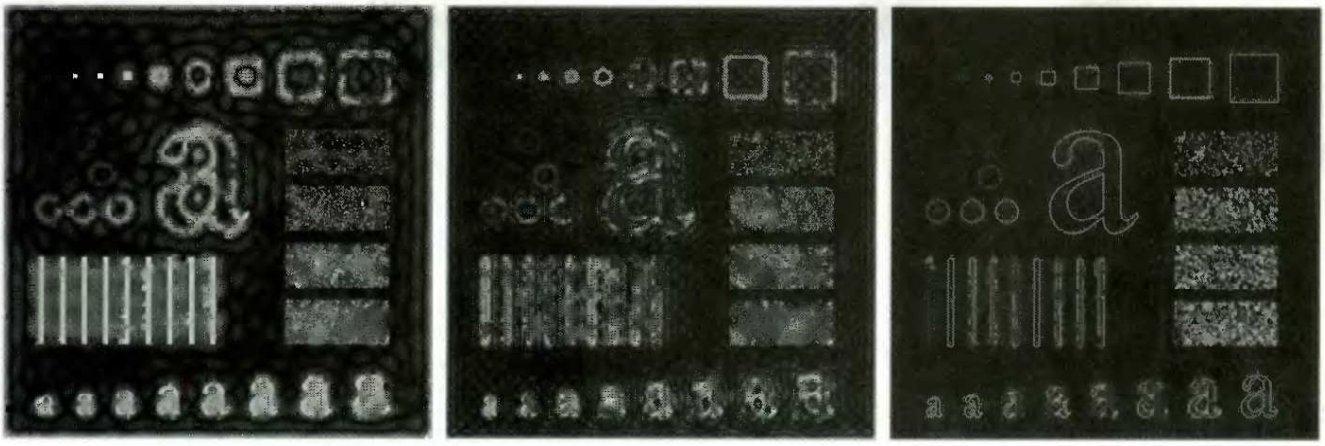


a b c

FIGURE 4.24 Results of ideal highpass filtering the image in Fig. 4.11(a) with $D_0 = 15$, 30, and 80, respectively. Problems with ringing are quite evident in (a) and (b).

these three objects is much closer to the background gray level, giving discontinuities of smaller magnitude). Looking at the "spot" size of the spatial representation of the IHPF in Fig. 4.23(a) and keeping in mind that filtering in the spatial domain is convolution of the filter with the image helps explain why the smaller objects and lines appear almost solid white. Look in particular at the three small squares in the top row and the thin, vertical bars. The situation improved somewhat with $D_0 = 30$. Edge distortion still is quite evident, but now we begin to see filtering on the smaller objects. Due to the now familiar inverse relationship between the frequency and spatial domains, we know that the spot size of this filter is smaller than the spot of the filter with $D_0 = 5$. The result for $D_0 = 80$ is more of what a highpass-filtered image should look like. Here, the edges are much cleaner and less distorted, and the smaller objects have been filtered properly.

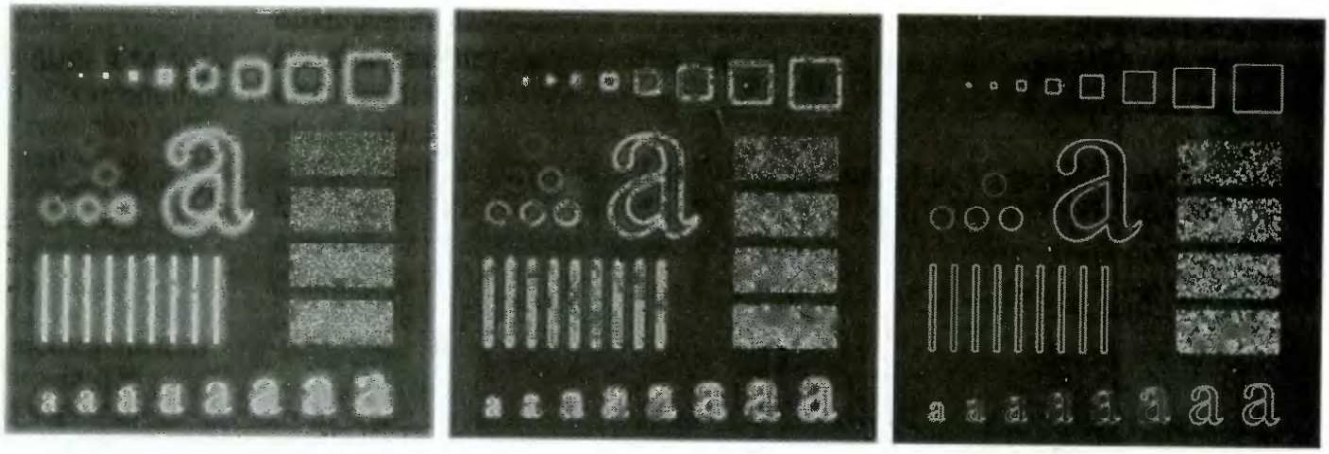
4.4.2 Butterworth Highpass Filters

The transfer function of the Butterworth highpass filter (BHPF) of order n and with cutoff frequency locus at a distance D_0 from the origin is given by

$$H(u, v) = \frac{1}{1 + \left[D_0/D(u, v)\right]^{2n}} \tag{4.4-3}$$

where $D(u, v)$ is given in Eq. (4.3-3). Equation (4.4-3) follows directly from Eqs. (4.4-1) and (4.3-6). The middle row of Fig. 4.22 shows an image and cross section of a BHPF function.

As in the case of lowpass filters, we can expect Butterworth highpass filters to behave smoother than IHPFs. The performance of a BHPF, of order 2 and with D_0 set to the same values as in Fig. 4.24, is shown in Fig. 4.25. The boundaries are much less distorted than in Fig. 4.24, even for the smallest value of cutoff frequency. Since the center spot sizes of the IHPF and the BHPF are similar [see Figs. 4.23(a) and (b)], the performance of the two filters in terms of filtering the smaller objects is comparable. The transition into higher values of cutoff frequencies is much smoother with the BHPF.



a b c

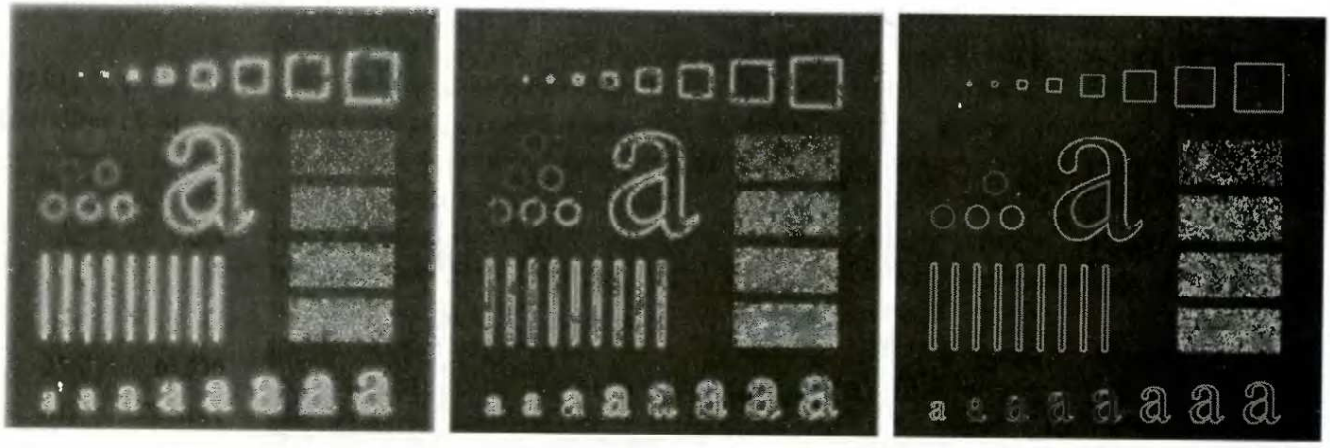
FIGURE 4.25 Results of highpass filtering the image in Fig. 4.11(a) using a BHPF of order 2 with $D_0 = 15$, 30, and 80, respectively. These results are much smoother than those obtained with an ILPF.

4.4.3 Gaussian Highpass Filters

The transfer function of the Gaussian highpass filter (GHPF) with cutoff frequency locus at a distance D_0 from the origin is given by

$$H(u, v) = 1 - e^{-D^2(u,v)/2D_0^2} \tag{4.4-4}$$

where $D(u, v)$ is given in Eq. (4.3-3). This equation follows directly from Eqs. (4.4-1) and (4.3-8). The third row in Fig. 4.22 shows a perspective plot, image, and cross section of the GHPF function. Following the same format as for the BHPF, we show in Fig. 4.26 comparable results using GHPFs. As expected, the results obtained are smoother than with the previous two filters. Even the filtering of the smaller objects and thin bars is cleaner with the Gaussian filter.



a b c

FIGURE 4.26 Results of highpass filtering the image of Fig. 4.11(a) using a GHPF of order 2 with $D_0 = 15$, 30, and 80, respectively. Compare with Figs. 4.24 and 4.25.

As discussed in Section 4.2.4, it is possible to construct highpass filters as the difference of Gaussian lowpass filters. These difference filters have more parameters and, therefore, allow more control over the filter shape. However, the simple filter of Eq. (4.4-4) usually is quite adequate in practice, and it is an easier formulation for experimenting.

4.4.4 The Laplacian in the Frequency Domain

It can be shown that

$$\Im\left[\frac{d^n f(x)}{dx^n}\right] = (ju)^n F(u). \tag{4.4-5}$$

From this simple expression, it follows that

$$\begin{aligned}\Im\left[\frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}\right] &= (ju)^2 F(u,v) + (jv)^2 F(u,v) \\ &= -(u^2 + v^2)F(u,v).\end{aligned} \tag{4.4-6}$$

The expression inside the brackets on the left side of Eq. (4.4-6) is recognized as the Laplacian of $f(x, y)$, defined in Eq. (3.7-1). Thus, we have the important result

$$\Im[\nabla^2 f(x,y)] = -(u^2 + v^2)F(u,v), \tag{4.4-7}$$

which simply says that the Laplacian can be implemented in the frequency domain by using the filter

$$H(u,v) = -(u^2 + v^2). \tag{4.4-8}$$

As in all filtering operations in this chapter, the assumption is that the origin of $F(u, v)$ has been centered by performing the operation $f(x, y)(-1)^{x+y}$ prior to taking the transform of the image. As discussed earlier, if f (and F) are of size $M \times N$, this operation shifts the center transform so that $(u, v) = (0, 0)$ is at point $(M/2, N/2)$ in the frequency rectangle. As before, the center of the filter function also needs to be shifted:

$$H(u,v) = -[(u - M/2)^2 + (v - N/2)^2]. \tag{4.4-9}$$

The Laplacian-filtered image in the spatial domain is obtained by computing the inverse Fourier transform of $H(u, v)F(u, v)$:

$$\nabla^2 f(x,y) = \Im^{-1}\{-[(u - M/2)^2 + (v - N/2)^2]F(u,v)\}. \tag{4.4-10}$$

Conversely, computing the Laplacian in the spatial domain using Eq. (3.7-1) and computing the Fourier transform of the result is equivalent to multiplying $F(u, v)$ by $H(u, v)$. We express this dual relationship in the familiar Fourier-transform-pair notation

$$\nabla^2 f(x,y) \Leftrightarrow -[(u - M/2)^2 + (v - N/2)^2]F(u,v). \tag{4.4-11}$$

The spatial domain Laplacian filter function obtained by taking the inverse Fourier transform of Eq. (4.4-9) has some interesting properties, as Fig. 4.27 shows. Figure 4.27(a) is a 3-D perspective plot of Eq. (4.4-9). The function is

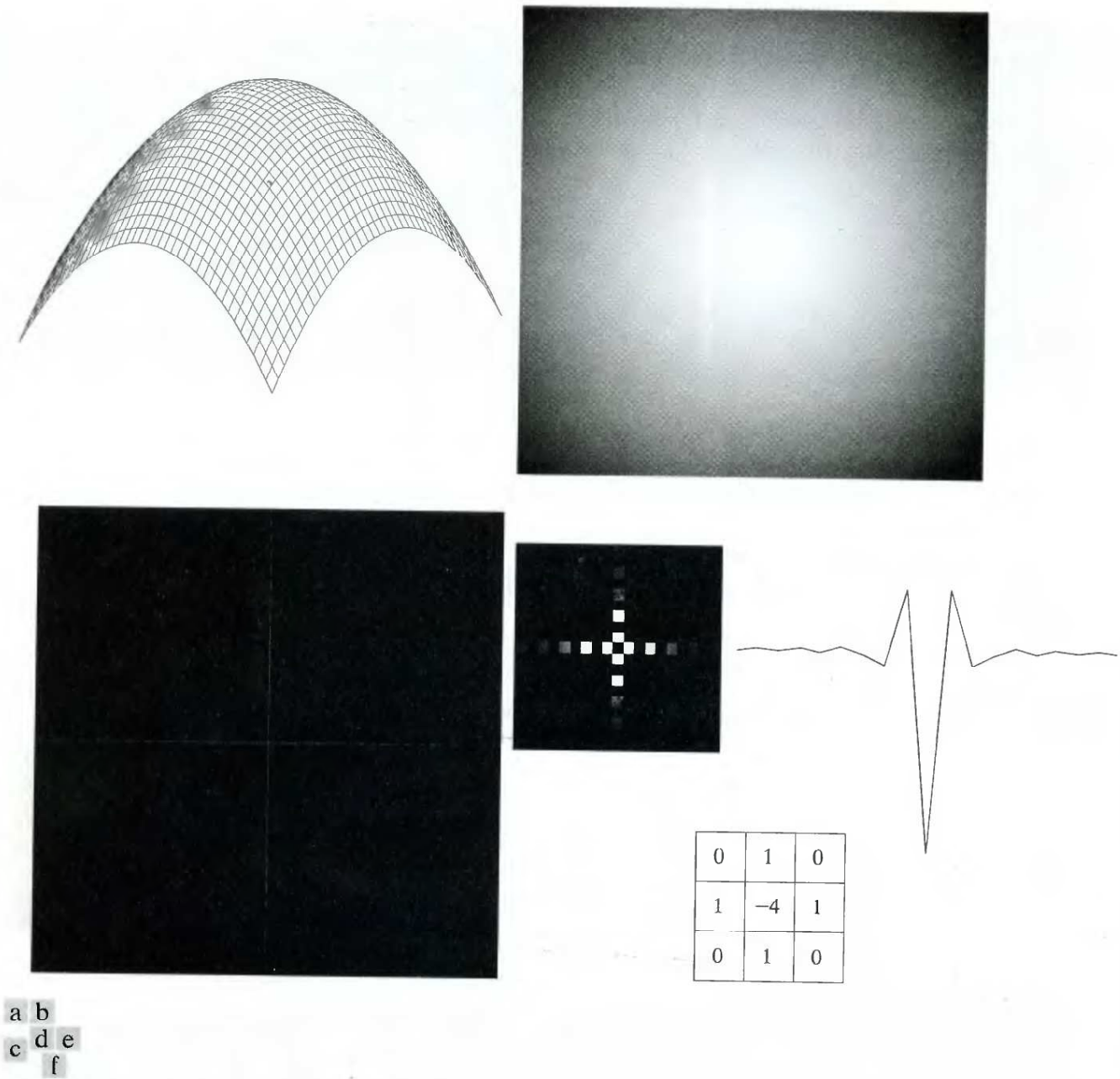


FIGURE 4.27 (a) 3-D plot of Laplacian in the frequency domain. (b) Image representation of (a). (c) Laplacian in the spatial domain obtained from the inverse DFT of (b). (d) Zoomed section of the origin of (c). (e) Gray-level profile through the center of (d). (f) Laplacian mask used in Section 3.7.

centered at $(M/2, N/2)$, and its value at the top of the dome is zero. All other values are negative. Figure 4.27(b) shows $H(u, v)$ as an image, also centered. Figure 4.27(c) is the Laplacian in the *spatial* domain, obtained by multiplying by $H(u, v)$ by $(-1)^{u+v}$, taking the inverse Fourier transform, and multiplying the real part of the result by $(-1)^{x+y}$. Figure 4.27(d) is a zoomed section at about the origin of Fig. 4.27(c). Figure 4.27(e) is a horizontal gray-level profile passing through the center of the zoomed section. Finally, Fig. 4.27(f) shows the mask we used in Section 3.7 to implement the definition of the discrete

Laplacian in the spatial domain, given in Eq. (3.7-4). A horizontal profile through the center of this mask has the same basic shape as the profile in Fig. 4.27(e) (that is, a negative value between two smaller positive values). It is of interest to note that, had we started in the frequency domain, we would have been led to a spatial mask similar in form to Fig. 3.39(a).

As in Eq. (3.7-5), we form an enhanced image $g(x, y)$ by subtracting the Laplacian from the original image:

$$g(x, y) = f(x, y) - \nabla^2 f(x, y). \tag{4.4-12}$$

The Laplacian is subtracted from (rather than added to) the original because of the negative sign in Eq. (4.4-8). This is consistent with the formulation in the spatial domain given in Eq. (3.7-5). We would have arrived at the same conclusion by noting that the center spike in Fig. 4.27(e) is negative, with its immediate neighbors being positive.

