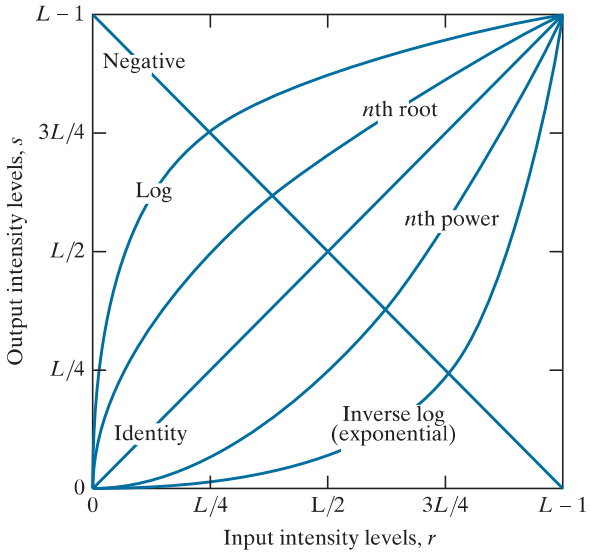
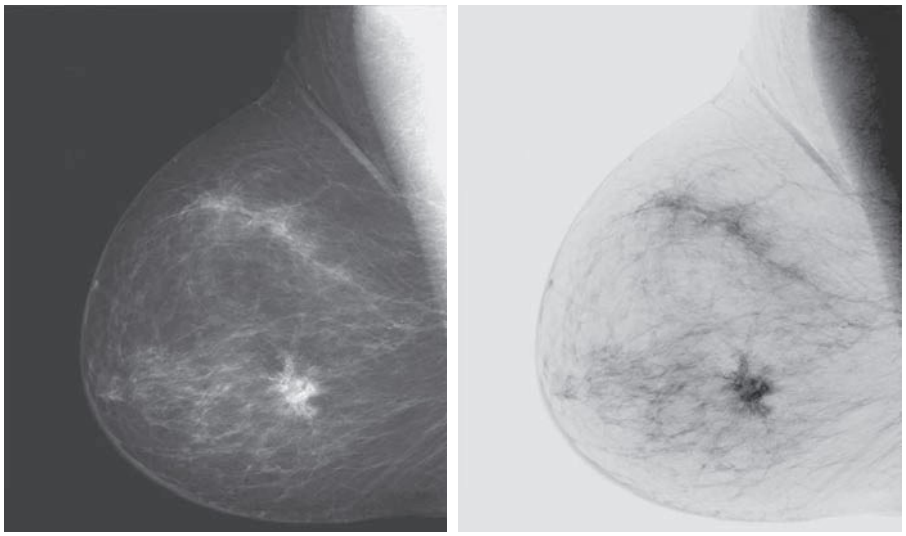


FIGURE 3.3 Some basic intensity transformation functions. Each curve was scaled *independently* so that all curves would fit in the same graph. Our interest here is on the *shapes* of the curves, not on their relative values.

Reversing the intensity levels of a digital image in this manner produces the equivalent of a photographic negative. This type of processing is used, for example, in enhancing white or gray detail embedded in dark regions of an image, especially when the black areas are dominant in size. Figure 3.4 shows an example. The original image is a digital mammogram showing a small lesion. Despite the fact that the visual content is the same in both images, some viewers find it easier to analyze the fine details of the breast tissue using the negative image.



a b

FIGURE 3.4 (a) A digital mammogram. (b) Negative image obtained using Eq. (3-3). (Image (a) Courtesy of General Electric Medical Systems.)

