As noted earlier, humans see a broad spectrum of colors and color shades. However, color perception differs between individuals. Not only that, but color across devices such as monitors and printers can vary significantly unless these devices are properly calibrated.

Color transformations can be performed on most desktop computers. In conjunction with digital cameras, flatbed scanners, and ink-jet printers, they turn a personal computer into a *digital darkroom*. Also, commercial devices exist that use a combination of spectrometer measurements and software to develop color profiles that can then be loaded on monitors and printers to calibrate their color responses.

The effectiveness of the transformations examined in this section is judged ultimately in print. Because these transformations are developed, refined, and evaluated on monitors, it is necessary to maintain a high degree of color consistency between the monitors used and the eventual output devices. This is best accomplished with a device-independent color model that relates the color gamuts (see Section 6.1) of the monitors and output devices, as well as any other devices being used, to one another. The success of this approach depends on the quality of the color profiles used to map each device to the model, as well as the model itself. The model of choice for many color management systems (CMS) is the CIE $L^*a^*b^*$ model, also called CIELAB (CIE [1978], Robertson [1977]).

The $L^*a^*b^*$ color components are given by the following equations:

$$L^* = 116 \cdot h\left(\frac{Y}{Y_W}\right) - 16 \tag{6-31}$$

$$a^* = 500\left[h\left(\frac{X}{X_W}\right) - h\left(\frac{Y}{Y_W}\right)\right] \tag{6-32}$$

and

$$b^* = 200\left[h\left(\frac{Y}{Y_W}\right) - h\left(\frac{Z}{Z_W}\right)\right] \tag{6-33}$$

where

$$h(q) = \begin{cases} \sqrt[3]{q} & q > 0.008856 \\ 7.787q + 16/116 & q \le 0.008856 \end{cases} \tag{6-34}$$

and X_W, Y_W, and Z_W are reference white tristimulus values—typically the white of a perfectly reflecting diffuser under CIE standard D65 illumination (defined by $x = 0.3127$ and $y = 0.3290$ in the CIE chromaticity diagram of Fig. 6.5). The $L^*a^*b^*$ color space is *colorimetric* (i.e., colors perceived as matching are encoded identically), *perceptually uniform* (i.e., color differences among various hues are perceived uniformly—see the classic paper by MacAdams [1942]), and *device independent*. While $L^*a^*b^*$ colors are not directly displayable (conversion to another color space is required), the $L^*a^*b^*$ gamut encompasses the entire visible spectrum and can represent accurately the colors of any display, print, or input device. Like the HSI system, the $L^*a^*b^*$ system is an excellent decoupler of intensity (represented by lightness L^*) and color (represented by a^* for red minus green and b^* for green minus blue), making it useful in both image manipulation (tone and contrast editing) and image compression applications. Studies indicate that the degree to which