As in the spatial domain, where we obtained the enhanced image with a single mask, it is possible to perform the entire operation in the frequency domain with only one filter, given by $H(u, v) = [1 + [(u - M/2)^2 + (v - N/2)^2]]$. In this case the enhanced image is obtained with a single inverse transform operation:

$$g(x, y) = \Im^{-1}\{[1 + ((u - M/2)^2 + (v - N/2)^2)]F(u, v)\}. \tag{4.4-13}$$

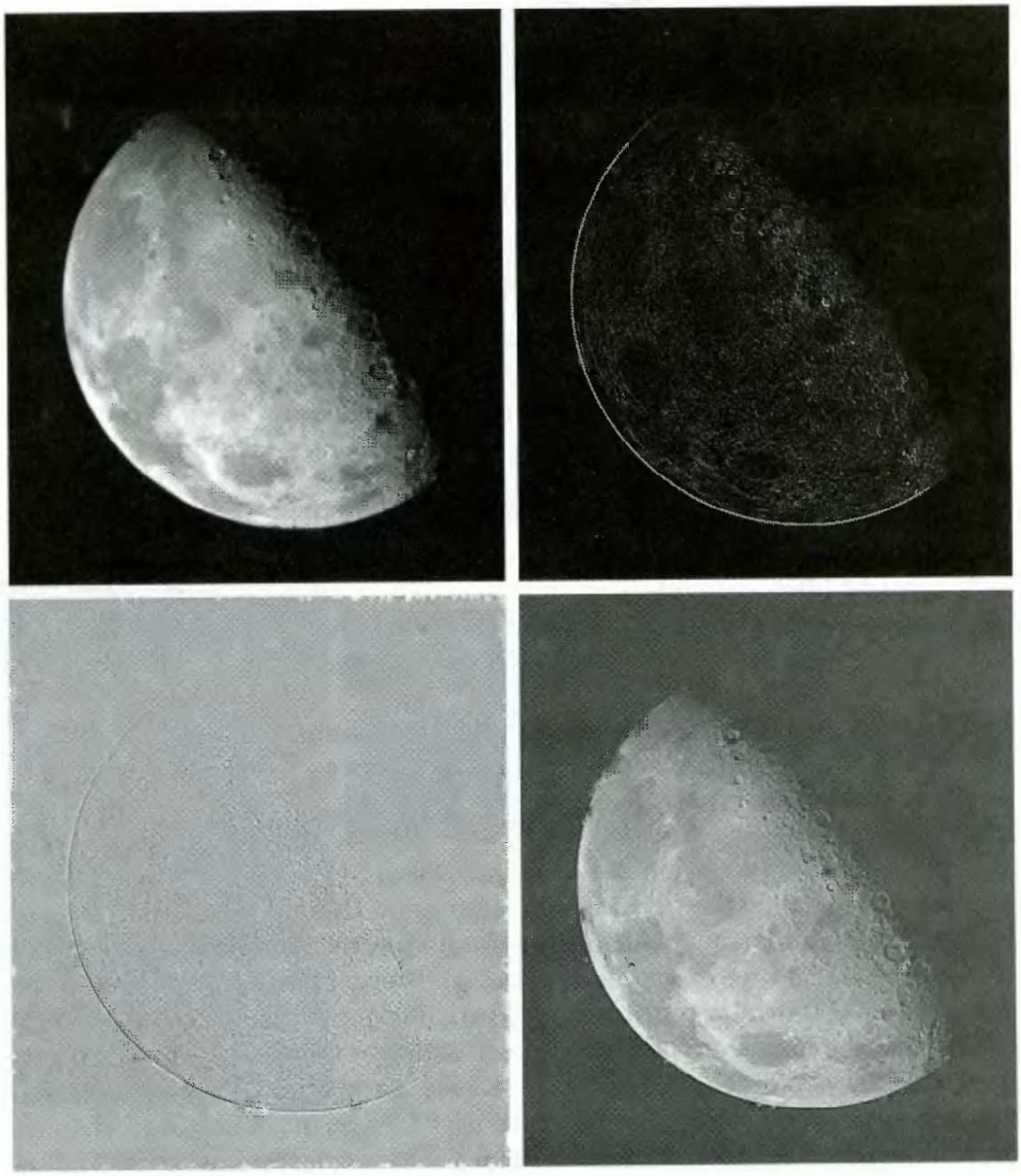
In order to obtain the proper result with this implementation, care must be exercised in scaling the filter because the squared filter variables can be several orders of magnitude greater than 1.

EXAMPLE 4.7: Illustration of the Laplacian in the frequency domain.

■ Figure 4.28(a) is the same as Fig. 3.40(a). Figure 4.28(b) shows the result of filtering this image in the frequency domain using Eq. (4.4-10). As is typical of Laplacian-filtered images, which contain both positive and negative values of comparable magnitudes, scaling is necessary here. Figure 4.28(c) shows the image scaled (for display purposes only) so that its most negative value is scaled to zero and the maximum positive value is scaled to the maximum displayable gray level (255 in this case). Finally, Fig. 4.28(d) shows the enhanced result obtained by using Eq. (4.4-12). The increase in sharpness of small feature detail is unmistakable, as expected of results obtained by using the Laplacian. The sequence of images just presented should be compared with Fig. 3.40, which shows exactly the same sequence of steps, but computed using only spatial domain techniques. The results are identical for all practical purposes. ■

4.4.5 Unsharp Masking, High-Boost Filtering, and High-Frequency Emphasis Filtering

All the filtered images in Sections 4.4.1 through 4.4.3 have one thing in common: Their average background intensity has been reduced to near black. This is due to the fact that the highpass filters we applied to those images eliminate the zero-frequency component of their Fourier transforms (see the discussion in Section 4.2.3 regarding this phenomenon). As discussed in Section 3.7.2, the solution to this problem consists of adding a portion of the image back to the filtered result. In fact, enhancement using the Laplacian does precisely this, by adding back the entire image to the filtered result. Sometimes it is advantageous



a b
c d

FIGURE 4.28 (a) Image of the North Pole of the moon. (b) Laplacian filtered image. (c) Laplacian image scaled. (d) Image enhanced by using Eq. (4.4-12). (Original image courtesy of NASA.)

to increase the contribution made by the original image to the overall filtered result. This approach, called *high-boost filtering*, is a generalization of *unsharp masking*. These concepts were introduced in Section 3.7.2. We repeat them here using frequency domain concepts and notation.

Unsharp masking consists simply of generating a sharp image by subtracting from an image a blurred version of itself. Using frequency domain terminology, this means obtaining a highpass-filtered image by subtracting from the image a lowpass-filtered version of itself. That is,

$$f_{\text{hp}}(x, y) = f(x, y) - f_{\text{lp}}(x, y). \tag{4.4-14}$$

High-boost filtering generalizes this by multiplying $f(x, y)$ by a constant $A \geq 1$:

$$f_{\text{hb}} = Af(x, y) - f_{\text{lp}}(x, y). \tag{4.4-15}$$

Thus, high-boost filtering gives us the flexibility to increase the contribution made by the image to the overall enhanced result. This equation may be written as

$$f_{hb}(x, y) = (A - 1)f(x, y) + f(x, y) - f_{lp}(x, y). \tag{4.4-16}$$

Then, using Eq. (4.4-14), we obtain

$$f_{hb}(x, y) = (A - 1)f(x, y) + f_{hp}(x, y). \tag{4.4-17}$$

This result is based on a highpass rather than a lowpass image. When $A = 1$, high-boost filtering reduces to regular highpass filtering. As A increases past 1, the contribution made by the image itself becomes more dominant.

From Eq. (4.4-14), $F_{hp}(u, v) = F(u, v) - F_{lp}(u, v)$. But, $F_{lp}(u, v) = H_{lp}(u, v)F(u, v)$, where H_{lp} is the transfer function of a lowpass filter. Therefore, unsharp masking can be implemented directly in the frequency domain by using the composite filter

$$H_{hp}(u, v) = 1 - H_{lp}(u, v). \tag{4.4-18}$$

Note that this result agrees with Eq. (4.4-1). Similarly, high-boost filtering can be implemented with the composite filter

$$H_{hb}(u, v) = (A - 1) + H_{hp}(u, v) \tag{4.4-19}$$

with $A \geq 1$. The process consists of multiplying this filter by the (centered) transform of the input image and then taking the inverse transform of the product. Multiplication of the real part of this result by $(-1)^{x+y}$ gives us the high-boost filtered image $f_{hb}(x, y)$ in the spatial domain.

EXAMPLE 4.8: High boost filtering in the frequency domain.

■ Figure 4.29 shows the same sequence as in Fig. 3.43, but using frequency domain computations. Figure 4.29(a) is the input image, and Fig. 4.29(b) is the highpass-filtered image. In order to compare the results in this example with those in Fig. 3.43, we used the Laplacian as the highpass filter, computed using Eq. (4.4-10). We elected not to use a composite filter to make scaling of the Laplacian easier.

The image in Fig. 4.29(c) was obtained using Eq. (4.4-17) with $A = 2$. As in Fig. 3.43(c), this image is sharper, but it still is too dark. Figure 4.29(d) was obtained with $A = 2.7$, which in effect means that the input image was multiplied by 1.7 before the Laplacian was subtracted from it. As in Fig. 3.43, this is an improved result. However, Fig. 4.29(d) is not as sharp as Fig. 4.43(d). The reason for this is that a frequency domain representation of the Laplacian is closer to the mask that excludes the diagonal neighbors [see Fig. 4.27(f)]. We know from the example in Fig. 3.41 that a mask that includes the diagonal neighbors produces slightly sharper results. The differences are not generally noticeable when the details are small (as in the moon example), but they do become evident for images with larger features. ■

Sometimes it is advantageous to accentuate the contribution to enhancement made by the high-frequency components of an image. In this case, we simply multiply a highpass filter function by a constant and add an offset so that the

Thus, high-boost filtering gives us the flexibility to increase the contribution made by the image to the overall enhanced result. This equation may be written as

$$f_{hb}(x, y) = (A - 1)f(x, y) + f(x, y) - f_{lp}(x, y). \tag{4.4-16}$$

Then, using Eq. (4.4-14), we obtain

$$f_{hb}(x, y) = (A - 1)f(x, y) + f_{hp}(x, y). \tag{4.4-17}$$

This result is based on a highpass rather than a lowpass image. When $A = 1$, high-boost filtering reduces to regular highpass filtering. As A increases past 1, the contribution made by the image itself becomes more dominant.

From Eq. (4.4-14), $F_{hp}(u, v) = F(u, v) - F_{lp}(u, v)$. But, $F_{lp}(u, v) = H_{lp}(u, v)F(u, v)$, where H_{lp} is the transfer function of a lowpass filter. Therefore, unsharp masking can be implemented directly in the frequency domain by using the composite filter

$$H_{hp}(u, v) = 1 - H_{lp}(u, v). \tag{4.4-18}$$

Note that this result agrees with Eq. (4.4-1). Similarly, high-boost filtering can be implemented with the composite filter

$$H_{hb}(u, v) = (A - 1) + H_{hp}(u, v) \tag{4.4-19}$$

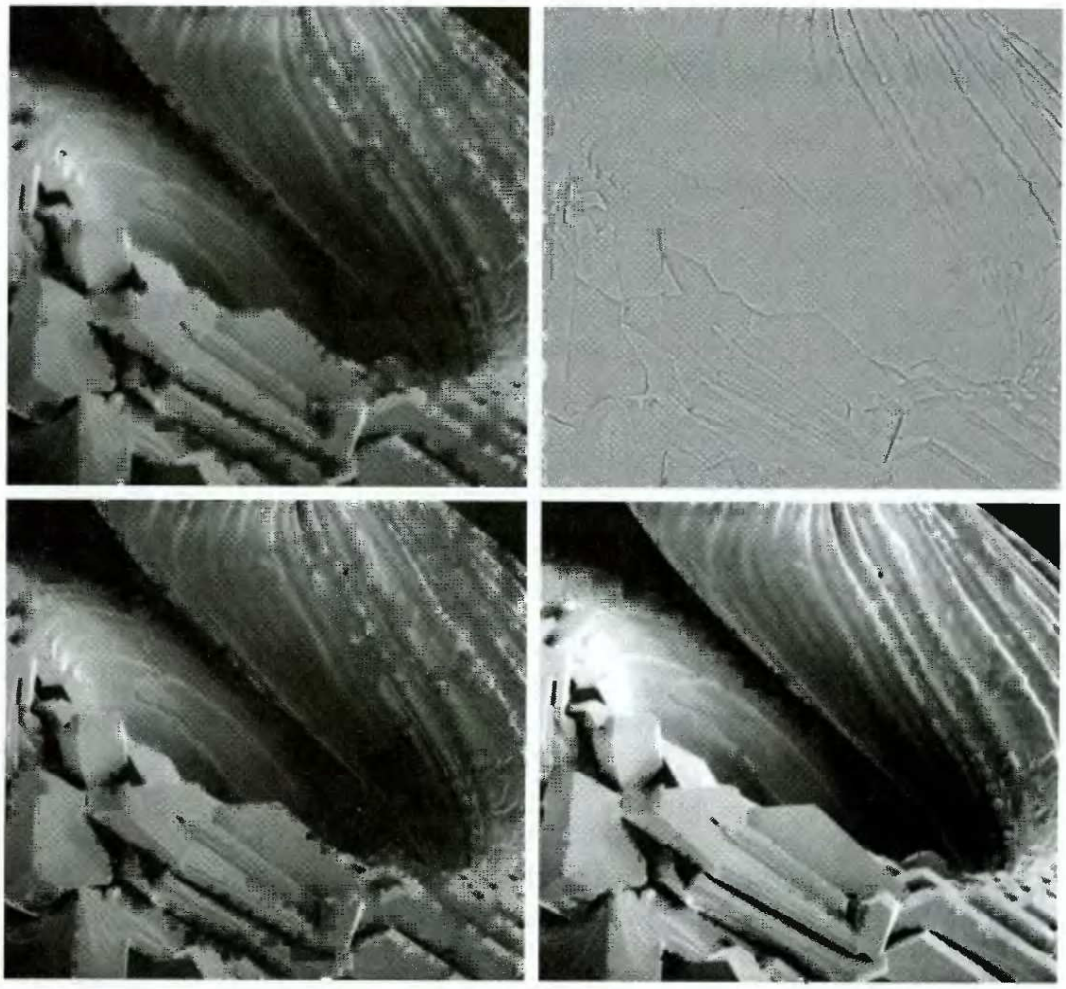
with $A \geq 1$. The process consists of multiplying this filter by the (centered) transform of the input image and then taking the inverse transform of the product. Multiplication of the real part of this result by $(-1)^{x+y}$ gives us the high-boost filtered image $f_{hb}(x, y)$ in the spatial domain.

EXAMPLE 4.8: High boost filtering in the frequency domain.

■ Figure 4.29 shows the same sequence as in Fig. 3.43, but using frequency domain computations. Figure 4.29(a) is the input image, and Fig. 4.29(b) is the highpass-filtered image. In order to compare the results in this example with those in Fig. 3.43, we used the Laplacian as the highpass filter, computed using Eq. (4.4-10). We elected not to use a composite filter to make scaling of the Laplacian easier.

The image in Fig. 4.29(c) was obtained using Eq. (4.4-17) with $A = 2$. As in Fig. 3.43(c), this image is sharper, but it still is too dark. Figure 4.29(d) was obtained with $A = 2.7$, which in effect means that the input image was multiplied by 1.7 before the Laplacian was subtracted from it. As in Fig. 3.43, this is an improved result. However, Fig. 4.29(d) is not as sharp as Fig. 4.43(d). The reason for this is that a frequency domain representation of the Laplacian is closer to the mask that excludes the diagonal neighbors [see Fig. 4.27(f)]. We know from the example in Fig. 3.41 that a mask that includes the diagonal neighbors produces slightly sharper results. The differences are not generally noticeable when the details are small (as in the moon example), but they do become evident for images with larger features. ■

Sometimes it is advantageous to accentuate the contribution to enhancement made by the high-frequency components of an image. In this case, we simply multiply a highpass filter function by a constant and add an offset so that the



a b
c d

FIGURE 4.29 Same as Fig. 3.43, but using frequency domain filtering. (a) Input image. (b) Laplacian of (a). (c) Image obtained using Eq. (4.4-17) with $A = 2$. (d) Same as (c), but with $A = 2.7$. (Original image courtesy of Mr. Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene.)

zero frequency term is not eliminated by the filter. This process, called *high-frequency emphasis*, has a filter transfer function given by

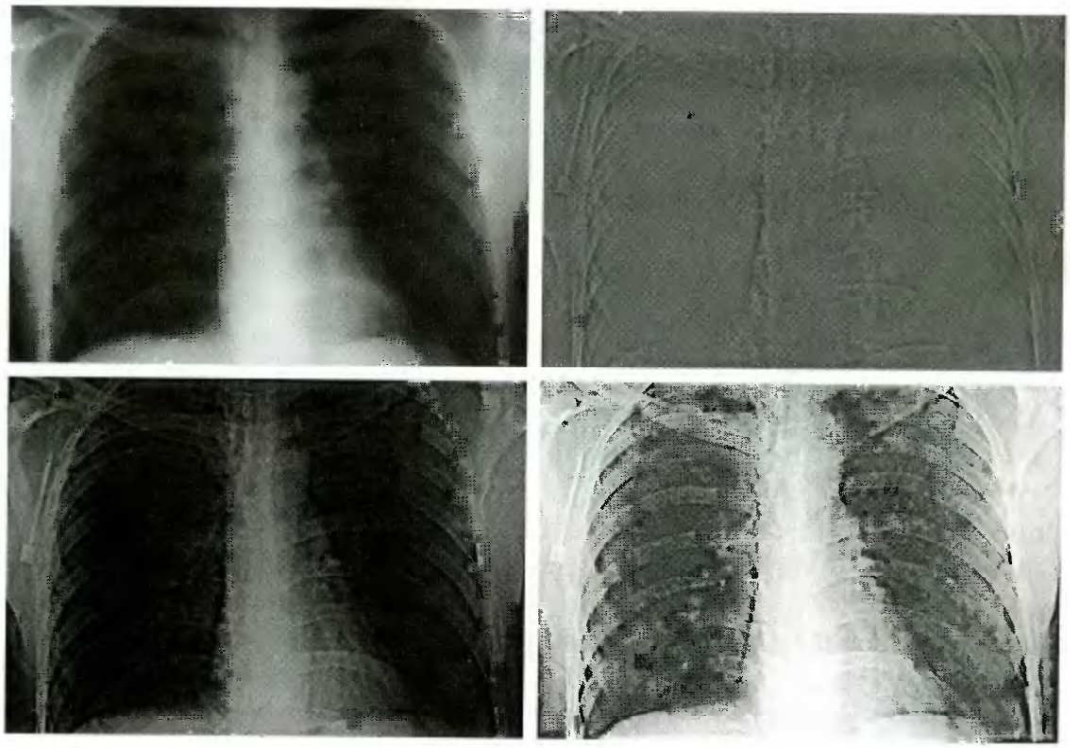
$$H_{\text{hfe}}(u, v) = a + bH_{\text{hp}}(u, v) \tag{4.4-20}$$

where $a \geq 0$ and $b > a$. Typical values of a are in the range 0.25 to 0.5 and typical values of b are in the range 1.5 to 2.0. With reference to Eq. (4.4-17), we see that high-frequency emphasis reduces to high-boost filtering when $a = (A - 1)$ and $b = 1$. When $b > 1$, the high frequencies are emphasized, thus giving this procedure its name.

EXAMPLE 4.9: High-frequency emphasis filtering.

■ Figure 4.30(a) shows a chest X-ray with a narrow range of gray levels. Our main objective is to sharpen the image. X-rays cannot be focused in the same manner that lenses are focused, and the resulting images generally tend to be slightly blurred. Since the gray levels in this particular image are biased toward the dark end of the gray scale, we also take the opportunity here to give an example of how spatial domain processing can be used to complement frequency domain filtering.

Figure 4.30(b) shows the result of highpass filtering using a Butterworth filter of order 2 and a value of D_0 equal to 5% of the image vertical dimension.



a b
c d

FIGURE 4.30 (a) A chest X-ray image. (b) Result of Butterworth highpass filtering. (c) Result of high-frequency emphasis filtering. (d) Result of performing histogram equalization on (c). (Original image courtesy Dr. Thomas R. Gest, Division of Anatomical Sciences, University of Michigan Medical School.)