LOG TRANSFORMATIONS

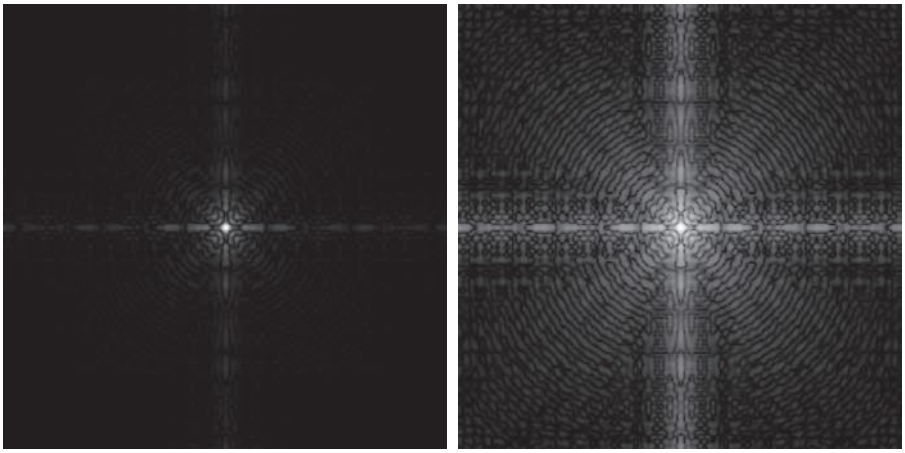
The general form of the log transformation in Fig. 3.3 is

$$s = c\log(1+r) \tag{3-4}$$

where c is a constant and it is assumed that $r \geq 0$. The shape of the log curve in Fig. 3.3 shows that this transformation maps a narrow range of low intensity values in the input into a wider range of output levels. For example, note how input levels in the range $[0, L/4]$ map to output levels to the range $[0, 3L/4]$. Conversely, higher values of input levels are mapped to a narrower range in the output. We use a transformation of this type to expand the values of dark pixels in an image, while compressing the higher-level values. The opposite is true of the inverse log (exponential) transformation.

Any curve having the general shape of the log function shown in Fig. 3.3 would accomplish this spreading/compressing of intensity levels in an image, but the power-law transformations discussed in the next section are much more versatile for this purpose. The log function has the important characteristic that it compresses the dynamic range of pixel values. An example in which pixel values have a large dynamic range is the Fourier spectrum, which we will discuss in Chapter 4. It is not unusual to encounter spectrum values that range from 0 to 10^6 or higher. Processing numbers such as these presents no problems for a computer, but image displays cannot reproduce faithfully such a wide range of values. The net effect is that intensity detail can be lost in the display of a typical Fourier spectrum.

Figure 3.5(a) shows a Fourier spectrum with values in the range 0 to 1.5×10^6. When these values are scaled linearly for display in an 8-bit system, the brightest pixels dominate the display, at the expense of lower (and just as important) values of the spectrum. The effect of this dominance is illustrated vividly by the relatively small area of the image in Fig. 3.5(a) that is not perceived as black. If, instead of displaying the values in this manner, we first apply Eq. (3-4) (with $c = 1$ in this case) to the spectrum values, then the range of values of the result becomes 0 to 6.2. Transforming values in this way enables a greater range of intensities to be shown on the display. Figure 3.5(b) shows the result of scaling the intensity range linearly to the



a b

FIGURE 3.5 (a) Fourier spectrum displayed as a grayscale image. (b) Result of applying the log transformation in Eq. (3-4) with $c = 1$. Both images are scaled to the range $[0, 255]$.

interval [0,255] and showing the spectrum in the same 8-bit display. The level of detail visible in this image as compared to an unmodified display of the spectrum is evident from these two images. Most of the Fourier spectra in image processing publications, including this book, have been scaled in this manner.

POWER-LAW (GAMMA) TRANSFORMATIONS

Power-law transformations have the form

$$s = cr^{\gamma} \tag{3-5}$$

where c and γ are positive constants. Sometimes Eq. (3-5) is written as $s = c(r+\varepsilon)^{\gamma}$ to account for offsets (that is, a measurable output when the input is zero). However, offsets typically are an issue of display calibration, and as a result they are normally ignored in Eq. (3-5). Figure 3.6 shows plots of s as a function of r for various values of γ. As with log transformations, power-law curves with fractional values of γ map a narrow range of dark input values into a wider range of output values, with the opposite being true for higher values of input levels. Note also in Fig. 3.6 that a family of transformations can be obtained simply by varying γ. Curves generated with values of $\gamma > 1$ have exactly the opposite effect as those generated with values of $\gamma < 1$. When $c = \gamma = 1$ Eq. (3-5) reduces to the identity transformation.