Highpass filtering is not overly sensitive to this parameter, as long as the radius of the filter is not so small that frequencies near the origin of the transform are passed. As expected, the filtered result is rather featureless, but it shows faintly the principal edges in the image. The advantage of high-emphasis filtering (with $a = 0.5$ and $b = 2.0$ in this case) is shown in the image of Fig. 4.30(c). Although the image is still dark, the gray-level tonality due to the low frequency components was not lost.

As indicated in Section 3.3, an image characterized by gray levels in a narrow range of the gray scale is an ideal candidate for histogram equalization. As shown in Fig. 4.30(d), this indeed was an appropriate method to further enhance the image. Note the clarity of the bone structure and other details that simply are not visible in any of the other three images. The final enhanced image is a little noisy, but this is typical of X-ray images when their gray scale is expanded. The result obtained using a combination of high-frequency emphasis and histogram equalization is superior to the result that would be obtained by using either method alone. ■

4.5 Homomorphic Filtering

The illumination-reflectance model introduced in Section 2.3.4 can be used to develop a frequency domain procedure for improving the appearance of an image by simultaneous gray-level range compression and contrast enhancement. From the discussion in Section 2.3.4, an image $f(x, y)$ can be expressed as the product of illumination and reflectance components:

$$f(x, y) = i(x, y)r(x, y). \tag{4.5-1}$$

Equation (4.5-1) cannot be used directly to operate separately on the frequency components of illumination and reflectance because the Fourier transform of the product of two functions is not separable; in other words,

$$\Im\{f(x, y)\} \neq \Im\{i(x, y)\}\Im\{r(x, y)\}.$$

Suppose, however, that we define

$$\begin{aligned} z(x, y) &= \ln f(x, y) \\ &= \ln i(x, y) + \ln r(x, y). \end{aligned} \tag{4.5-2}$$

Then

$$\begin{aligned} \Im\{z(x, y)\} &= \Im\{\ln f(x, y)\} \\ &= \Im\{\ln i(x, y)\} + \Im\{\ln r(x, y)\} \end{aligned} \tag{4.5-3}$$

or

$$Z(u, v) = F_i(u, v) + F_r(u, v) \tag{4.5-4}$$

where $F_i(u, v)$ and $F_r(u, v)$ are the Fourier transforms of $\ln i(x, y)$ and $\ln r(x, y)$, respectively.

If we process $Z(u, v)$ by means of a filter function $H(u, v)$ then, from Eq. (4.2-27),

$$\begin{aligned} S(u, v) &= H(u, v)Z(u, v) \\ &= H(u, v)F_i(u, v) + H(u, v)F_r(u, v) \end{aligned} \tag{4.5-5}$$

where $S(u, v)$ is the Fourier transform of the result. In the spatial domain,

$$\begin{aligned} s(x, y) &= \Im^{-1}\{S(u, v)\} \\ &= \Im^{-1}\{H(u, v)F_i(u, v)\} + \Im^{-1}\{H(u, v)F_r(u, v)\}. \end{aligned} \tag{4.5-6}$$

By letting

$$i'(x, y) = \Im^{-1}\{H(u, v)F_i(u, v)\} \tag{4.5-7}$$

and

$$r'(x, y) = \Im^{-1}\{H(u, v)F_r(u, v)\}, \tag{4.5-8}$$

Eq. (4.5-6) can be expressed in the form

$$s(x, y) = i'(x, y) + r'(x, y). \tag{4.5-9}$$

Finally, as $z(x, y)$ was formed by taking the logarithm of the original image $f(x, y)$, the inverse (exponential) operation yields the desired enhanced image, denoted by $g(x, y)$; that is,

$$\begin{aligned} g(x, y) &= e^{s(x,y)} \\ &= e^{i'(x,y)} \cdot e^{r'(x,y)} \\ &= i_0(x, y)r_0(x, y) \end{aligned} \tag{4.5-10}$$

where

$$i_0(x, y) = e^{i'(x,y)} \tag{4.5-11}$$

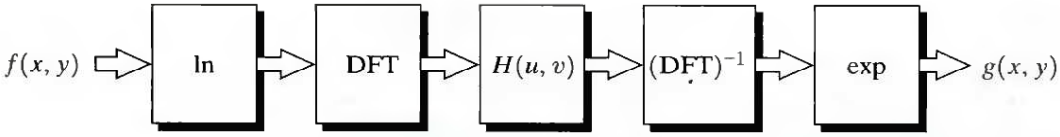


FIGURE 4.31 Homomorphic filtering approach for image enhancement.

and

$$r_0(x, y) = e^{r'(x,y)} \tag{4.5-12}$$

are the illumination and reflectance components of the output image.

The enhancement approach using the foregoing concepts is summarized in Fig. 4.31. This method is based on a special case of a class of systems known as *homomorphic systems*. In this particular application, the key to the approach is the separation of the illumination and reflectance components achieved in the form shown in Eq. (4.5-4). The *homomorphic filter function* $H(u, v)$ can then operate on these components separately, as indicated in Eq. (4.5-5).

The illumination component of an image generally is characterized by slow spatial variations, while the reflectance component tends to vary abruptly, particularly at the junctions of dissimilar objects. These characteristics lead to associating the low frequencies of the Fourier transform of the logarithm of an image with illumination and the high frequencies with reflectance. Although these associations are rough approximations, they can be used to advantage in image enhancement.

A good deal of control can be gained over the illumination and reflectance components with a homomorphic filter. This control requires specification of a filter function $H(u, v)$ that affects the low- and high-frequency components of the Fourier transform in different ways. Figure 4.32 shows a cross section of such a filter. If the parameters γ_L and γ_H are chosen so that $\gamma_L < 1$ and $\gamma_H > 1$, the filter function shown in Fig. 4.32 tends to decrease the contribution made by the low frequencies (illumination) and amplify the contribution made by high frequencies (reflectance). The net result is simultaneous dynamic range compression and contrast enhancement.

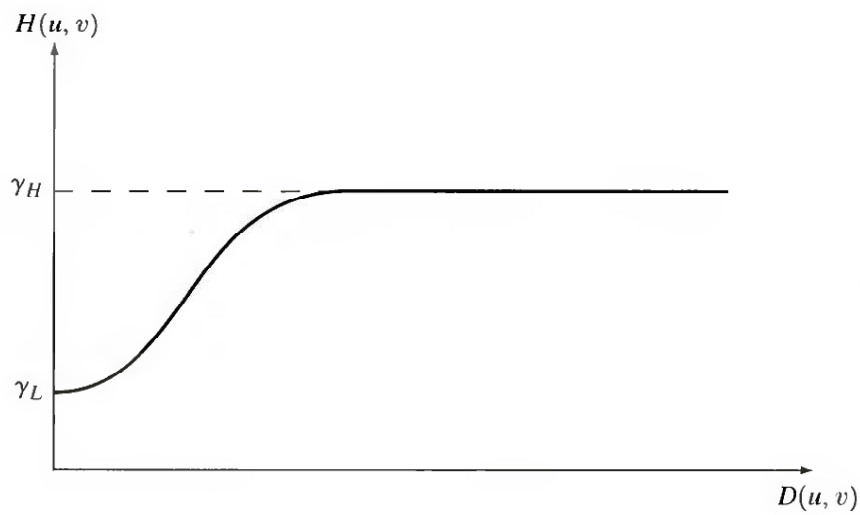
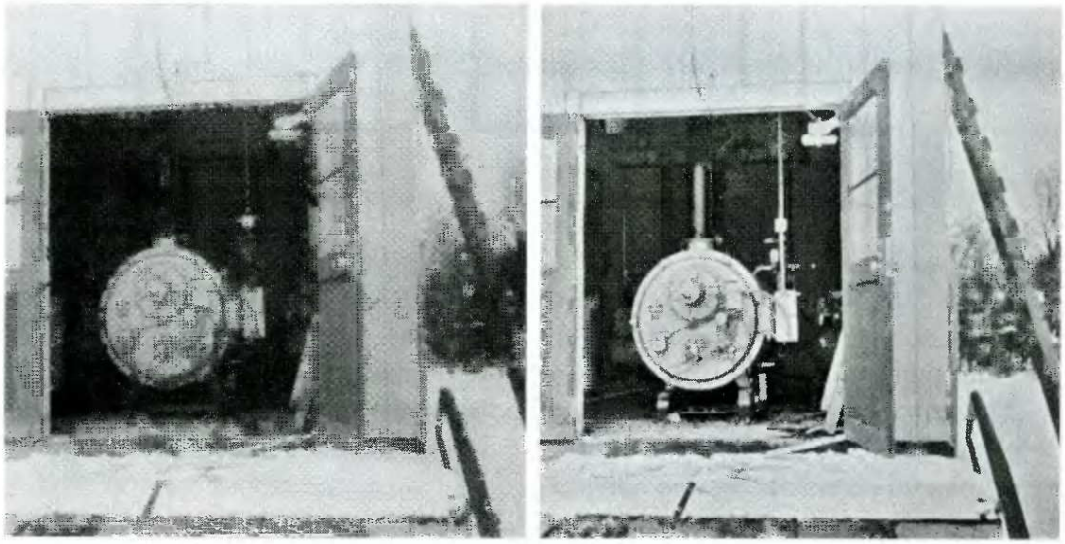


FIGURE 4.32 Cross section of a circularly symmetric filter function. $D(u, v)$ is the distance from the origin of the centered transform.



a b

FIGURE 4.33 (a) Original image. (b) Image processed by homomorphic filtering (note details inside shelter). (Stockham.)

The curve shape shown in Fig. 4.32 can be approximated using the basic form of any of the ideal highpass filters discussed in the previous section. For example, using a slightly modified form of the Gaussian highpass filter gives us

$$H(u, v) = (\gamma_H - \gamma_L)\left[1 - e^{-c\left(D^2(u,v)/D_0^2\right)}\right] + \gamma_L \tag{4.5-13}$$

where $D^2(u, v)$ is given in Eq. (4.3-3) and the constant c has been introduced to control the sharpness of the slope of the filter function as it transitions between γ_L and γ_H. This type of filter is similar to the high-frequency emphasis filter we discussed at the end of Section 4.4.

EXAMPLE 4.10: Enhancement by homomorphic filtering.

Figure 4.33 is typical of the results that can be obtained with the homomorphic filtering function in Fig. 4.32. In the original image shown in Fig. 4.33(a) the details inside the shelter are obscured by the glare from the outside walls. Figure 4.33(b) shows the result of processing this image by homomorphic filtering, with $\gamma_L = 0.5$ and $\gamma_H = 2.0$ in the filter function of Fig. 4.32. A reduction of dynamic range in the brightness, together with an increase in contrast, brought out the details of objects inside the shelter and balanced the gray levels of the outside wall. The enhanced image also is sharper.

4.6 Implementation

In this section we discuss details basic to implementing the Fourier transform. We begin with a summary of some additional properties of the 2-D Fourier transform and conclude with a brief review of the fast Fourier transform (FFT).

4.6.1 Some Additional Properties of the 2-D Fourier Transform

Translation

The Fourier transform pair has the following translation properties:

$$f(x, y)e^{j2\pi(u_0x/M + v_0y/N)} \Leftrightarrow F(u - u_0, v - v_0) \tag{4.6-1}$$

and

$$f(x - x_0, y - y_0) \Leftrightarrow F(u, v)e^{-j2\pi(ux_0/M + vy_0/N)}, \tag{4.6-2}$$

where, as introduced in Eq. (4.2-31), the double arrow is used to designate a Fourier transform pair. When $u_0 = M/2$ and $v_0 = N/2$, it follows that

$$\begin{aligned} e^{j2\pi(u_0 x/M + v_0 y/N)} &= e^{j\pi(x+y)} \\ &= (-1)^{x+y}. \end{aligned}$$

In this case, Eq. (4.6-1) becomes

$$f(x, y)(-1)^{x+y} \Leftrightarrow F(u - M/2, v - N/2) \tag{4.6-3}$$

and, similarly,

$$f(x - M/2, y - N/2) \Leftrightarrow F(u, v)(-1)^{(u+v)}. \tag{4.6-4}$$

We see that Eq. (4.6-3) is the same as Eq. (4.2-21), which we used for centering the transform. These results are based on the variables u and v having values in the range $[0, M - 1]$ and $[0, N - 1]$, respectively. In a computer implementation these variables will run from $u = 1$ to M and $v = 1$ to N, in which case the actual center of the transform will be at $u = (M/2) + 1$ and $v = (N/2) + 1$.

Distributivity and scaling

From the definition of the Fourier transform it follows that

$$\Im[f_1(x, y) + f_2(x, y)] = \Im[f_1(x, y)] + \Im[f_2(x, y)] \tag{4.6-5}$$

and, in general, that

$$\Im[f_1(x, y) \cdot f_2(x, y)] \neq \Im[f_1(x, y)] \cdot \Im[f_2(x, y)]. \tag{4.6-6}$$

In other words, the Fourier transform is distributive over addition, but not over multiplication. Identical comments apply to the inverse Fourier transform. Similarly, for two scalars a and b,

$$af(x, y) \Leftrightarrow aF(u, v) \tag{4.6-7}$$

and

$$f(ax, by) \Leftrightarrow \frac{1}{|ab|} F(u/a, v/b). \tag{4.6-8}$$

Rotation

If we introduce the polar coordinates

$$x = r\cos\theta \qquad y = r\sin\theta \qquad u = \omega\cos\varphi \qquad v = \omega\sin\varphi$$

then $f(x, y)$ and $F(u, v)$ become $f(r, \theta)$ and $F(\omega, \varphi)$, respectively. Direct substitution into the definition of the Fourier transform yields

$$f(r, \theta + \theta_0) \Leftrightarrow F(\omega, \varphi + \theta_0). \tag{4.6-9}$$

This expression indicates that rotating $f(x, y)$ by an angle θ_0 rotates $F(u, v)$ by the same angle. Similarly, rotating $F(u, v)$ rotates $f(x, y)$ by the same angle.

Periodicity and conjugate symmetry

The discrete Fourier transform has the following periodicity properties:

$$F(u, v) = F(u + M, v) = F(u, v + N) = F(u + M, v + N). \quad (4.6\text{-}10)$$

The inverse transform also is periodic:

$$f(x, y) = f(x + M, y) = f(x, y + N) = f(x + M, y + N). \quad (4.6\text{-}11)$$

The idea of conjugate symmetry was introduced in Section 4.2, and is repeated here for convenience:

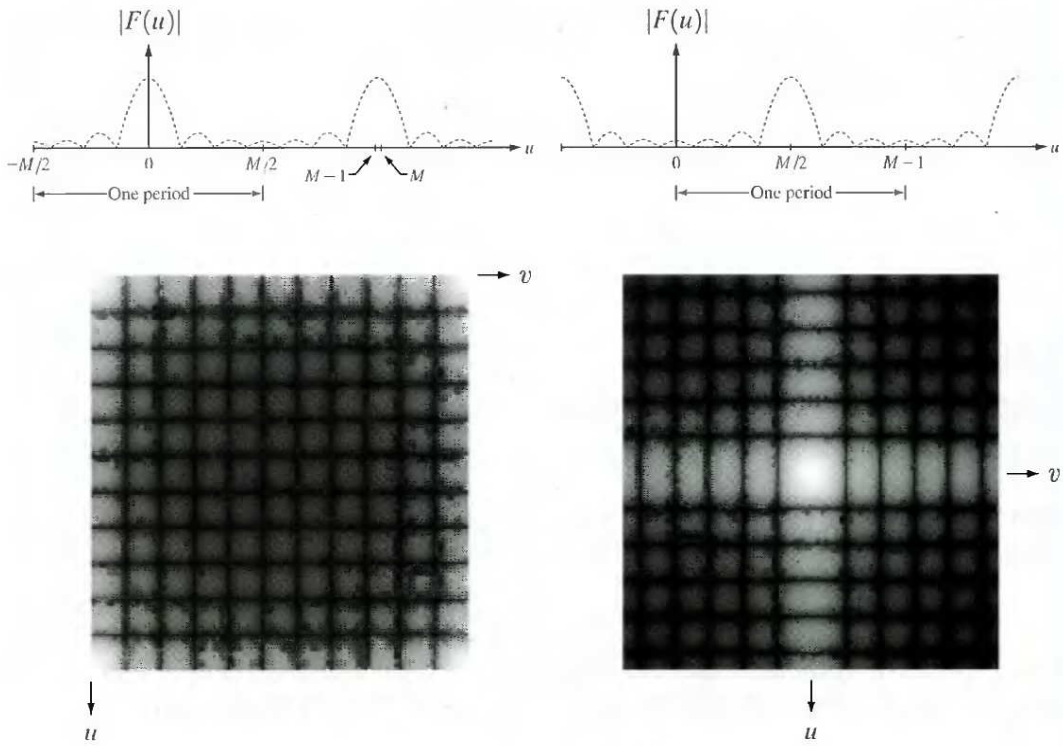
$$F(u, v) = F^*(-u, -v) \quad (4.6\text{-}12)$$

from which it follows that the spectrum also is symmetric about the origin:

$$|F(u, v)| = |F(-u, -v)|. \quad (4.6\text{-}13)$$

The validity of these equations is easily established from Eqs. (4.2-16) and (4.2-17).

The importance of the periodicity property is illustrated in Fig. 4.34(a), which shows the spectrum for a one-dimensional transform $F(u)$ [see Eqs. (4.2-5) and (4.2-10)]. From Eq. (4.6-10), $F(u) = F(u + M)$, from which it follows that $|F(u)| = |F(u + M)|$. Also, because of Eq. (4.6-13), $|F(u)| = |F(-u)|$. The periodicity property indicates that $F(u)$ has a period of length M, and the symmetry property indicates that the spectrum is centered on the origin, as Fig. 4.34(a) shows. This figure and the preceding comments demonstrate that the magnitudes of the transform values from $(M/2) + 1$ to $M - 1$ are reflections of the values in the half period to the left of the origin. Because the discrete Fourier transform has been formulated for values of u in the interval $[0, M - 1]$, the result of this formulation yields two back-to-back half periods in this interval. To



a b
c d

FIGURE 4.34 (a) Fourier spectrum showing back-to-back half periods in the interval $[0, M - 1]$. (b) Shifted spectrum showing a full period in the same interval. (c) Fourier spectrum of an image, showing the same back-to-back properties as (a), but in two dimensions. (d) Centered Fourier spectrum.

display one full period, all that is necessary is to move the origin of the transform to the point $u = M/2$, as Fig. 4.34(b) shows. To do so, we simply multiply $f(x)$ by $(-1)^x$ prior to taking the transform, as indicated in Eq. (4.6-3).

Two-dimensional spectra are analyzed in a similar manner. Figures 4.34(c) and (d) show what a typical spectrum looks like before and after centering (only one period is shown). The center of the transform in this case is at the top, left corner of the picture, and four 2-D periodic components extend "back to back" from the four corners of the image toward its center, which contains the highest-frequency component. By contrast, the centered transform, obtained using Eq. (4.6-3), has the origin (zero frequency components) at the center of Fig. 4.34(d), and its components extend outward from there with conjugate symmetry. Figure 4.34(d) clearly shows that centering the transform not only helps with visualization but, as mentioned several times already in this chapter, it also simplifies filtering.