The response of many devices used for image capture, printing, and display obey a power law. By convention, the exponent in a power-law equation is referred to as *gamma* [hence our use of this symbol in Eq. (3-5)]. The process used to correct these power-law response phenomena is called *gamma correction* or *gamma encoding*. For example, cathode ray tube (CRT) devices have an intensity-to-voltage response that is a power function, with exponents varying from approximately 1.8 to 2.5. As the curve for $\gamma = 2.5$ in Fig. 3.6 shows, such display systems would tend to produce

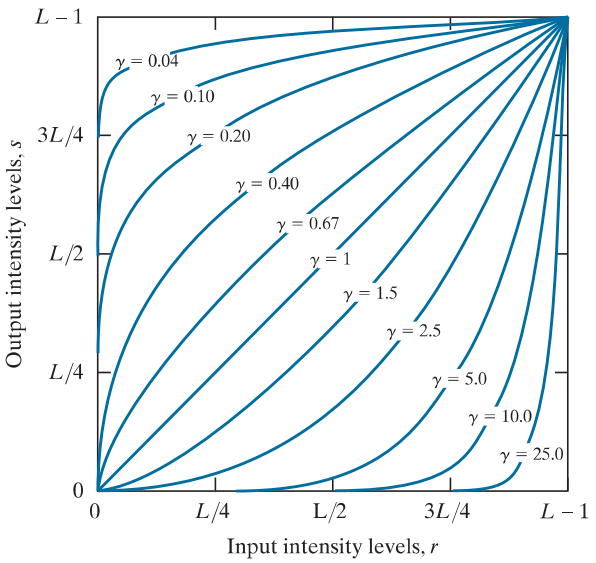


FIGURE 3.6 Plots of the gamma equation $s = cr^{\gamma}$ for various values of γ ($c = 1$ in all cases). Each curve was scaled *independently* so that all curves would fit in the same graph. Our interest here is on the *shapes* of the curves, not on their relative values.

a b
c d

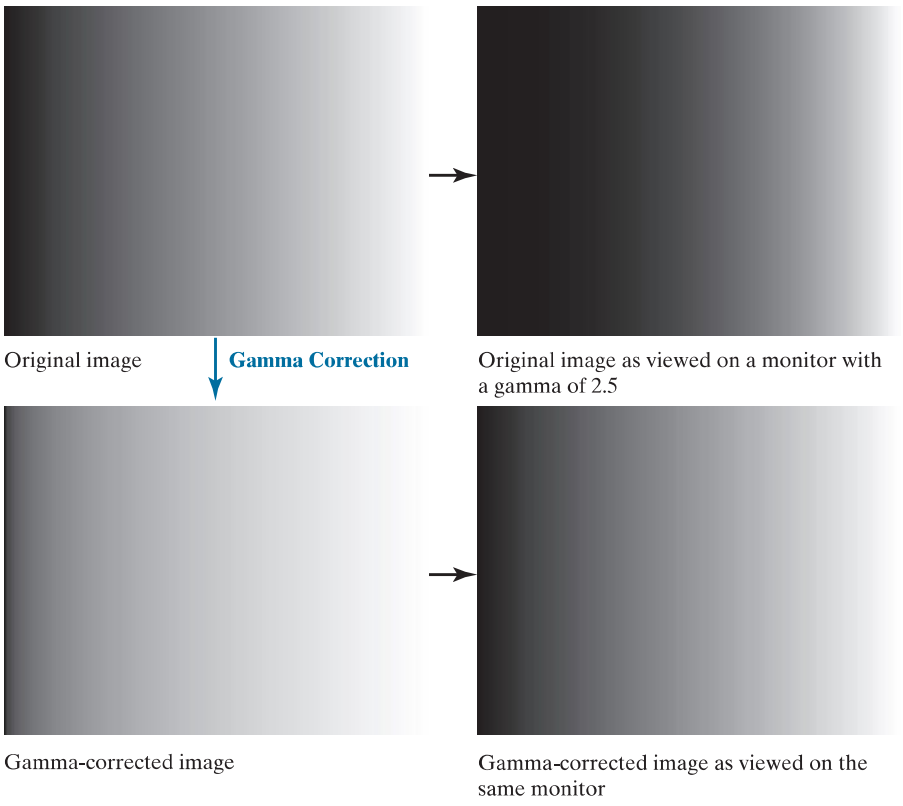


FIGURE 3.7
(a) Intensity ramp image. (b) Image as viewed on a simulated monitor with a gamma of 2.5. (c) Gamma-corrected image. (d) Corrected image as viewed on the same monitor. Compare (d) and (a).