Separability

The discrete Fourier transform in Eq. (4.2-16) can be expressed in the separable form

$$\begin{aligned} F(u, v) &= \frac{1}{M}\sum_{x=0}^{M-1} e^{-j2\pi ux/M} \frac{1}{N}\sum_{y=0}^{N-1} f(x, y)e^{-j2\pi vy/N} \\ &= \frac{1}{M}\sum_{x=0}^{M-1} F(x, v)e^{-j2\pi ux/M} \end{aligned} \tag{4.6-14}$$

where

$$F(x, v) = \frac{1}{N}\sum_{y=0}^{N-1} f(x, y)e^{-j2\pi vy/N}. \tag{4.6-15}$$

For *each* value of x, and for values of $v = 0, 1, 2, \ldots, N - 1$, this equation is a complete 1-D Fourier transform. In other words, $F(x, v)$ is the Fourier transform along one row of $f(x, y)$. By varying x from 0 to $M - 1$, we compute the Fourier transform along all rows of $f(x, y)$. Thus far, the frequency variable u has remained constant. To complete the 2-D transform we have to vary u from 0 to $M - 1$ in Eq. (4.6-14). After a little thought, it becomes evident that this involves computing the 1-D transform along each *column* of $F(x, v)$. This is an important result. It tells us that we can compute the 2-D transform by first computing a 1-D transform along each row of the input image, and then computing a 1-D transform along each column of this intermediate result. The same comments hold if we reverse the order of computation: columns first, followed by rows. The procedure is summarized in Fig. 4.35.

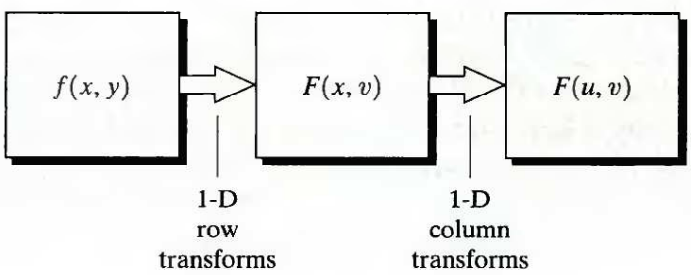


FIGURE 4.35 Computation of the 2-D Fourier transform as a series of 1-D transforms.

A similar development applies to computing the 2-D inverse Fourier transform. We first compute 1-D inverse transforms along each row of $F(u, v)$ and then compute inverse 1-D transforms along each column of the intermediate result. As shown in the following section, it is possible to implement the inverse transform using a 1-D *forward* Fourier transform algorithm.

4.6.2 Computing the Inverse Fourier Transform Using a Forward Transform Algorithm

As noted in the previous section, 2-D Fourier transforms can be computed via the application of 1-D transforms. The 1-D Fourier transform pair was defined in Section 4.2.1. We repeat the two equations here for convenience:

$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)e^{-j2\pi ux/M} \tag{4.6-16}$$

for $u = 0, 1, 2, \ldots, M-1$, and

$$f(x) = \sum_{u=0}^{M-1} F(u)e^{j2\pi ux/M} \tag{4.6-17}$$

for $x = 0, 1, 2, \ldots, M-1$. Taking the complex conjugate of Eq. (4.6-17) and dividing both sides by M yields

$$\frac{1}{M}f^*(x) = \frac{1}{M}\sum_{u=0}^{M-1} F^*(u)e^{-j2\pi ux/M}. \tag{4.6-18}$$

Comparing this result with Eq. (4.6-16) shows that the right side of Eq. (4.6-18) is in the form of the forward Fourier transform. Therefore, inputting $F^*(u)$ into an algorithm designed to compute the forward transform gives the quantity $f^*(x)/M$. Taking the complex conjugate and multiplying by M yields the desired inverse $f(x)$. A similar analysis for two variables yields:

$$\frac{1}{MN}f^*(x, y) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F^*(u, v)e^{-j2\pi(ux/M+vy/N)}, \tag{4.6-19}$$

which is in the form of a 2-D forward Fourier transform. If $f(x)$ or $f(x, y)$ are real functions (e.g., an image), then the complex conjugate on the left of Eq. (4.6-19) is unnecessary; we simply take the real part of the result, ignoring the parasitic complex terms that are typical in most Fourier transform computations.

Computation of the 2-D transform by successive passes of the 1-D transform is a frequent source of confusion when the technique we have just developed is used to obtain the inverse. Keep in mind the procedure outlined in the previous section, and avoid being misled by Eq. (4.6-18). In other words, when a 1-D algorithm is used to compute the 2-D inverse, we do not compute the complex conjugate after processing each row or column. Instead, the function $F^*(u, v)$ is treated as if it were $f(x, y)$ in the forward, 2-D transform procedure summarized in Fig. 4.35. The complex conjugate (or real part, if applicable) of the result, multiplied by MN, yields the proper inverse $f(x, y)$. We emphasize that the preceding comments regarding the constants M and N are based on the definition of the discrete Fourier transform that has all constants associated

with the forward transform. As indicated in Sections 4.2.1 and 4.2.2, it is not unusual to encounter the constants distributed in a different way between the forward and inverse transforms. Therefore, in order to avoid being off by a scale factor, care must be exercised in the placement of the constants when computing the inverse transform if these constants are distributed differently from the way that we do in this book.

4.6.3 More on Periodicity: the Need for Padding

It was explained in Section 4.2.4 that, based on the convolution theorem, multiplication in the frequency domain is equivalent to convolution in the spatial domain, and vice versa. When working with discrete variables and the Fourier transform, we need to keep in mind the periodicity of the various functions involved (Section 4.6.1). Although it may not be intuitive, this periodicity is a mathematical byproduct of the way in which the discrete Fourier transform pair is defined. Periodicity is part of the process, and it cannot be ignored.

Figure 4.36 illustrates the significance of periodicity. The left column of this figure shows convolution computed using the 1-D version of Eq. (4.2-30):

$$f(x) * h(x) = \frac{1}{M}\sum_{m=0}^{M-1} f(m)h(x-m). \tag{4.6-20}$$

We also take the opportunity here to explain convolution in a little more detail. To simplify the notation, simple numbers instead of general symbols are used for the height and length of the functions. Figures 4.36(a) and (b) show the two functions we wish to convolve. Each function consists of 400 points. The first step in convolution is to mirror (flip) one of the functions about the origin. In this case this was done to the second function, which is shown as $h(-m)$ in Fig. 4.36(c). The next step is to "slide" $h(-m)$ past $f(m)$. This is done by adding a constant, x, to $h(-m)$; that is, we form $h(x-m)$, as shown in Fig. 4.36(d). Note that this is only *one* displacement value. This simple step is a frequent source of confusion when first encountered. It helps to remember that this is precisely what convolution is all about. In other words, to perform convolution we flip one of the functions and slide it past the other. At each displacement (each value of x) the entire summation in Eq. (4.6-20) is carried out. This summation is nothing more than the sum of products of f and h at a given displacement. The displacement x ranges over all values required to completely slide h past f. Figure 4.36(e) shows the result of completely sliding h past f and computing Eq. (4.6-20) at each value of x. In this case x had to range for 0 to 799 for $h(x-m)$ to slide completely past f. This figure is the convolution of the two functions. Keep clearly in mind that the variable in convolution is x.

We know from the convolution theorem introduced in Section 4.2 [see Eq. (4.2-31)] that we can obtain exactly the same result given in Eq. (4.6-20) by taking the inverse Fourier transform of the product $F(u)H(u)$. However, we also know from the discussion on periodicity earlier in this section that the discrete Fourier transform automatically takes the input functions as periodic. In other words, using the DFT allows us to perform convolution in the frequency domain, but the functions are treated as periodic, with a period equal to the length of the functions.

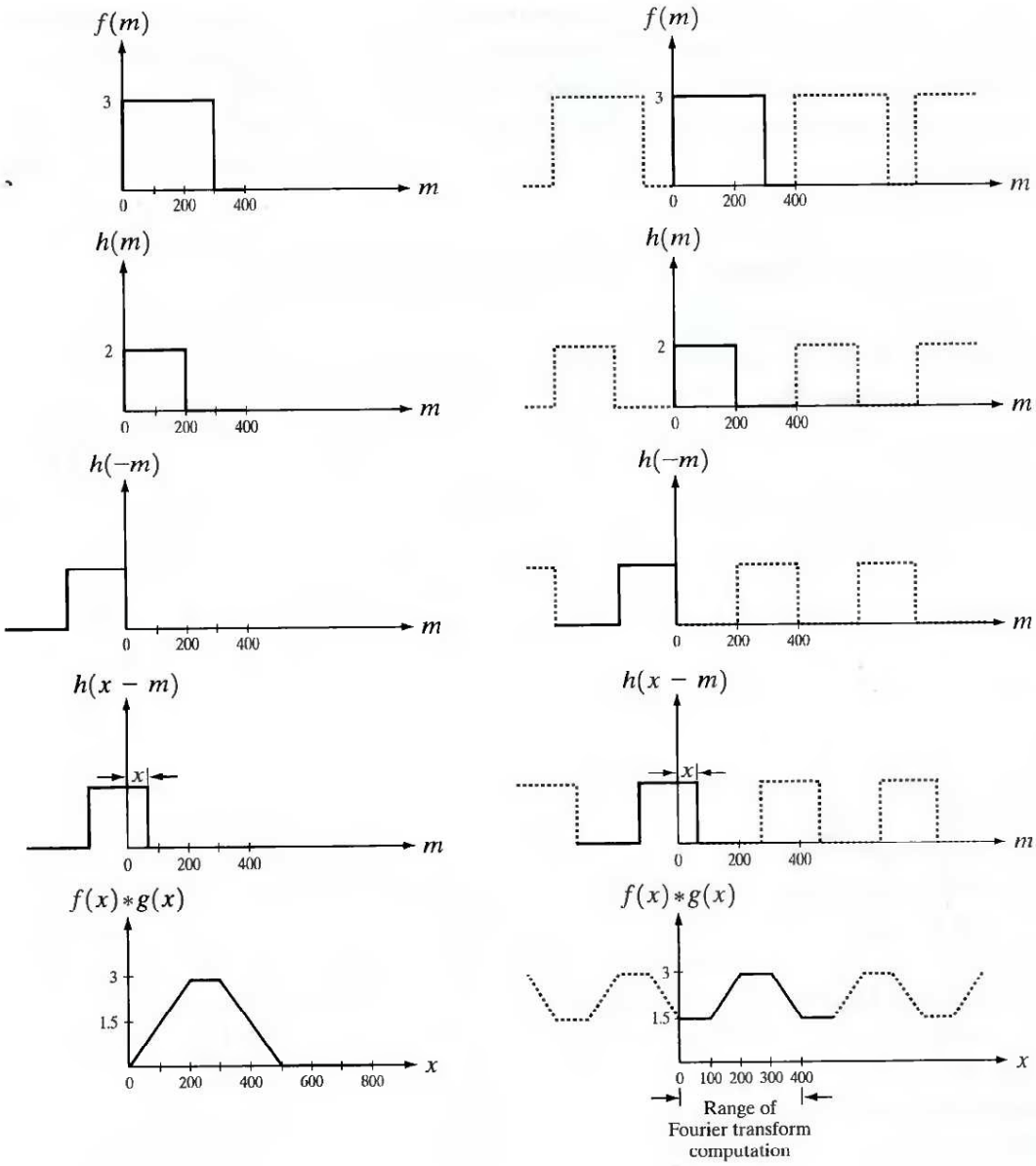


FIGURE 4.36 Left: convolution of two discrete functions. Right: convolution of the same functions, taking into account the implied periodicity of the DFT. Note in (j) how data from adjacent periods corrupt the result of convolution.

We can examine the implications of this periodicity with the aid of the right column of Fig. 4.36. Figure 4.36(f) is the same as Fig. 4.36(a), but with periods of the same function extending infinitely in both directions (extended sections are shown dashed). The same applies to Figs. 4.36(g) through (i). We now perform convolution by sliding $h(x - m)$ past $f(m)$. The sliding is accomplished by varying x, as before. However, now the periodic extensions of $h(x - m)$ introduce values that were not there in our computations on the left side of Fig. 4.36. For example, when $x = 0$ in Fig. 4.36(i), we see that part of the first extended period to the right of $h(x - m)$ lies inside the part of $f(m)$ in Fig. 4.36(f) that starts at the origin (shown solid). As $h(x - m)$ slides to the right, the section that was inside $f(m)$ starts to move out to the right, but it is replaced by an identical section from the left side of $h(x - m)$. This causes the convolution to have a constant value, as shown in the segment $[0, 100]$ in Fig. 4.36(j). The seg-

ment from 100 through 400 is correct, but periodicity starts again, thus causing part of the tail of the convolution function to be lost, as can be seen by comparing the solid lines in Figs. 4.36(j) and 4.36(e).

In the frequency domain the procedure would be to compute the Fourier transforms of the functions in Figs. 4.36(a) and (b). According to the convolution theorem, the two transforms would then be multiplied and the inverse Fourier transform taken. The result would be the 400 points comprising the convolution shown in solid in Fig. 4.36(j). This simple illustration shows that failure to handle the periodicity issue properly will give incorrect results if the convolution function is obtained using the Fourier transform. The result will have erroneous data at the beginning and have missing data at the end.

The solution to this problem is straightforward. Assume that f and h consist of A and B points, respectively. We append zeros to both functions so that they have identical periods, denoted by P. This procedure yields *extended*, or *padded*, functions given by

$$f_e(x) = \begin{cases} f(x) & 0 \le x \le A - 1 \\ 0 & A \le x \le P \end{cases} \tag{4.6-21}$$

and

$$g_e(x) = \begin{cases} g(x) & 0 \le x \le B - 1 \\ 0 & B \le x \le P. \end{cases} \tag{4.6-22}$$

It can be shown (Brigham [1988]) that, unless we choose $P \ge A + B - 1$, the individual periods of the convolution will overlap. We already saw in Fig. 4.36 the result of this phenomenon, which is commonly referred to as *wraparound error*. If $P = A + B - 1$, the periods will be adjacent. If $P > A + B - 1$, the periods will be separated, with the degree of separation being equal to the difference between P and $A + B - 1$.

The results obtained after extending the functions in Fig. 4.36(a) and (b) are shown in Figs. 4.37(a) and (b). In this case we chose $P = A + B - 1$ (799), so we know that the periods of the convolution will be adjacent. Following a procedure identical to the previous explanation, we arrive at the convolution function shown in Fig. 4.37(e). One period of this result is identical to Fig. 4.36(e), which we know to be correct. Thus, if we wanted to compute the convolution in the frequency domain, we would (1) obtain the Fourier transform of the two extended sequences (each of which is 800 points long); (2) multiply the two transforms; and (3) compute the inverse Fourier transform. The result would be the correct 800-point convolution function shown in the period highlighted in Fig. 4.37(e).

Extension of these concepts to 2-D functions follows the same line of reasoning. Suppose that we have two images $f(x, y)$ and $h(x, y)$ of sizes $A \times B$ and $C \times D$, respectively. As in the 1-D case, these arrays must be assumed periodic with some period P in the x-direction and Q in the y-direction. Wraparound error in 2-D convolution is avoided by choosing

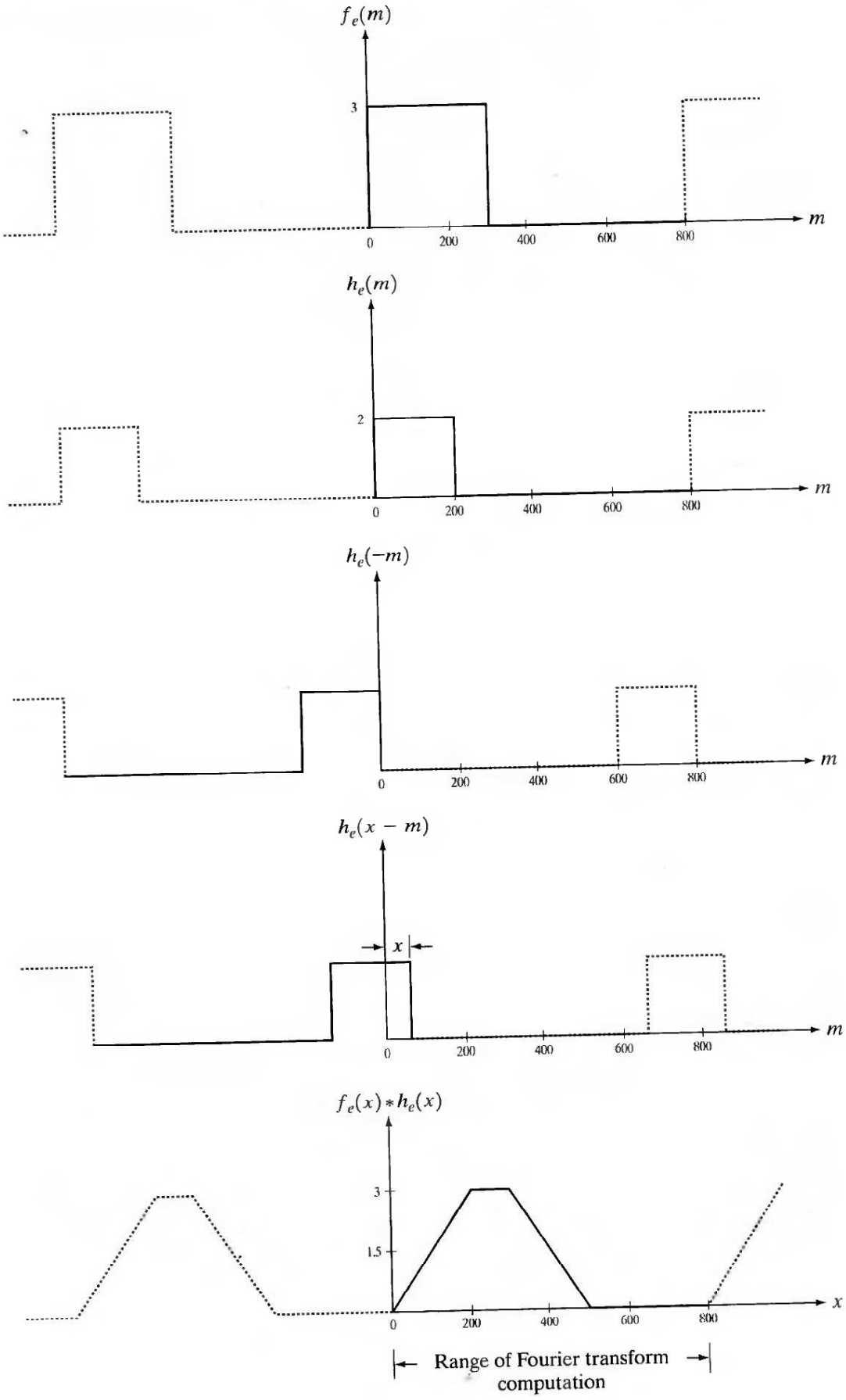
$$P \ge A + C - 1 \tag{4.6-23}$$

and

$$Q \ge B + D - 1. \tag{4.6-24}$$

a
b
c
d
e

FIGURE 4.37 Result of performing convolution with extended functions. Compare Figs. 4.37(e) and 4.36(e).



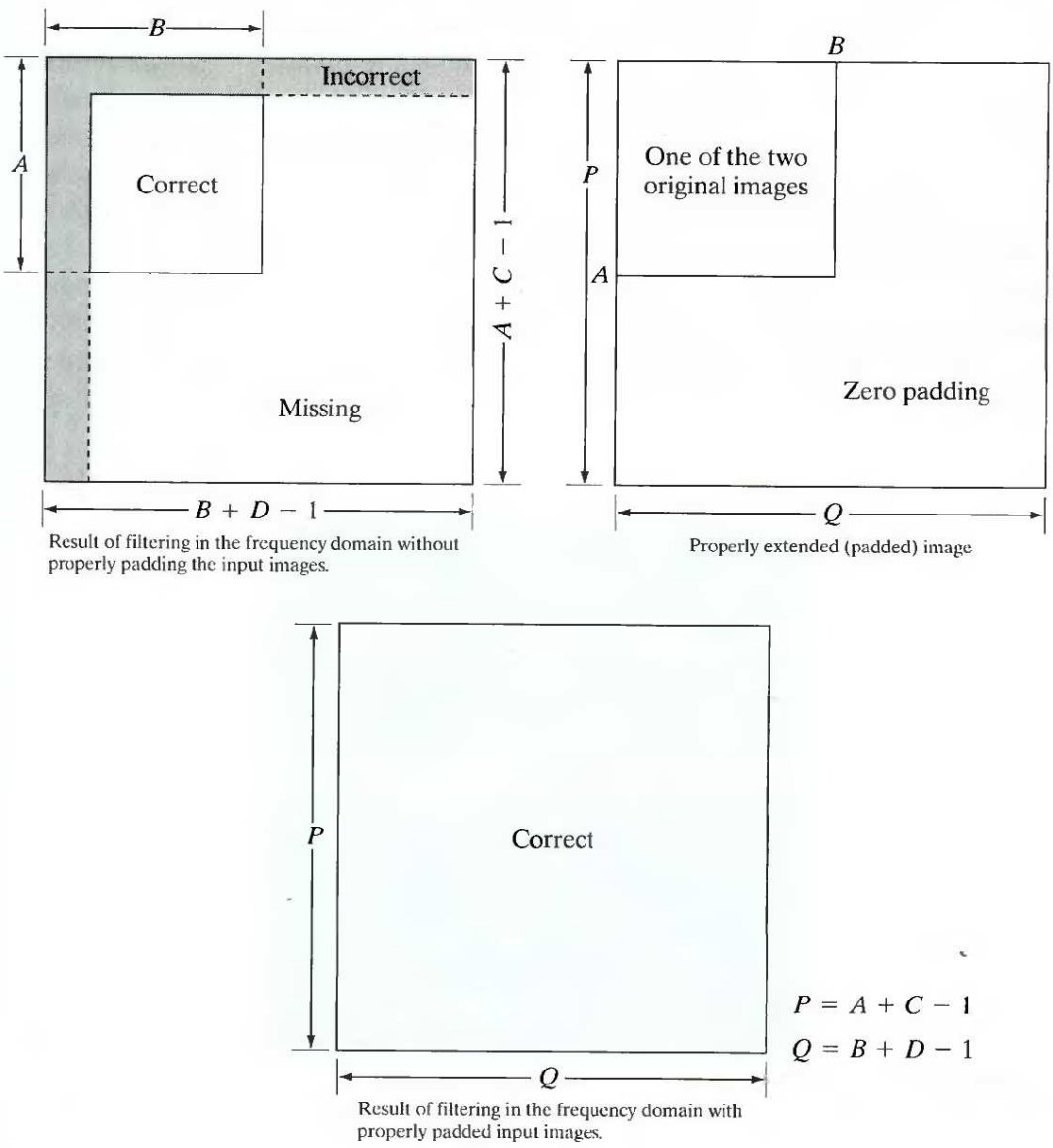
The periodic sequences are formed by extending $f(x, y)$ and $h(x, y)$ as follows:

$$f_e(x, y) = \begin{cases} f(x, y) & 0 \le x \le A - 1 \quad \text{and} \quad 0 \le y \le B - 1 \\ 0 & A \le x \le P \quad \text{or} \quad B \le y \le Q \end{cases} \tag{4.6-25}$$

and

$$h_e(x, y) = \begin{cases} h(x, y) & 0 \le x \le C - 1 \quad \text{and} \quad 0 \le y \le D - 1 \\ 0 & C \le x \le P \quad \text{or} \quad D \le y \le Q \end{cases} \tag{4.6-26}$$

The issue of padding is central to filtering. When we implement any of the frequency domain filters discussed in this chapter, we do it by multiplying the filter transfer function by the transform of the image we wish to process. By the convolution theorem, we know that this is the same as convolving the spatial representation of the filter with the image. Thus, unless proper padding is implemented, the results will be erroneous. This is illustrated in Fig. 4.38. For the sake of simplicity in the figure, we assume that f and h are square and that they are



a b
c

FIGURE 4.38 Illustration of the need for function padding. (a) Result of performing 2-D convolution without padding. (b) Proper function padding. (c) Correct convolution result.

both of the same size, where h is the inverse DFT of the filter $H(u, v)$. Figure 4.38(a) shows what the result of filtering would be if the images were not padded. This is the generic result we would obtain if we computed the Fourier transform of an input image that was not padded, multiplied it by a filter function of the same size (also not padded) and computed the inverse transform. The result would be of size $A \times B$, the same as the input image, as shown in the top, left quadrant in Fig. 4.38(a). As in the 1-D case, the leading edges of the image would contain erroneous data induced by periodicity (shown shaded), and there would be missing data at the trailing edges. By properly padding the input image *and* the filter function as shown in Fig. 4.38(b), the result would be a correctly filtered image of size $P \times Q$, as shown in Fig. 4.38(c). This image is twice the size of the original in both directions, and thus has four times as many pixels. However, as will be shown shortly, the area of interest typically is cropped out of this larger image.

It is important to note that the approach just described calls for the frequency domain filter function to be multiplied by $(-1)^{u+v}$, inverse transformed, padded with 0's, and then forward transformed. All other aspects of filtering are as described in Section 4.2.3. Note also that the inverse Fourier transform of the filter has both real and imaginary parts. Although the imaginary components of the filters with which we deal in this book typically are many orders of magnitude smaller than the real components, it is not good practice in general to ignore imaginary components in intermediate Fourier computations. Thus, both real and imaginary components are padded prior to generating the padded frequency domain filter via computation of the forward transform.

Figure 4.39 shows the padded spatial representation (only the real part is shown) of the ideal lowpass filter used to generate Fig. 4.12(c). The padded area of 0's is shown in black. An ideal lowpass filter was selected for illustration because it has the most visible "structure" in the spatial domain. The padding used

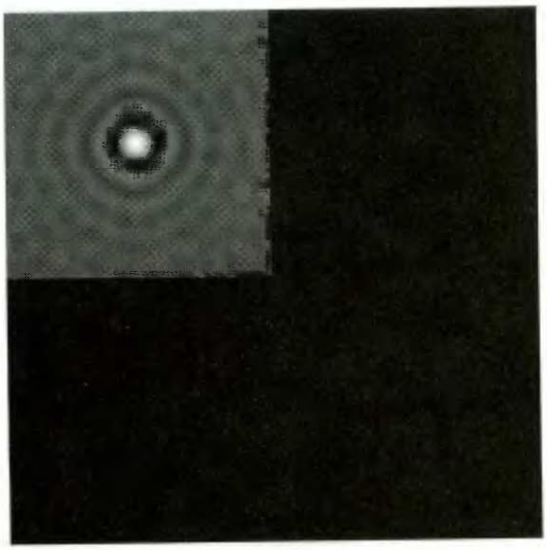


FIGURE 4.39 Padded lowpass filter in the spatial domain (only the real part is shown).

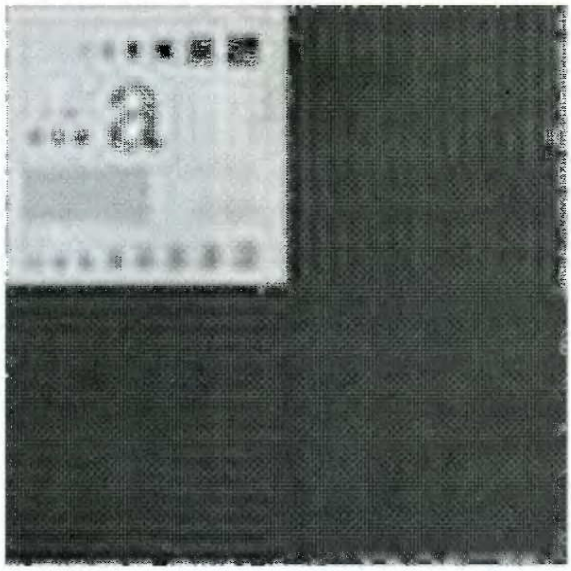


FIGURE 4.40 Result of filtering with padding. The image is usually cropped to its original size since there is little valuable information past the image boundaries.

was the minimum size required, which, when images and filters are squares of the same size, simply doubles the size in both dimensions.

Figure 4.40 shows the result of filtering with padded functions using the approach just discussed. It is easy to visualize how convolving the filter in Fig. 4.39 with a padded version of Fig. 4.12(a) would generate Fig. 4.40. It also is evident in this case that three-quarters of the result contain no valuable information, so cropping back to the original image gives the desired filtered result. We are assured by using padding that the cropped image is free of wraparound error.

4.6.4 The Convolution and Correlation Theorems

Convolution was introduced in Section 4.2.4 and its implementation was discussed in additional detail in Section 4.6.3. We repeat it briefly here to facilitate comparison with a similar process called *correlation*. The discrete convolution of two functions $f(x, y)$ and $h(x, y)$ of size $M \times N$ is denoted by $f(x, y) * h(x, y)$ and is defined by the expression

$$f(x, y) * h(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n)h(x - m, y - n). \quad (4.6\text{-}27)$$

From the discussion in Section 4.2.4, we know that the convolution theorem consists of the following relationships between the two functions and their Fourier transforms:

$$f(x, y) * h(x, y) \Leftrightarrow F(u, v)H(u, v) \quad (4.6\text{-}28)$$

and

$$f(x, y)h(x, y) \Leftrightarrow F(u, v) * H(u, v). \quad (4.6\text{-}29)$$

The correlation of two functions $f(x, y)$ and $h(x, y)$ is defined as

$$f(x, y) \circ h(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f^*(m, n) h(x + m, y + n) \quad (4.6\text{-}30)$$

where f^* denotes the complex conjugate of f. We normally deal with real functions (images), in which case $f^* = f$. The correlation function has exactly the same *form* as the convolution function given in Eq. (4.6-27), with the exception of the complex conjugate and the fact that the second term in the summation has positive instead of negative signs. This means that h is not mirrored about the origin. Everything else in the implementation of correlation is identical to convolution, including the need for padding.

Given the similarity of convolution and correlation, it is not surprising that there is a *correlation theorem*, analogous to the convolution theorem. Let $F(u, v)$ and $H(u, v)$ denote the Fourier transforms of $f(x, y)$ and $h(x, y)$, respectively. One-half of the correlation theorem states that spatial correlation, $f(x, y) \circ h(x, y)$, and the frequency domain product, $F^*(u, v)H(u, v)$, constitute a Fourier transform pair. This result, formally stated as

$$f(x, y) \circ h(x, y) \Leftrightarrow F^*(u, v)H(u, v), \quad (4.6\text{-}31)$$

indicates that correlation in the spatial domain can be obtained by taking the inverse Fourier transform of the product $F^*(u, v)H(u, v)$, where F^* is the complex conjugate of F. An analogous result is that correlation in the frequency domain reduces to multiplication in the spatial domain; that is,

$$f^*(x, y)h(x, y) \Leftrightarrow F(u, v) \circ H(u, v). \quad (4.6\text{-}32)$$

These two results comprise the correlation theorem. It is assumed that all functions have been properly extended by padding.

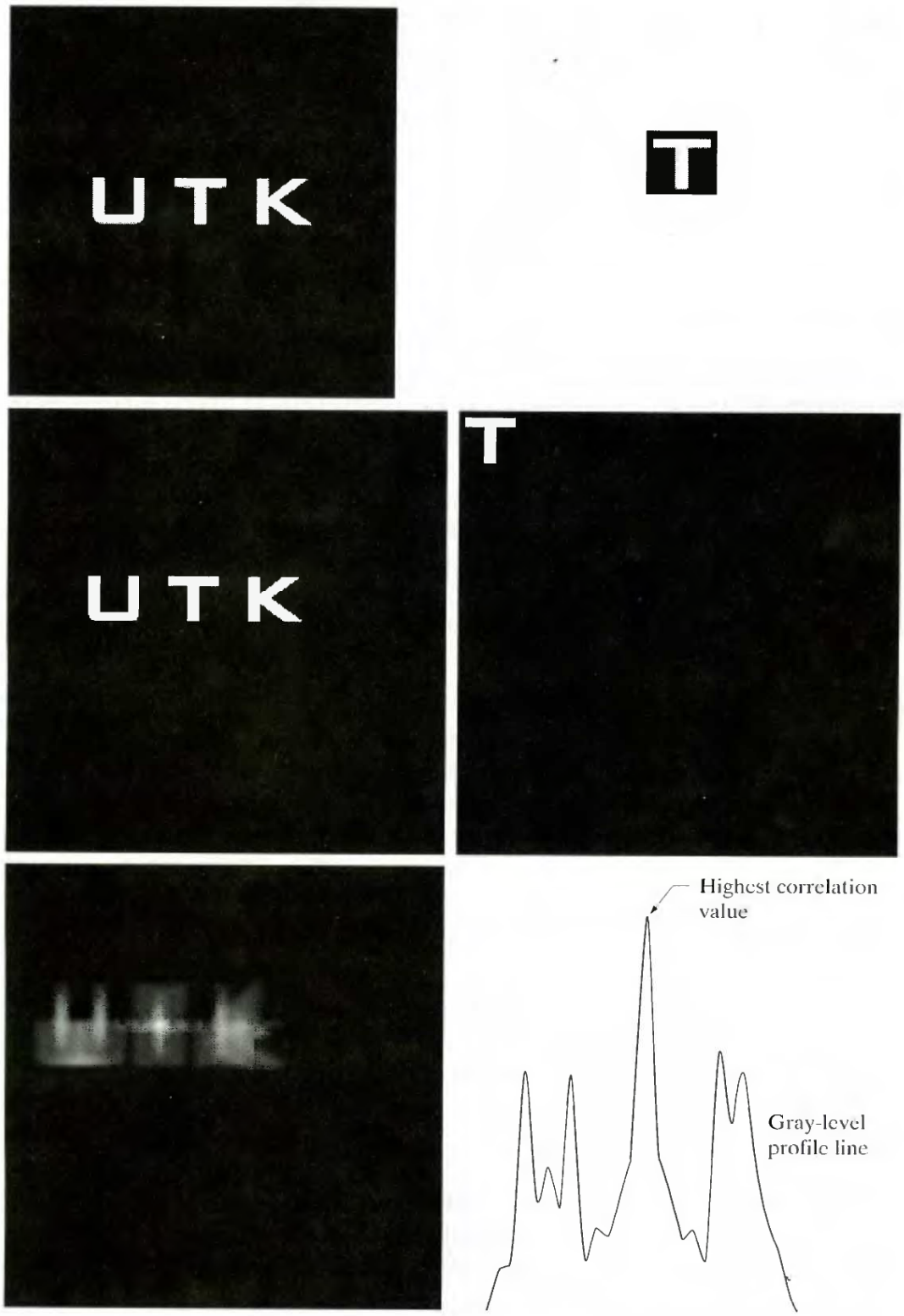
As we know by now, convolution is the tie between filtering in the spatial and frequency domains. The principal use of correlation is for matching. In *matching*, $f(x, y)$ is an image containing objects or regions. If we want to determine whether f contains a particular object or region in which we are interested, we let $h(x, y)$ be that object or region (we normally call this image a *template*). Then, if there is a match, the correlation of the two functions will be maximum at the location where h finds a correspondence in f. Preprocessing, like scaling and alignment, is necessary in most practical applications, but the bulk of the process is performing the correlation.

Finally, we point out that the term *cross correlation* often is used in place of the term *correlation* to clarify that the images being correlated are different. This is as opposed to *autocorrelation*, in which both images are identical. In the latter case, we have the *autocorrelation theorem*, which follows directly from Eq. (4.6-31):

$$f(x, y) \circ f(x, y) \Leftrightarrow |F(u, v)|^2. \quad (4.6\text{-}33)$$

On the right side, we used the fact that the product of a complex quantity and its complex conjugate is the magnitude of the complex quantity squared. In words, this result states that the Fourier transform of the spatial autocorrelation is the power spectrum defined in Eq. (4.2-20). Similarly,

$$|f(x, y)|^2 \Leftrightarrow F(u, v) \circ F(u, v). \quad (4.6\text{-}34)$$



a b
c d
e f

FIGURE 4.41 (a) Image. (b) Template. (c) and (d) Padded images. (e) Correlation function displayed as an image. (f) Horizontal profile line through the highest value in (e), showing the point at which the best match took place.

EXAMPLE 4.11: Image correlation.

■ Figure 4.41 shows a simple illustration of image padding and correlation. Figure 4.41(a) is the image and Fig. 4.41(b) is the template. The image and template are of size 256×256 and 38×42 pixels, respectively. In this case, $A = B = 256$, $C = 38$, and $D = 42$. This gives the minimum values for the extended functions: $P = A + C - 1 = 293$ and $Q = B + D - 1 = 297$. We

chose equal padding dimensions of 298 × 298. The padded images are shown in Figs. 4.41(c) and (d). The spatial correlation of the two padded images is displayed as an image in Fig. 4.41(e). As indicated in Eq. (4.6-31), the correlation function was obtained by computing the transforms of the padded images, taking the complex conjugate of one of them (we chose the template), multiplying the two transforms, and computing the inverse DFT. It is left as an exercise for the reader (Problem 4.23) to discuss what Fig. 4.41(e) would look like if we had taken the complex conjugate of the other transform instead.