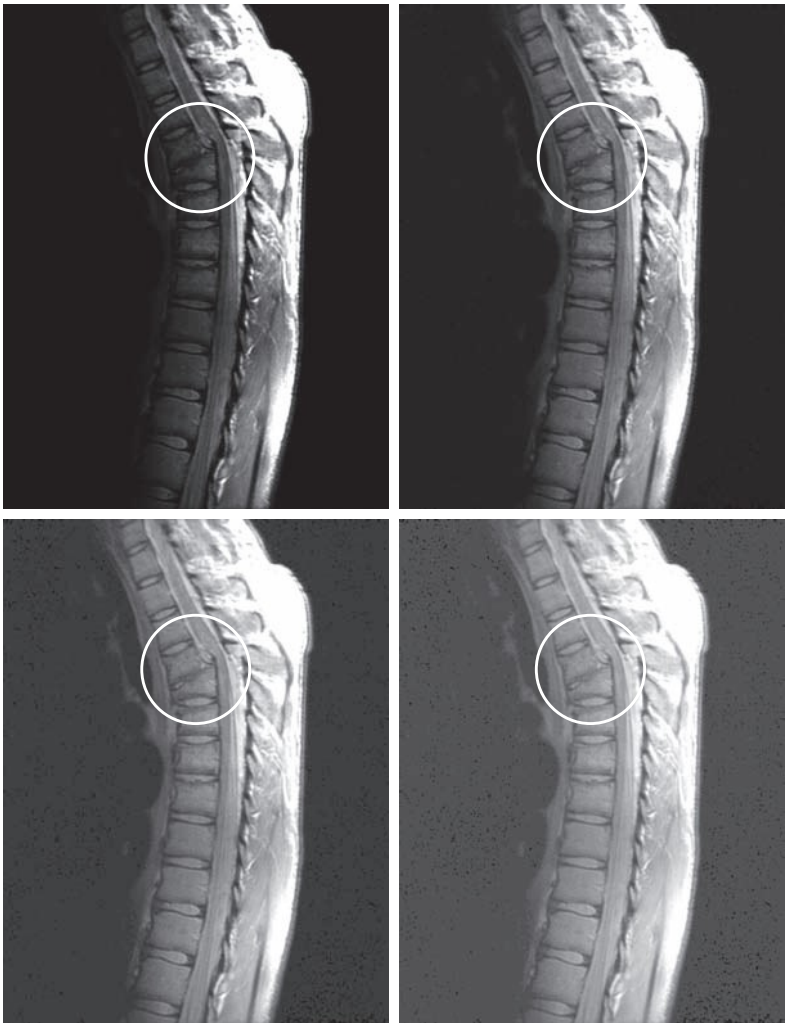
Sometimes, a higher gamma makes the displayed image look better to viewers than the original because of an increase in contrast. However, the objective of gamma correction is to produce a *faithful* display of an input image.

images that are darker than intended. Figure 3.7 illustrates this effect. Figure 3.7(a) is an image of an intensity ramp displayed in a monitor with a gamma of 2.5. As expected, the output of the monitor appears darker than the input, as Fig. 3.7(b) shows.

In this case, gamma correction consists of using the transformation $s = r^{1/2.5} = r^{0.4}$ to preprocess the image before inputting it into the monitor. Figure 3.7(c) is the result. When input into the same monitor, the gamma-corrected image produces an output that is close in appearance to the original image, as Fig. 3.7(d) shows. A similar analysis as above would apply to other imaging devices, such as scanners and printers, the difference being the device-dependent value of gamma (Poynton [1996]).

EXAMPLE 3.1: Contrast enhancement using power-law intensity transformations.