As expected, we see in Fig. 4.41(e) that the highest value of the correlation function occurs at the point where the template is exactly on top of the "T" in the image. As in convolution, it is important to keep in mind that the variables in the correlation function in the spatial domain are *displacements*. For example, the top, left corner of Fig. 4.41(e) corresponds to zero displacement of one function with respect to the other. The value of each pixel in Fig. 4.41(e) is the value of the correlation function at *one* location of displacement; that is, for one specific value of the pair (x, y) in Eq. (4.6-30). Also, we note that the correlation function has the same dimensions as the padded images. Finally, Fig. 4.41(f) shows a horizontal gray-level profile passing through the highest value in Fig. 4.41(e). This figure simply confirms that the highest peak in the correlation function is located at the point where the best match of the template and the image occurs. ■

4.6.5 Summary of Properties of the 2-D Fourier Transform

All the properties of the Fourier transform discussed in this chapter are summarized in Table 4.1. A footnote identifies the items requiring that functions be padded in order to avoid incorrect results. As before, the double arrows are used to denote that the expressions form a Fourier transform pair. That is, the expression on the right of the double arrows is obtained by taking the forward Fourier transform of the expression on the left; the expression on the left is obtained by taking the inverse Fourier transform of the expression on the right.

4.6.6 The Fast Fourier Transform

As indicated in Section 4.1, one of the main reasons that the DFT has become an essential tool in signal processing was the development of the fast Fourier transform (FFT). Computing the 1-D Fourier transform of M points using Eq. (4.2-5) directly requires on the order of M^2 multiplication/addition operations. The FFT accomplishes the same task on the order of $M \log_2 M$ operations. If, for example, $M = 1024$, the brute-force method will require approximately 10^6 operations, while the FFT will require approximately 10^4 operations. This is a computational advantage of 100 to 1. If this advantage does not seem significant, imagine being able to complete a given project in one year as opposed to 100. This is the difference between possible and practically impossible. And the story gets better. The bigger the problem, the greater the computational advantage. If, for instance, $M = 8192\ (2^{13})$, the computational advantage grows to 600 to 1. These types of numbers are great motivators for wanting to learn more about how an FFT algorithm works. In this section we take a look at the derivation of a fundamental decomposition of the DFT that leads to the FFT. The focus is on the FFT of one

variable. As indicated in Section 4.6.1, the 2-D Fourier transform can be obtained by successive passes of a 1-D transform algorithm.

The FFT algorithm developed in this section is based on the so-called *successive doubling method*. For notational convenience we express Eq. (4.2-5) in the form

$$F(u) = \frac{1}{M}\sum_{x=0}^{M-1} f(x)W_M^{ux} \tag{4.6-35}$$

where

$$W_M = e^{-j2\pi/M} \tag{4.6-36}$$

and M is assumed to be of the form

$$M = 2^n \tag{4.6-37}$$

with n being a positive integer. Hence, M can be expressed as

$$M = 2K \tag{4.6-38}$$

with K also being a positive integer. Substitution of Eq. (4.6-38) into Eq. (4.6-35) yields

$$\begin{aligned} F(u) &= \frac{1}{2K}\sum_{x=0}^{2K-1} f(x)W_{2K}^{ux} \\ &= \frac{1}{2}\left[\frac{1}{K}\sum_{x=0}^{K-1} f(2x)W_{2K}^{u(2x)} + \frac{1}{K}\sum_{x=0}^{K-1} f(2x+1)W_{2K}^{u(2x+1)}\right]. \end{aligned} \tag{4.6-39}$$

However, it can be shown using Eq. (4.6-36) that $W_{2K}^{2uk} = W_K^{ux}$, so Eq. (4.6-39) can be expressed as

$$F(u) = \frac{1}{2}\left[\frac{1}{K}\sum_{x=0}^{K-1} f(2x)W_{2K}^{ux} + \frac{1}{K}\sum_{x=0}^{K-1} f(2x+1)W_K^{ux}W_{2K}^{u}\right]. \tag{4.6-40}$$

Defining

$$F_{\text{even}}(u) = \frac{1}{K}\sum_{x=0}^{K-1} f(2x)W_K^{ux} \tag{4.6-41}$$

for $u = 0, 1, 2, \ldots, K-1$, and

$$F_{\text{odd}}(u) = \frac{1}{K}\sum_{x=0}^{K-1} f(2x+1)W_K^{ux} \tag{4.6-42}$$

for $u = 0, 1, 2, \ldots, K-1$, reduces Eq. (4.6-40) to

$$F(u) = \frac{1}{2}\left[F_{\text{even}}(u) + F_{\text{odd}}(u)W_{2K}^{u}\right]. \tag{4.6-43}$$

Also, because $W_M^{u+M} = W_M^u$ and $W_{2M}^{u+M} = -W_{2M}^u$, Eqs. (4.6-41) through (4.6-43) give

$$F(u+K) = \frac{1}{2}\left[F_{\text{even}}(u) - F_{\text{odd}}(u)W_{2K}^{u}\right]. \tag{4.6-44}$$

TABLE 4.1
Summary of some important properties of the 2-D Fourier transform.

Property	Expression(s)
Fourier transform	$F(u,v) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)}$
Inverse Fourier transform	$f(x,y) = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ux/M+vy/N)}$
Polar representation	$F(u,v) = \lvert F(u,v)\rvert e^{-j\phi(u,v)}$
Spectrum	$\lvert F(u,v)\rvert = \left[R^2(u,v) + I^2(u,v)\right]^{1/2}$, $R = \text{Real}(F)$ and $I = \text{Imag}(F)$
Phase angle	$\phi(u,v) = \tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right]$
Power spectrum	$P(u,v) = \lvert F(u,v)\rvert^2$
Average value	$\bar{f}(x,y) = F(0,0) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)$
Translation	$f(x,y)e^{j2\pi(u_0x/M+v_0y/N)} \Leftrightarrow F(u-u_0, v-v_0)$ $f(x-x_0, y-y_0) \Leftrightarrow F(u,v)e^{-j2\pi(ux_0/M+vy_0/N)}$ When $x_0 = u_0 = M/2$ and $y_0 = v_0 = N/2$, then $f(x,y)(-1)^{x+y} \Leftrightarrow F(u-M/2, v-N/2)$ $f(x-M/2, y-N/2) \Leftrightarrow F(u,v)(-1)^{u+v}$
Conjugate symmetry	$F(u,v) = F^*(-u,-v)$ $\lvert F(u,v)\rvert = \lvert F(-u,-v)\rvert$
Differentiation	$\frac{\partial^n f(x,y)}{\partial x^n} \Leftrightarrow (ju)^n F(u,v)$ $(-jx)^n f(x,y) \Leftrightarrow \frac{\partial^n F(u,v)}{\partial u^n}$
Laplacian	$\nabla^2 f(x,y) \Leftrightarrow -(u^2+v^2)F(u,v)$
Distributivity	$\Im[f_1(x,y) + f_2(x,y)] = \Im[f_1(x,y)] + \Im[f_2(x,y)]$ $\Im[f_1(x,y)\cdot f_2(x,y)] \neq \Im[f_1(x,y)]\cdot\Im[f_2(x,y)]$
Scaling	$af(x,y) \Leftrightarrow aF(u,v), f(ax,by) \Leftrightarrow \frac{1}{\lvert ab\rvert}F(u/a, v/b)$
Rotation	$x = r\cos\theta \quad y = r\sin\theta \quad u = \omega\cos\varphi \quad v = \omega\sin\varphi$ $f(r,\theta+\theta_0) \Leftrightarrow F(\omega,\varphi+\theta_0)$
Periodicity	$F(u,v) = F(u+M,v) = F(u,v+N) = F(u+M,v+N)$ $f(x,y) = f(x+M,y) = f(x,y+N) = f(x+M,y+N)$
Separability	See Eqs. (4.6-14) and (4.6-15). Separability implies that we can compute the 2-D transform of an image by first computing 1-D transforms along each row of the image, and then computing a 1-D transform along each column of this intermediate result. The reverse, columns and then rows, yields the same result.

TABLE 4.1 *(continued)* Summary of some important properties of the 2-D Fourier transform.

Property	Expression(s)
Computation of the inverse Fourier transform using a forward transform algorithm	$\frac{1}{MN} f^*(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F^*(u, v) e^{-j2\pi(ux/M + vy/N)}$ This equation indicates that inputting the function $F^*(u, v)$ into an algorithm designed to compute the forward transform (right side of the preceding equation) yields $f^*(x, y)/MN$. Taking the complex conjugate and multiplying this result by MN gives the desired inverse.
Convolution†	$f(x, y) * h(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n) h(x - m, y - n)$
Correlation†	$f(x, y) \circ h(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f^*(m, n) h(x + m, y + n)$
Convolution theorem†	$f(x, y) * h(x, y) \Leftrightarrow F(u, v)H(u, v)$; $f(x, y)h(x, y) \Leftrightarrow F(u, v) * H(u, v)$
Correlation theorem†	$f(x, y) \circ h(x, y) \Leftrightarrow F^*(u, v)H(u, v)$; $f^*(x, y)h(x, y) \Leftrightarrow F(u, v) \circ H(u, v)$
Some useful FT pairs:	
Impulse	$\delta(x, y) \Leftrightarrow 1$
Gaussian	$A2\pi\sigma^2 e^{-2\pi^2\sigma^2(x^2+y^2)} \Leftrightarrow Ae^{-(u^2+v^2)/2\sigma^2}$
Rectangle	$\text{rect}[a, b] \Leftrightarrow ab \frac{\sin(\pi ua)}{(\pi ua)} \frac{\sin(\pi vb)}{(\pi vb)} e^{-j\pi(ua+vb)}$
Cosine	$\cos(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow \frac{1}{2}[\delta(u + Mu_0, v + Nv_0) + \delta(u - Mu_0, v - Nv_0)]$
Sine	$\sin(2\pi u_0 x + 2\pi v_0 y) \Leftrightarrow j\frac{1}{2}[\delta(u + Mu_0, v + Nv_0) - \delta(u - Mu_0, v - Nv_0)]$

† Assumes that functions have been extended by zero padding.

Careful analysis of Eqs. (4.6-41) through (4.6-44) reveals some interesting properties of these expressions. An M-point transform can be computed by dividing the original expression into two parts, as indicated in Eqs. (4.6-43) and (4.6-44). Computing the first half of $F(u)$ requires evaluation of the two $(M/2)$-point transforms given in Eqs. (4.6-41) and (4.6-42). The resulting values of $F_{\text{even}}(u)$ and $F_{\text{odd}}(u)$ are then substituted into Eq. (4.6-43) to obtain $F(u)$ for $u = 0, 1, 2, \ldots, (M/2 - 1)$. The other half then follows directly from Eq. (4.6-44) *without* additional transform evaluations.

In order to examine the computational implications of this procedure, let $m(n)$ and $a(n)$ represent the number of complex multiplications and additions,

respectively, required to implement it. As before, the number of samples is 2^n, where n is a positive integer. Suppose first that $n = 1$. A two-point transform requires the evaluation of $F(0)$; then $F(1)$ follows from Eq. (4.6-44). To obtain $F(0)$ first requires computing and $F_{\text{even}}(0)$ and $F_{\text{odd}}(0)$. In this case $K = 1$ and Eqs. (4.6-41) and (4.6-42) are one-point transforms. Because the Fourier transform of a single point is the sample itself, however, no multiplications or additions are required to obtain $F_{\text{even}}(0)$ and $F_{\text{odd}}(0)$. One multiplication of $F_{\text{odd}}(0)$ by W_2^0 and one addition yield $F(0)$ from Eq. (4.6-43). Then $F(1)$ follows from (4.6-44) with one more addition (subtraction is considered to be the same as addition). As $F_{\text{odd}}(0)W_2^0$ had already been computed, the total number of operations required for a two-point transform consists of $m(1) = 1$ multiplication and $a(1) = 2$ additions.

The next allowed value for n is 2. According to the preceding development, a four-point transform can be divided into two parts. The first half of $F(u)$ requires evaluation of two two-point transforms, as given in Eqs. (4.6-41) and (4.6-42) for $K = 2$. A two-point transform requires $m(1)$ multiplications and $a(1)$ additions, so evaluation of these two equations requires a total of $2m(1)$ multiplications and $2a(1)$ additions. Two further multiplications and additions are necessary to obtain $F(0)$ and $F(1)$ from Eq. (4.6-43). Because $F_{\text{odd}}(u)W_{2K}^u$ already had been computed for $u = \{0, 1\}$, two more additions give $F(2)$ and $F(3)$. The total is then $m(2) = 2m(1) + 2$ and $a(2) = 2a(1) + 4$.

When n is equal to 3, two four-point transforms are considered in the evaluation of $F_{\text{even}}(u)$ and $F_{\text{odd}}(u)$. They require $2m(2)$ multiplications and $2a(2)$ additions. Four more multiplications and eight more additions yield the complete transform. The total then is $m(3) = 2m(2) + 4$ and $a(3) = 2a(2) + 8$.

Continuing this argument for any positive integer value of n leads to recursive expressions for the number of multiplications and additions required to implement the FFT:

$$m(n) = 2m(n-1) + 2^{n-1} \quad n \geq 1 \tag{4.6-45}$$

and

$$a(n) = 2a(n-1) + 2^n \quad n \geq 1 \tag{4.6-46}$$

where $m(0) = 0$ and $a(0) = 0$ because the transform of a single point does not require any additions or multiplications.

Implementation of Eqs. (4.6-41) through (4.6-44) constitutes the successive doubling FFT algorithm. This name comes from the method of computing a two-point transform from two one-point transforms, a four-point transform from two two-point transforms, and so on, for any M equal to an integer power of 2. It is left as an exercise (Problem 4.25) to show that

$$m(n) = \frac{1}{2} M \log_2 M \tag{4.6-47}$$

and

$$a(n) = M \log_2 M. \tag{4.6-48}$$

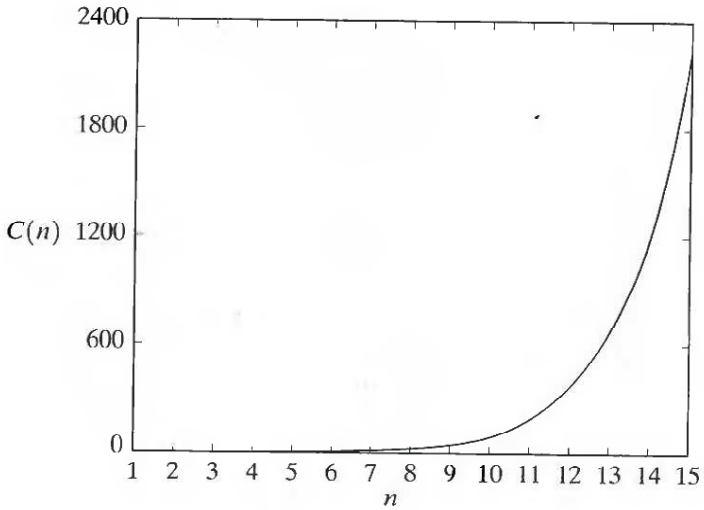


FIGURE 4.42 Computational advantage of the FFT over a direct implementation of the 1-D DFT. Note that the advantage increases rapidly as a function of n.

The computational advantage of the FFT over a direct implementation of the 1-D DFT is defined as

$$C(M) = \frac{M^2}{M \log_2 M} = \frac{M}{\log_2 M}. \tag{4.6-49}$$

Because it is assumed that $M = 2^n$, we can express Eq. (4.6-49) in terms of n:

$$C(n) = \frac{2^n}{n}. \tag{4.6-50}$$

A plot of this function is shown in Fig. 4.42. As noted earlier, the advantage increases rapidly as a function of n. For instance, when $n = 15$ (32,768 points), the FFT has nearly a 2200 to 1 advantage over the DFT. Thus, we would expect that the FFT can be computed nearly 2200 times faster than the DFT on the same machine.

There are so many excellent sources that cover details of the FFT that we will not dwell on this topic further (see, for example, Brigham [1988]). Virtually all comprehensive signal and image processing software packages have generalized implementations of the FFT that also handle cases in which the number of points is not an integer power of 2 (at the expense of computational efficiency). Free FFT programs also are readily available, principally over the Internet.

4.6.7 Some Comments on Filter Design

All the filters discussed in this chapter are specified in equation form. In order to use the filters, we simply sample the equations for the desired values of (u, v). This process results in the filter function $H(u, v)$. In all our examples, this function was multiplied by the (centered) DFT of the input image, and the inverse DFT was computed. All forward and inverse Fourier transforms in this chapter were computed with an FFT algorithm, using the procedure summarized in Fig. 4.35 and Section 4.6.2.

The approach to filtering discussed in this chapter is focused strictly on fundamentals, the focus being specifically to explain the effects of filtering in the frequency domain as clearly as possible. We know of no better way to do that than to treat filtering the way we did here. One can view this development as the basis for "prototyping" a filter. In other words, given a problem for which we want to find a filter, the frequency domain approach is an ideal tool for experimenting, quickly and with full control over filter parameters.

Once a filter for a specific application has been found, it often is of interest to implement the filter directly in the spatial domain, using firmware and/or hardware. Petrou and Bosdogianni [1999] present a nice tie between two-dimensional frequency domain filters and the corresponding digital filters. On the design of 2-D digital filters see Lu and Antoniou [1992].

Summary

The material presented in this chapter, combined with the development in Chapter 3, is a comprehensive foundation for image enhancement. Undoubtedly, it is clear to the reader by now that the area of image enhancement really is a collection of tools that have been found in practice to produce acceptable results in given applications. Most of the tools themselves are well grounded in mathematical and statistical concepts, but their use is strictly problem oriented. In other words, image enhancement is more an art than a science, and the definition of a "properly enhanced" image is highly subjective. In Chapter 5 we extend some of the mathematical concepts developed in this and the last chapter to the area of image restoration. Unlike enhancement, restoration techniques for "improving" images tend to be based on objective, rather than subjective, criteria. As such, restoration techniques are considerably more structured than the methods we have covered thus far.

Another major objective of this chapter was the development of the Fourier transform. Although this was done in the context of image enhancement, the methods developed are perfectly general, as will be seen by the various applications of the DFT in subsequent chapters.

References

For additional reading on the material in Section 4.1 see Hubbard [1998]. The books by Bracewell [2000] and Bracewell [1995] are a good introduction to the continuous Fourier transform and its extensions to two dimensions for image processing. These two books, in addition to Lim [1990], Castleman [1996], Petrou and Bosdogianni [1999], and Brigham [1988], provide comprehensive background for most of the discussion in Section 4.2.

For additional reading on the material in Sections 4.3 and 4.4, see Castleman [1996], Pratt [1991], and Hall [1979]. Effective handling of issues on filter implementation (like ringing) still is a topic of interest, as exemplified by Bakir and Reeves [2000]. For unsharp masking and high-boost filtering, see Schowengerdt [1983]. The material on homomorphic filtering (Section 4.5) is based on a paper by Stockham [1972]; see also the books by Oppenheim and Schafer [1975] and Pitas and Venetsanopoulos [1990]. Brinkman et al. [1998] combine unsharp masking and homomorphic filtering for the enhancement of magnetic resonance images. For the generation of digital filters (Section 4.6.7) based on the frequency domain formulations discussed in this chapter, see Lu and Antoniou [1992] and Petrou and Bosdogianni [1999].

As noted in Section 4.1, "discovery" of the fast Fourier transform (Section 4.6.6) was a major milestone in the popularization of the DFT as a fundamental signal processing tool. Our presentation of the FFT in Section 4.6.6 is based on a paper by Cooley and Tuckey [1965] and on the book by Brigham [1988], who also discusses several implementations of the FFT, including bases other than 2. Formulation of the fast Fourier transform is often credited to Cooley and Tukey [1965]. The FFT, however, has an interesting history worth sketching here. In response to the Cooley-Tukey paper, Rudnick [1966] reported that he was using a similar technique, whose number of operations also was proportional to $M \log_2 M$ and which was based on a method published by Danielson and Lanczos [1942]. These authors, in turn, referenced Runge [1903, 1905] as the source of their technique. The latter two papers, together with the lecture notes of Runge and König [1924], contain the essential computational advantages of present FFT algorithms. Similar techniques also were published by Yates [1937], Stumpff [1939], Good [1958], and Thomas [1963]. A paper by Cooley, Lewis, and Welch [1967a] presents a historical summary and an interesting comparison of results prior to the 1965 Cooley-Tukey paper.

Problems

★ **4.1** Show that $F(u)$ and $f(x)$ in Eqs. (4.2-5) and (4.2-6) are a Fourier transform pair. You can do this by substituting Eq. (4.2-6) for $f(x)$ into Eq. (4.2-5) and showing that the left and right sides are equal. Repeat the process by substituting Eq. (4.2-5) for $F(u)$ into Eq. (4.2-6). You will need the following orthogonality property of exponentials:

$$\sum_{x=0}^{M-1} e^{j2\pi rx/M} e^{-j2\pi ux/M} = \begin{cases} M & \text{if } r = u \\ 0 & \text{otherwise.} \end{cases}$$

See inside front cover

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

4.2 Show that the Fourier transform and its inverse are linear processes (see Section 2.6 regarding linearity).

4.3 Let $F(u, v)$ denote the DFT of an image $f(x, y)$. We know from the discussion in Section 4.2.3 that multiplying $F(u, v)$ by a filter function $H(u, v)$ and taking the inverse Fourier transform will alter the appearance of the image, depending on the nature of the filter. Suppose that $H(u, v) = A$, a positive constant. The net effect of filtering will be to multiply the image by the same constant. Using the convolution theorem, explain mathematically why the pixels in the spatial domain representation are multiplied by the same constant.

★ **4.4** A Gaussian lowpass filter in the frequency domain has the transfer function

$$H(u, v) = Ae^{-(u^2+v^2)/2\sigma^2}.$$

Show that the corresponding filter in the *spatial* domain has the form

$$h(x, y) = A2\pi\sigma^2 e^{-2\pi^2\sigma^2(x^2+y^2)}.$$

(*Hint:* Treat the variables as continuous to simplify manipulation.)

4.5 As shown in Eq. (4.4-1), a highpass filter has the transfer function

$$H_{\text{hp}}(u, v) = 1 - H_{\text{lp}}(u, v)$$

where $H_{\text{lp}}(u, v)$ is the transfer function of the corresponding lowpass filter. Using the result in Problem 4.4, what is the form of the *spatial domain* Gaussian highpass filter?

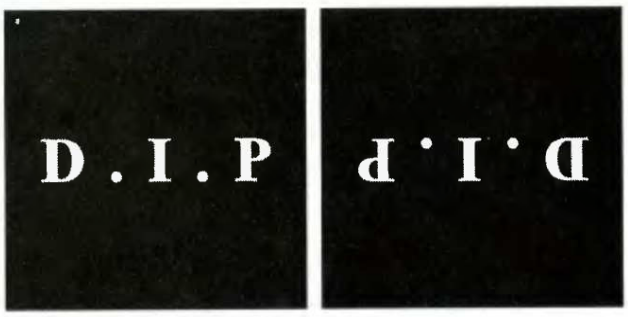
4.6 ★ **(a)** Prove the validity of Eq. (4.2-21).

(b) Prove the validity of Eqs. (4.6-1) and (4.6-2).

4.7 What is the source of nearly periodic bright points in the horizontal axis of the spectrum in Fig. 4.11(b)?

★4.8 Each of the spatial filters shown in Fig. 4.23 has a strong spike at the origin. Explain the source of these spikes.

4.9 Consider the images shown. The image on the right was obtained by (a) multiplying the image on the left by $(-1)^{x+y}$; (b) computing the DFT; (c) taking the complex conjugate of the transform; (d) computing the inverse DFT; and (e) multiplying the real part of the result by $(-1)^{x+y}$. Explain (mathematically) why the image on the right appears as it does.



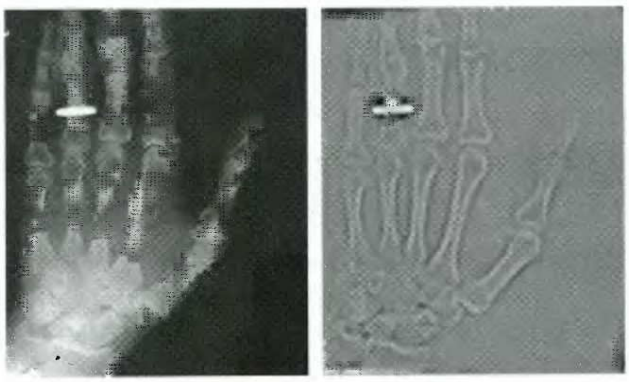
4.10 Show that if a filter transfer function $H(u, v)$ is real and symmetric, then the corresponding spatial domain filter $h(x, y)$ also is real and symmetric.

★4.11 Prove the validity of the convolution theorem. For simplicity, limit your development to continuos functions of one variable.

4.12 Consider the images shown. The image on the right was obtained by lowpass filtering the image on the left with a Gaussian lowpass filter, and then highpass filtering the result with a Gaussian highpass filter. The dimension of the images is 420×344, and $D_0 = 25$ was used for both filters.

(a) Consider the figure on the right. Explain why the center part of the ring appears so bright and solid, when the dominant characteristic of the filtered image consists of edges on the outer boundary of objects (e.g., fingers, wrist bones) with a darker area in between. In other words, would you not expect the highpass filter to render the constant area inside the ring dark, since a highpass filter eliminates the dc term?

(b) Do you think the result would have been different if the order of the filtering process had been reversed?



Original image courtesy of Dr. Thomas R. Gest, Division of Anatomical Sciences, University of Michigan Medical School.

4.13 Given an image of size $M \times N$, you are asked to perform an experiment that consists of repeatedly lowpass filtering the image using a Gaussian lowpass filter with a given cutoff D_0. You may ignore computational round-off errors. Let $k_{\min}$ denote the smallest positive number representable in the machine in which the proposed experiment will be conducted.

★ **(a)** Let K denote the number of applications of the filter. Can you predict, without doing the experiment what the result (image) will be for a sufficiently large value of K? If so, what is that result?

(b) Derive an expression for the *minimum* value of K that will guarantee the result that you predicted.

4.14 Suppose that you form a lowpass *spatial* filter that averages the four immediate neighbors of a point (x, y) but excludes the point itself.

(a) Find the equivalent filter $H(u, v)$ in the frequency domain.

(b) Show that your result is a lowpass filter.

★ **4.15** The basic approach used to approximate a discrete derivative (Section 3.7) involves taking differences of the form $f(x + 1, y) - f(x, y)$.

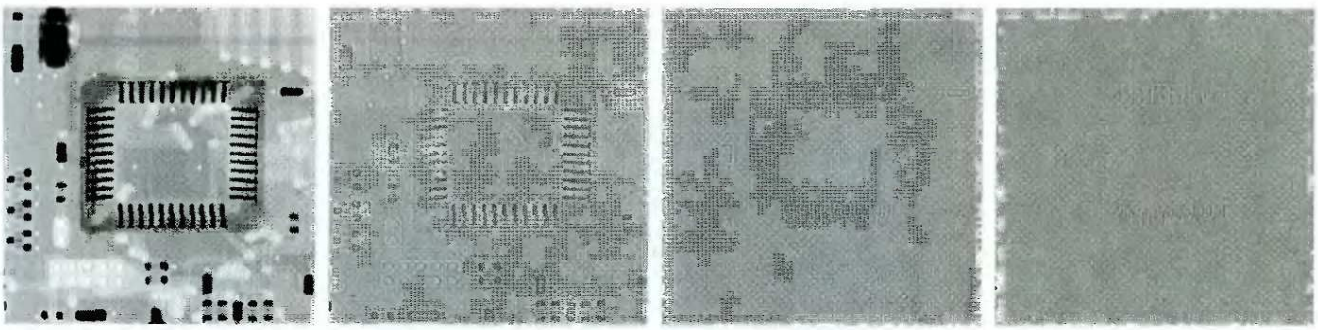
(a) Obtain the filter transfer function, $H(u, v)$, for performing the equivalent process in the frequency domain.

(b) Show that $H(u, v)$ is a highpass filter.

4.16 Consider the sequence of images shown. The image on the left is a segment of an X-ray image of a commercial printed circuit board. The images following it are, respectively, the results of subjecting the image to 1, 10, and 100 passes of a Gaussian highpass filter with $D_0 = 30$. The images are of size 330×334 pixels, with each pixel being represented by 8 bits of gray. They have been scaled for display, but this has no effect on the problem statement.

(a) There is a hint in these results that indicates that the image will cease to change after some finite number of passes. Show whether or not this in fact is the case. You may ignore computational round-off errors. Let $k_{\min}$ denote the smallest positive number representable in the machine in which the proposed experiment will be conducted.

(b) If you determined in (a) that changes would cease after a finite number of iterations, determine the minimum value of that number.



Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.

4.17 As shown in Fig. 4.30, combining high-frequency emphasis and histogram equalization is an effective method for achieving edge sharpening and contrast enhancement.

(a) Show whether or not it matters which process is applied first.

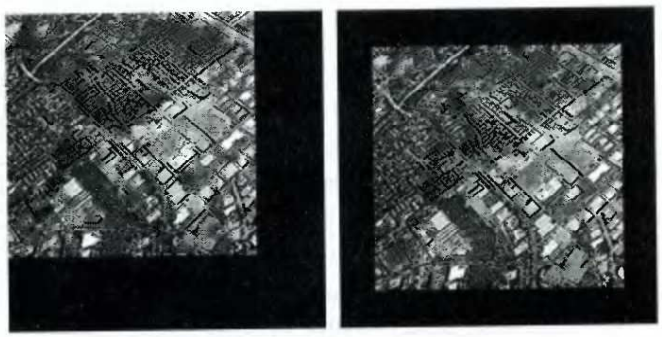
(b) If the order does matter, give a rationale for using one or the other method first.

★ 4.18 Can you think of a way to use the Fourier transform to compute (or partially compute) the magnitude of the gradient for use in image differentiation (see Section 3.7.3)? If your answer is yes, give a method to do it. If your answers is no, explain why.

4.19 In Section 4.4.4 we started with the definition of the Laplacian and derived a filter for performing the equivalent operation via the frequency domain. Start with the mask approximation shown in Fig. 4.27(f) and find the frequency domain filter that would implement that approximation.

4.20 Use the transfer function of a Butterworth highpass filter of order n to construct a homomorphic filter. Your filter must exhibit the characteristic shape shown in Fig. 4.32 and must include the parameters shown in that figure.

★ 4.21 The need for image padding when filtering in the frequency domain was discussed in some detail in Section 4.6.3. We showed in that section that images needed to be padded by appending zeros to the ends of rows and columns in the image (see the following image on the left). Do you think it would make a difference if we centered the image and surrounded it by a border of zeros instead (see image on the right), but without changing the total number of zeros used? Explain.

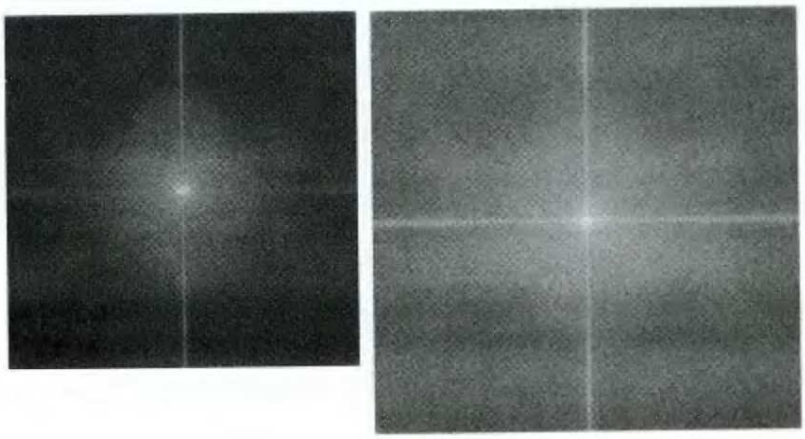


Original image courtesy of NASA.

4.22 The two Fourier spectra shown are of the same image. The spectrum on the left corresponds to the original image, and the spectrum on the right was obtained after the image was padded with zeros.

(a) Explain the difference in overall contrast.

(b) Explain the significant increase in signal strength along the vertical and horizontal axes of the spectrum shown on the right.



4.23 The correlation function shown in Fig. 4.41(e) was computed via the frequency domain, according to Eq. (4.6-31). The complex conjugate of the transform of the template in Fig. 4.41(d) was used in the computation. Sketch what Fig. 4.41(e) would look like if, instead, we had used the conjugate of the transform of the image.

4.24 With reference to Fig. 4.41, sketch what the correlation function [Fig. 4.41(e)] would look like if the letters UTK were only one pixel away from

★ **(a)** The left border of the image

★ **(b)** The right border of the image

(c) The top of the image

(d) The bottom of the image

In (a) and (b), assume that the letters are centered with respect to an imaginary horizontal line passing through the center of all the letters and also through the center of the image. In (c) and (d) they are centered with respect to a vertical line passing through the center of the letter T and also through the center of the image.

★ **4.25** Show the validity of Eqs. (4.6-47) and (4.6-48). (*Hint:* Use proof by induction.)

4.26 Suppose that you are given a set of images generated by an experiment dealing with the analysis of stellar events. Each image contains a set of bright, widely scattered dots corresponding to stars in a sparsely occupied section of the universe. The problem is that the stars are barely visible, due to superimposed illumination resulting from atmospheric dispersion. If these images are modeled as the product of a constant illumination component with a set of impulses, give an enhancement procedure based on homomorphic filtering designed to bring out the image components due to the stars themselves.

4.27 A skilled medical technician is charged with the job of inspecting a certain class of images generated by an electron microscope. In order to simplify the inspection task, the technician decides to use digital image enhancement and, to this end, examines a set of representative images and finds the following problems: (1) bright, isolated dots that are of no interest; (2) lack of sharpness; (3) not enough contrast in some images; and (4) shifts in the average gray-level value, when this value should be V to perform correctly certain intensity measurements. The technician wants to correct these problems and then display in white all gray levels in the band between I_1 and I_2, while keeping normal tonality in the remaining gray levels. Propose a sequence of processing steps that the technician can follow to achieve the desired goal. You may use techniques from both Chapters 3 and 4.

5 Image Restoration

Things which we see are not by themselves what we see. . . .
It remains completely unknown to us what the objects may be by themselves and apart from the receptivity of our senses. We know nothing but our manner of perceiving them.

Immanuel Kant

Preview

As in image enhancement, the ultimate goal of restoration techniques is to improve an image in some predefined sense. Although there are areas of overlap, image enhancement is largely a subjective process, while image restoration is for the most part an objective process. Restoration attempts to reconstruct or recover an image that has been degraded by using a priori knowledge of the degradation phenomenon. Thus restoration techniques are oriented toward modeling the degradation and applying the inverse process in order to recover the original image.

This approach usually involves formulating a criterion of goodness that will yield an optimal estimate of the desired result. By contrast, enhancement techniques basically are heuristic procedures designed to manipulate an image in order to take advantage of the psychophysical aspects of the human visual system. For example, contrast stretching is considered an enhancement technique because it is based primarily on the pleasing aspects it might present to the viewer, whereas removal of image blur by applying a deblurring function is considered a restoration technique.

The material developed in this chapter is strictly introductory. We consider the restoration problem only from the point where a degraded, *digital* image is given; thus we consider topics dealing with sensor, digitizer, and display degradations only superficially. These subjects, although of importance in the overall treatment of image restoration applications, are beyond the scope of the present discussion.

As in Chapters 3 and 4, some restoration techniques are best formulated in the spatial domain, while others are better suited for the frequency domain. For example, spatial processing is applicable when the only degradation is additive noise. On the other hand, degradations such as image blur are difficult to approach in the spatial domain using small masks. In this case, frequency domain filters based on various criteria of optimality are the approaches of choice. These filters also take into account the presence of noise. As in Chapter 4 (see comments in Section 4.6.7), a restoration filter that solves a given application in the frequency domain often is used as the basis for generating a digital filter that will be more suitable for routine operation using a hardware/firmware implementation.

5.1 A Model of the Image Degradation/Restoration Process

As Fig. 5.1 shows, the degradation process is modeled in this chapter as a degradation function that, together with an additive noise term, operates on an input image $f(x, y)$ to produce a degraded image $g(x, y)$. Given $g(x, y)$, some knowledge about the degradation function H, and some knowledge about the additive noise term $\eta(x, y)$, the objective of restoration is to obtain an estimate $\hat{f}(x, y)$ of the original image. We want the estimate to be as close as possible to the original input image and, in general, the more we know about H and η, the closer $\hat{f}(x, y)$ will be to $f(x, y)$. The approach used throughout most of this chapter is based on various types of image restoration filters.

It is shown in Section 5.5 that if H is a linear, position-invariant process, then the degraded image is given in the *spatial domain* by

$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y) \tag{5.1-1}$$

where $h(x, y)$ is the spatial representation of the degradation function and, as in Chapter 4, the symbol "$*$" indicates convolution. We know from the discussion in Sections 4.2.4 and 4.6.4 that convolution in the spatial domain is equal to multiplication in the frequency domain, so we may write the model in Eq. (5.1-1) in an equivalent *frequency domain* representation:

$$G(u, v) = H(u, v)F(u, v) + N(u, v) \tag{5.1-2}$$

where the terms in capital letters are the Fourier transforms of the corresponding terms in Eq. (5.1-1). These two equations are the bases for most of the material in this chapter.