In addition to gamma correction, power-law transformations are useful for general-purpose contrast manipulation. Figure 3.8(a) shows a magnetic resonance image (MRI) of a human upper thoracic spine with a fracture dislocation. The fracture is visible in the region highlighted by the circle. Because the image is predominantly dark, an expansion of intensity levels is desirable. This can be accomplished using a power-law transformation with a fractional exponent. The other images shown in the figure were obtained by processing Fig. 3.8(a) with the power-law transformation function of Eq. (3-5). The values



a b
c d

FIGURE 3.8
(a) Magnetic resonance image (MRI) of a fractured human spine (the region of the fracture is enclosed by the circle). (b)–(d) Results of applying the transformation in Eq. (3-5) with $c = 1$ and $\gamma = 0.6$, 0.4, and 0.3, respectively. (Original image courtesy of Dr. David R. Pickens, Department of Radiology and Radiological Sciences, Vanderbilt University Medical Center.)

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

EXAMPLE 3.2: Another illustration of power-law transformations.

Figure 3.9(a) shows the opposite problem of that presented in Fig. 3.8(a). The image to be processed

a b
c d

FIGURE 3.9
(a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3-5) with $\gamma = 3.0$, 4.0, and 5.0, respectively. ($c = 1$ in all cases.) (Original image courtesy of NASA.)

now has a washed-out appearance, indicating that a compression of intensity levels is desirable. This can be accomplished with Eq. (3-5) using values of γ greater than 1. The results of processing Fig. 3.9(a) with $\gamma = 3.0$, 4.0, and 5.0 are shown in Figs. 3.9(b) through (d), respectively. Suitable results were obtained using gamma values of 3.0 and 4.0. The latter result has a slightly more appealing appearance because it has higher contrast. This is true also of the result obtained with $\gamma = 5.0$. For example, the airport runways near the middle of the image appears clearer in Fig. 3.9(d) than in any of the other three images.

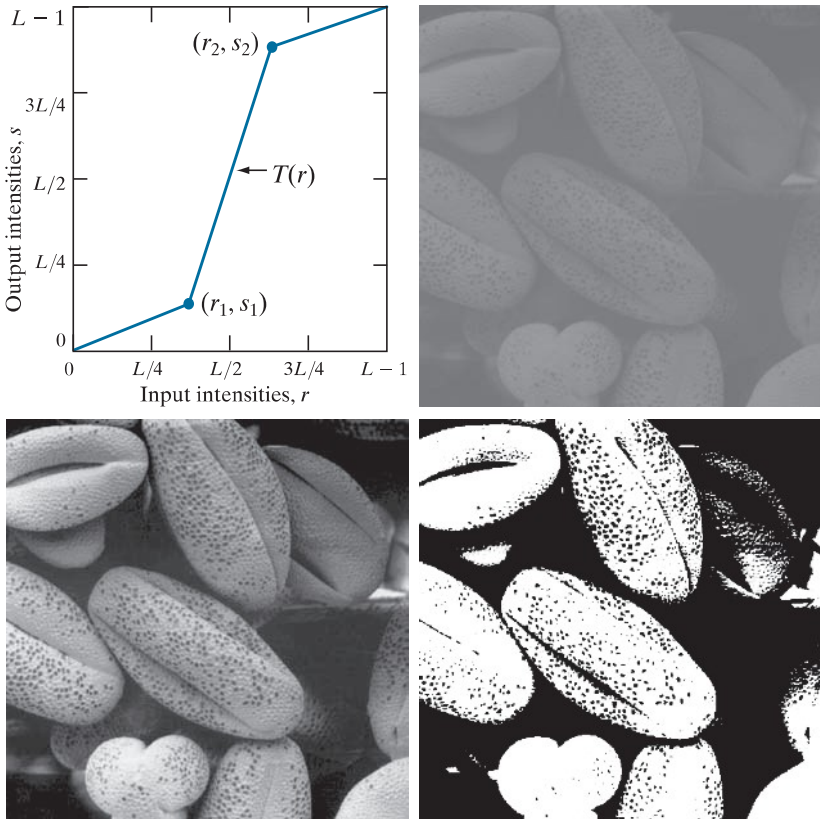
PIECEWISE LINEAR TRANSFORMATION FUNCTIONS

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

Contrast Stretching

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

Figure 3.10(a) shows a typical transformation used for contrast stretching. The locations of points (r_1, s_1) and (r_2, s_2) control the shape of the transformation function. If $r_1 = s_1$ and $r_2 = s_2$ the transformation is a linear function that produces no changes in intensity. If $r_1 = r_2$, $s_1 = 0$, and $s_2 = L-1$ the transformation becomes a *thresholding function* that creates a binary image [see Fig. 3.2(b)]. Intermediate values of (r_1, s_1) and (s_2, r_2) produce various degrees of spread in the intensity levels of the output image, thus affecting its contrast. In general, $r_1 \leq r_2$ and $s_1 \leq s_2$ is assumed so that the function is single valued and monotonically increasing. This preserves the order of intensity levels, thus preventing the creation of intensity artifacts. Figure 3.10(b) shows an 8-bit image with low contrast. Figure 3.10(c) shows the result of contrast stretching, obtained by setting $(r_1, s_1) = (r_{\min}, 0)$ and $(r_2, s_2) = (r_{\max}, L-1)$, where $r_{\min}$ and $r_{\max}$ denote the minimum and maximum intensity levels in the input image,



a b
c d

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

a b

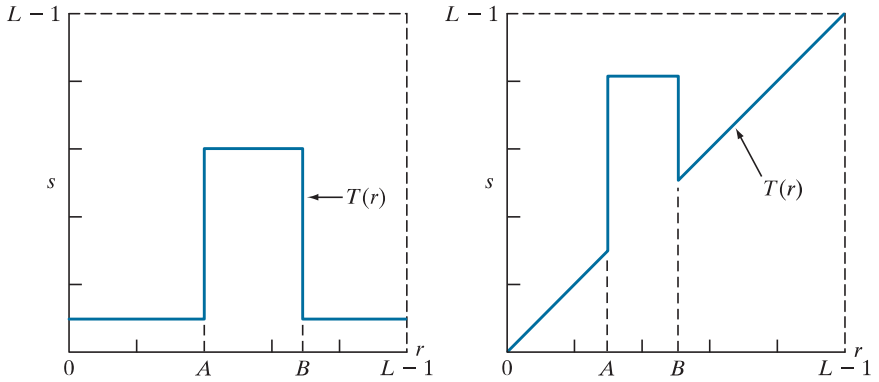


FIGURE 3.11
(a) This transformation function highlights range $[A, B]$ and reduces all other intensities to a lower level. (b) This function highlights range $[A, B]$ and leaves other intensities unchanged.

respectively. The transformation stretched the intensity levels linearly to the full intensity range, $[0, L-1]$. Finally, Fig. 3.10(d) shows the result of using the thresholding function, with $(r_1, s_1) = (m, 0)$ and $(r_2, s_2) = (m, L-1)$, where m is the mean intensity level in the image.

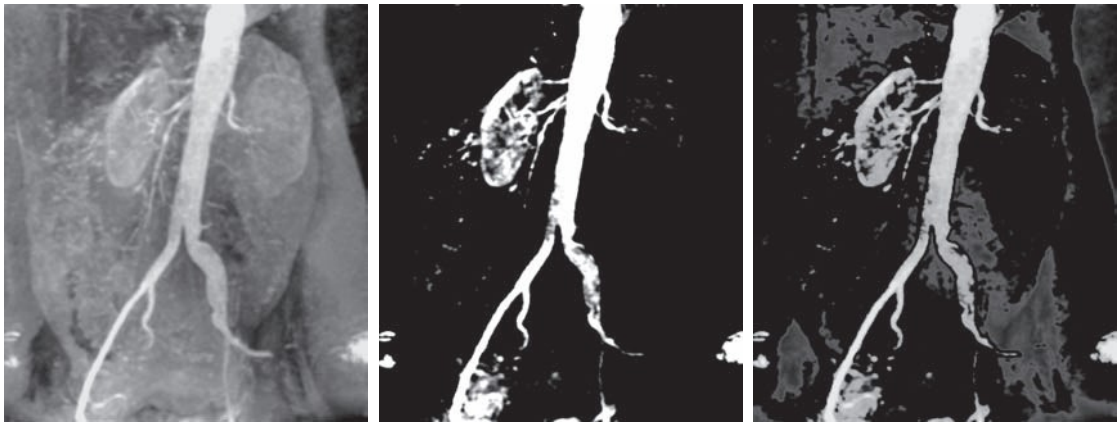
Intensity-Level Slicing

There are applications in which it is of interest to highlight a specific range of intensities in an image. Some of these applications include enhancing features in satellite imagery, such as masses of water, and enhancing flaws in X-ray images. The method, called *intensity-level slicing*, can be implemented in several ways, but most are variations of two basic themes. One approach is to display in one value (say, white) all the values in the range of interest and in another (say, black) all other intensities. This transformation, shown in Fig. 3.11(a), produces a binary image. The second approach, based on the transformation in Fig. 3.11(b), brightens (or darkens) the desired range of intensities, but leaves all other intensity levels in the image unchanged.

EXAMPLE 3.3: Intensity-level slicing.

Figure 3.12(a) is an aortic angiogram near the kidney area (see Section 1.3 for details on this image). The objective of this example is to use intensity-level slicing to enhance the major blood vessels that appear lighter than the background, as a result of an injected contrast medium. Figure 3.12(b) shows the result of using a transformation of the form in Fig. 3.11(a). The selected band was near the top of the intensity scale because the range of interest is brighter than the background. The net result of this transformation is that the blood vessel and parts of the kidneys appear white, while all other intensities are black. This type of enhancement produces a binary image, and is useful for studying the shape characteristics of the flow of the contrast medium (to detect blockages, for example).

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



a b c

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

Bit-Plane Slicing

Pixel values are integers composed of bits. For example, values in a 256-level grayscale image are composed of 8 bits (one byte). Instead of highlighting intensity-level ranges, as 3.3, we could highlight the contribution made to total image appearance by specific bits. As Fig. 3.13 illustrates, an 8-bit image may be considered as being composed of eight one-bit planes, with plane 1 containing the lowest-order bit of all pixels in the image, and plane 8 all the highest-order bits.

Figure 3.14(a) shows an 8-bit grayscale image and Figs. 3.14(b) through (i) are its eight one-bit planes, with Fig. 3.14(b) corresponding to the highest-order bit. Observe that the four higher-order bit planes, especially the first two, contain a significant amount of the visually-significant data. The lower-order planes contribute to more subtle intensity details in the image. The original image has a gray border whose intensity is 194. Notice that the corresponding borders of some of the bit

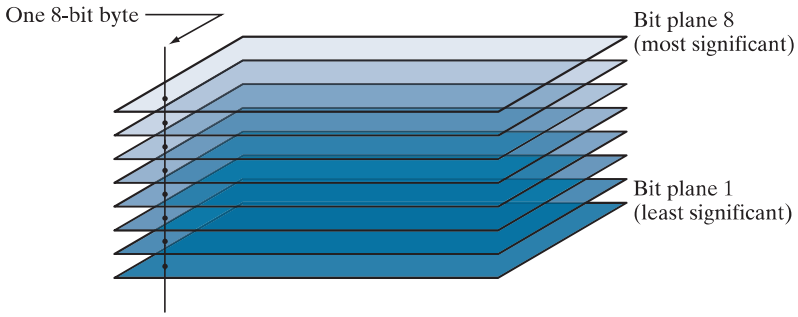
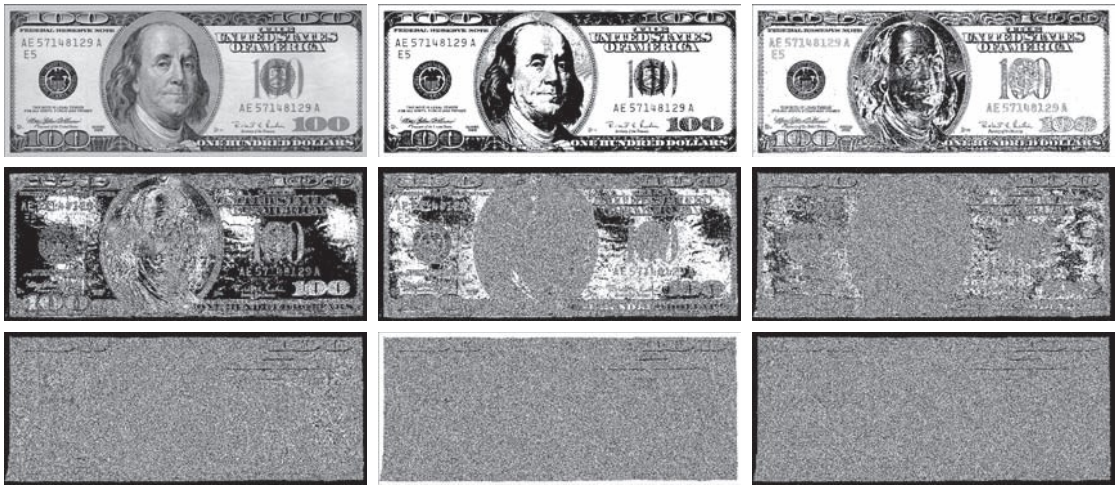


FIGURE 3.13 Bit-planes of an 8-bit image.



a b c
d e f
g h i

FIGURE 3.14 (a) An 8-bit gray-scale image of size 550×1192 pixels. (b) through (i) Bit planes 8 through 1, with bit plane 1 corresponding to the least significant bit. Each bit plane is a binary image..

planes are black (0), while others are white (1). To see why, consider a pixel in, say, the middle of the lower border of Fig. 3.14(a). The corresponding pixels in the bit planes, starting with the highest-order plane, have values 1 1 0 0 0 0 1 0, which is the binary representation of decimal 194. The value of any pixel in the original image can be similarly reconstructed from its corresponding binary-valued pixels in the bit planes by converting an 8-bit binary sequence to decimal.

The binary image for the 8th bit plane of an 8-bit image can be obtained by thresholding the input image with a transformation function that maps to 0 intensity values between 0 and 127, and maps to 1 values between 128 and 255. The binary image in Fig. 3.14(b) was obtained in this manner. It is left as an exercise (see Problem 3.3) to obtain the transformation functions for generating the other bit planes.

Decomposing an image into its bit planes is useful for analyzing the relative importance of each bit in the image, a process that aids in determining the adequacy of the number of bits used to quantize the image. Also, this type of decomposition is useful for image compression (the topic of Chapter 8), in which fewer than all planes are used in reconstructing an image. For example, Fig. 3.15(a) shows an image reconstructed using bit planes 8 and 7 of the preceding decomposition. The reconstruction is done by multiplying the pixels of the nth plane by the constant 2^{n-1}. This converts the nth significant binary bit to decimal. Each bit plane is multiplied by the corresponding constant, and all resulting planes are added to obtain the grayscale image. Thus, to obtain Fig. 3.15(a), we multiplied bit plane 8 by 128, bit plane 7 by 64, and added the two planes. Although the main features of the original image were restored, the reconstructed image appears flat, especially in the background. This

a b c **FIGURE 3.15** Image reconstructed from bit planes: (a) 8 and 7; (b) 8, 7, and 6; (c) 8, 7, 6, and 5.

is not surprising, because two planes can produce only four distinct intensity levels. Adding plane 6 to the reconstruction helped the situation, as Fig. 3.15(b) shows. Note that the background of this image has perceptible false contouring. This effect is reduced significantly by adding the 5th plane to the reconstruction, as Fig. 3.15(c) illustrates. Using more planes in the reconstruction would not contribute significantly to the appearance of this image. Thus, we conclude that, in this example, storing the four highest-order bit planes would allow us to reconstruct the original image in acceptable detail. Storing these four planes instead of the original image requires 50% less storage.

3.3 HISTOGRAM PROCESSING

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

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

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

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

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

Histogram shape is related to image appearance. For example, Fig. 3.16 shows images with four basic intensity characteristics: dark, light, low contrast, and high contrast; the image histograms are also shown. We note in the dark image that the most populated histogram bins are concentrated on the lower (dark) end of the intensity scale. Similarly, the most populated bins of the light image are biased toward the higher end of the scale. An image with low contrast has a narrow histo-