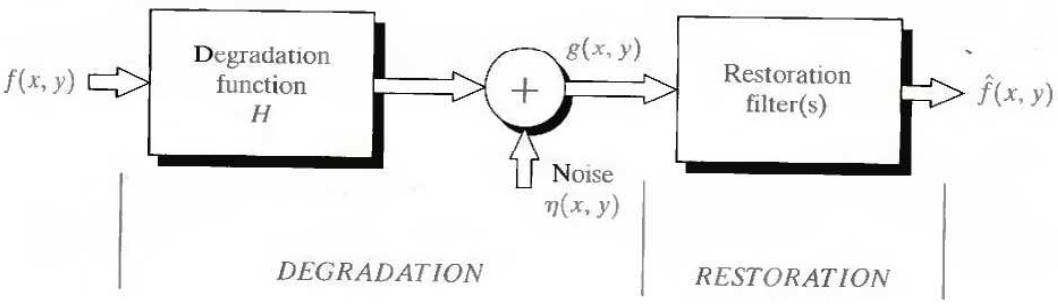


FIGURE 5.1 A model of the image degradation/restoration process.

In the following three sections, we assume that H is the identity operator, and we deal only with degradations due to noise. Beginning in Section 5.6 we consider a number of important image degradations functions and look at several methods for image restoration in the presence of both H and η.

5.2 Noise Models

The principal sources of noise in digital images arise during image acquisition (digitization) and/or transmission. The performance of imaging sensors is affected by a variety of factors, such as environmental conditions during image acquisition, and by the quality of the sensing elements themselves. For instance, in acquiring images with a CCD camera, light levels and sensor temperature are major factors affecting the amount of noise in the resulting image. Images are corrupted during transmission principally due to interference in the channel used for transmission. For example, an image transmitted using a wireless network might be corrupted as a result of lightning or other atmospheric disturbance.

5.2.1 Spatial and Frequency Properties of Noise

Relevant to our discussion are parameters that define the spatial characteristics of noise, and whether the noise is correlated with the image. Frequency properties refer to the frequency content of noise in the Fourier sense (i.e., as opposed to the electromagnetic spectrum). For example, when the Fourier spectrum of noise is constant, the noise usually is called *white noise*. This terminology is a carry over from the physical properties of white light, which contains nearly all frequencies in the visible spectrum in equal proportions. From the discussion in Chapter 4, it is not difficult to show that the Fourier spectrum of a function containing all frequencies in equal proportions is a constant.

With the exception of spatially periodic noise (Section 5.2.3), we assume in this chapter that noise is independent of spatial coordinates, and that it is uncorrelated with respect to the image itself (that is, there is no correlation between pixel values and the values of noise components). Although these assumptions are at least partially invalid in some applications (quantum-limited imaging, such as in X-ray and nuclear-medicine imaging, is a good example), the complexities of dealing with spatially dependent and correlated noise are beyond the scope of our discussion.

5.2.2 Some Important Noise Probability Density Functions

See inside front cover

Consult the book web site for a brief review of probability theory.

Based on the assumptions in the previous section, the *spatial* noise descriptor with which we shall be concerned is the statistical behavior of the gray-level values in the noise component of the model in Fig. 5.1. These may be considered random variables, characterized by a probability density function (PDF). The following are among the most common PDFs found in image processing applications.

Gaussian noise

Because of its mathematical tractability in both the spatial and frequency domains, Gaussian (also called *normal*) noise models are used frequently in prac-

tice. In fact, this tractability is so convenient that it often results in Gaussian models being used in situations in which they are marginally applicable at best.

The PDF of a Gaussian random variable, z, is given by

$$p(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(z-\mu)^2/2\sigma^2} \tag{5.2-1}$$

where z represents gray level, μ is the mean of average value of z, and σ is its standard deviation. The standard deviation squared, σ^2, is called the *variance* of z. A plot of this function is shown in Fig. 5.2(a). When z is described by Eq. (5.2-1), approximately 70% of its values will be in the range $[(\mu - \sigma), (\mu + \sigma)]$, and about 95% will be in the range $[(\mu - 2\sigma), (\mu + 2\sigma)]$.

Rayleigh noise

The PDF of Rayleigh noise is given by

$$p(z) = \begin{cases} \frac{2}{b}(z - a)e^{-(z-a)^2/b} & \text{for } z \geq a \\ 0 & \text{for } z < a. \end{cases} \tag{5.2-2}$$

The mean and variance of this density are given by

$$\mu = a + \sqrt{\pi b/4} \tag{5.2-3}$$

and

$$\sigma^2 = \frac{b(4 - \pi)}{4}. \tag{5.2-4}$$

Figure 5.2(b) shows a plot of the Rayleigh density. Note the displacement from the origin and the fact that the basic shape of this density is skewed to the right. The Rayleigh density can be quite useful for approximating skewed histograms.

Erlang (Gamma) noise

The PDF of Erlang noise is given by

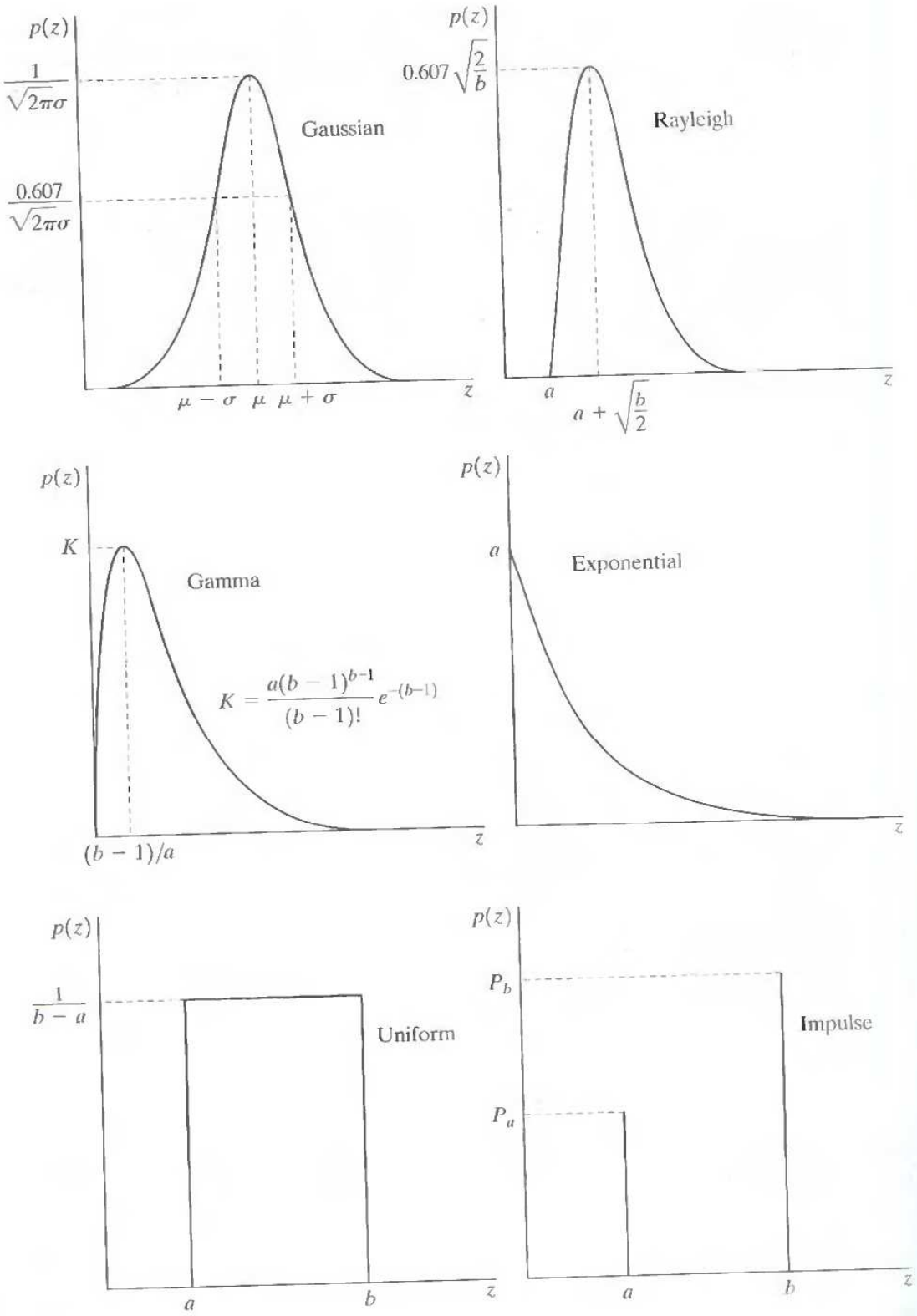
$$p(z) = \begin{cases} \frac{a^b z^{b-1}}{(b-1)!} e^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases} \tag{5.2-5}$$

where the parameters are such that $a > 0$, b is a positive integer, and "!" indicates factorial. The mean and variance of this density are given by

$$\mu = \frac{b}{a} \tag{5.2-6}$$

and

$$\sigma^2 = \frac{b}{a^2}. \tag{5.2-7}$$



a b
c d
e f

FIGURE 5.2 Some important probability density functions.

Figure 5.2(c) shows a plot of this density. Although Eq. (5.2-5) often is referred to as the *gamma density*, strictly speaking this is correct only when the denominator is the gamma function, $\Gamma(b)$. When the denominator is as shown, the density is more appropriately called the *Erlang density*.

Exponential noise

The PDF of *exponential* noise is given by

$$p(z) = \begin{cases} ae^{-az} & \text{for } z \geq 0 \\ 0 & \text{for } z < 0 \end{cases} \tag{5.2-8}$$

where $a > 0$. The mean and variance of this density function are

$$\mu = \frac{1}{a} \tag{5.2-9}$$

and

$$\sigma^2 = \frac{1}{a^2}. \tag{5.2-10}$$

Note that this PDF is a special case of the Erlang PDF, with $b = 1$. Figure 5.2(d) shows a plot of this density function.

Uniform noise

The PDF of *uniform* noise is given by

$$p(z) = \begin{cases} \dfrac{1}{b-a} & \text{if } a \leq z \leq b \\ 0 & \text{otherwise.} \end{cases} \tag{5.2-11}$$

The mean of this density function is given by

$$\mu = \frac{a+b}{2} \tag{5.2-12}$$

and its variance by

$$\sigma^2 = \frac{(b-a)^2}{12}. \tag{5.2-13}$$

Figure 5.2(e) shows a plot of the uniform density.

Impulse (salt-and-pepper) noise

The PDF of (*bipolar*) *impulse* noise is given by

$$p(z) = \begin{cases} P_a & \text{for } z = a \\ P_b & \text{for } z = b \\ 0 & \text{otherwise} \end{cases} \tag{5.2-14}$$

If $b > a$, gray-level b will appear as a light dot in the image. Conversely, level a will appear like a dark dot. If either P_a or P_b is zero, the impulse noise is called

unipolar. If neither probability is zero, and especially if they are approximately equal, impulse noise values will resemble salt-and-pepper granules randomly distributed over the image. For this reason, bipolar impulse noise also is called *salt-and-pepper* noise. *Shot* and *spike* noise also are terms used to refer to this type of noise. In our discussion we will use the terms *impulse* or *salt-and-pepper* noise interchangeably.

Noise impulses can be negative or positive. Scaling usually is part of the image digitizing process. Because impulse corruption usually is large compared with the strength of the image signal, impulse noise generally is digitized as extreme (pure black or white) values in an image. Thus, the assumption usually is that a and b are "saturated" values, in the sense that they are equal to the minimum and maximum allowed values in the digitized image. As a result, negative impulses appear as black (pepper) points in an image. For the same reason, positive impulses appear white (salt) noise. For an 8-bit image this means that $a = 0$ (black) and $b = 255$ (white). Figure 5.2(f) shows the PDF of impulse noise.

As a group, the preceding PDFs provide useful tools for modeling a broad range of noise corruption situations found in practice. For example, Gaussian noise arises in an image due to factors such as electronic circuit noise and sensor noise due to poor illumination and/or high temperature. The Rayleigh density is helpful in characterizing noise phenomena in range imaging. The exponential and gamma densities find application in laser imaging. Impulse noise is found in situations where quick transients, such as faulty switching, take place during imaging, as mentioned in the previous paragraph. The uniform density is perhaps the least descriptive of practical situations. However, the uniform density is quite useful as the basis for numerous random number generators that are used in simulations (Peebles [1993]).

EXAMPLE 5.1: Sample noisy images and their histograms.

Figure 5.3 shows a test pattern well-suited for illustrating the noise models just discussed. This is a suitable pattern to use because it is composed of simple, constant areas that span the gray scale from black to near white in only three increments. This facilitates visual analysis of the characteristics of the various noise components added to the image.

Figure 5.4 shows the test pattern after addition of the six types of noise discussed thus far in this section. Shown below each image is the histogram computed directly from that image. The parameters of the noise were chosen in each case so that the histogram corresponding to the three gray levels in the test pattern would start to merge. This made the noise quite visible, without obscuring the basic structure of the underlying image.

We see a close correspondence in comparing the histograms in Fig. 5.4 with the PDFs in Fig. 5.2. The histogram for the salt-and-pepper example has an extra peak at the white end of the spectrum because the noise components were pure black and white, and the lightest component of the test pattern (the circle) is light gray. With the exception of slightly different overall intensity, it is difficult to differentiate visually between the first five images in Fig. 5.4, even though their histograms are significantly different. The salt-and-pepper appearance of

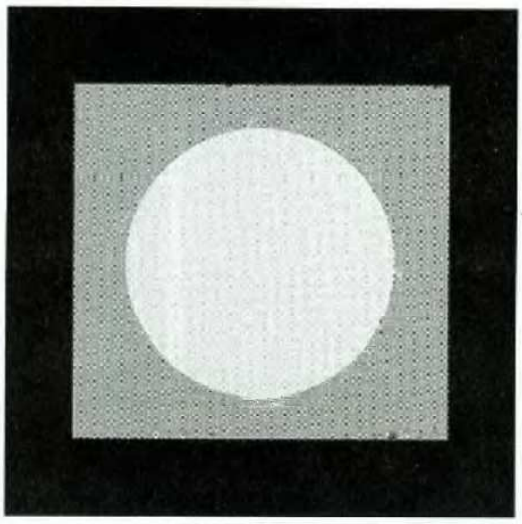


FIGURE 5.3 Test pattern used to illustrate the characteristics of the noise PDFs shown in Fig. 5.2.

the image corrupted by impulse noise is the only one that is visually indicative of the type of noise causing the degradation. ■

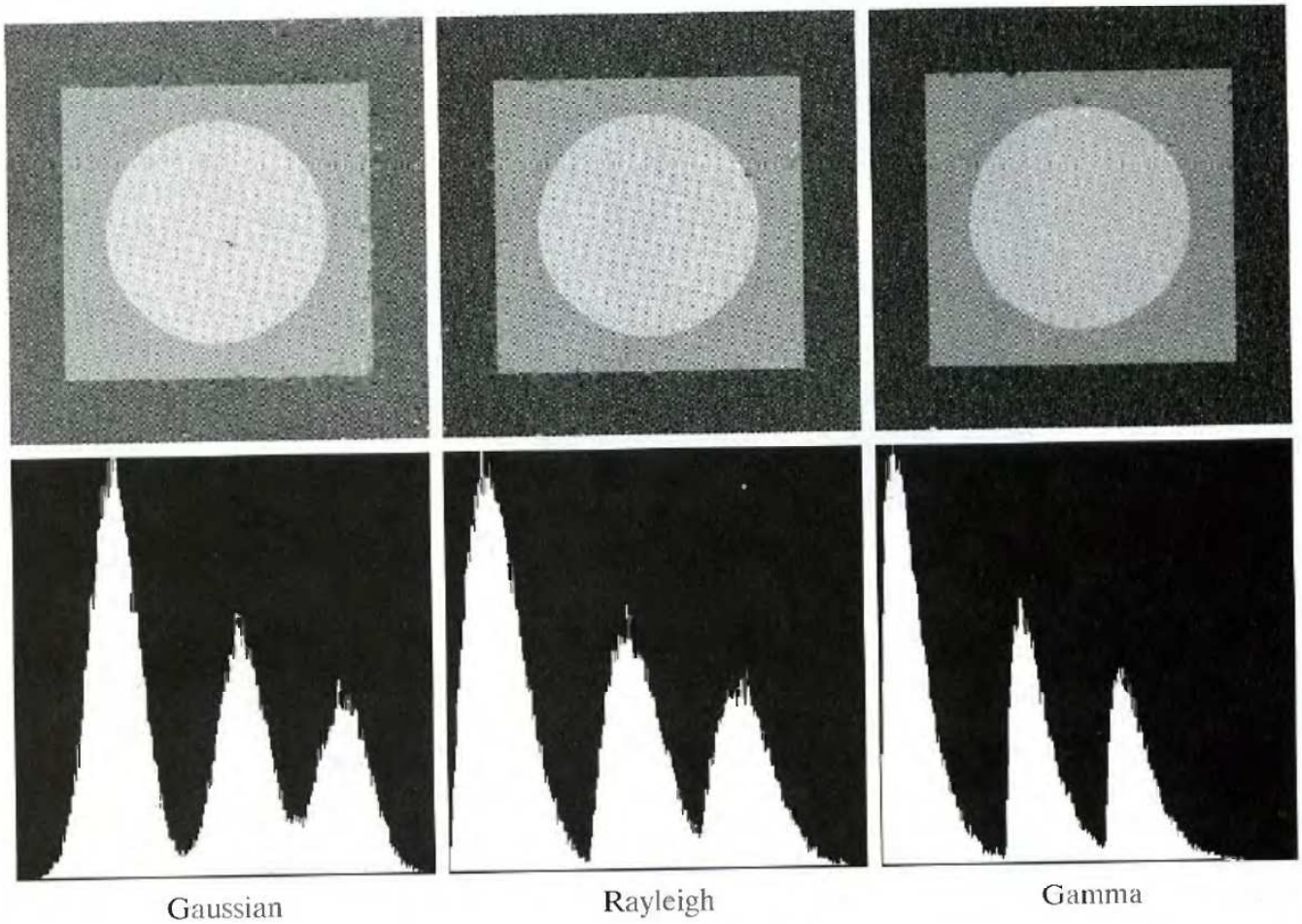
5.2.3 Periodic Noise

Periodic noise in an image irises typically from electrical or electromechanical interference during image acquisition. This is the only type of spatially dependent noise that will be considered in this chapter. As discussed in Section 5.4, periodic noise can be reduced significantly via frequency domain filtering. For example, consider the image shown in Fig. 5.5(a). This image is severely corrupted by (spatial) sinusoidal noise of various frequencies. The Fourier transform of a pure sinusoid is a pair of conjugate impulses located at the conjugate frequencies of the sine wave (Table 4.1). Thus, if the amplitude of a sine wave in the spatial domain is strong enough, we would expect to see in the spectrum of the image a pair of impulses for each sine wave in the image. As shown in Fig. 5.5(b), this is indeed the case, with the impulses appearing in an approximate circle because the frequency values in this particular case are so arranged. We will have much more to say in Section 5.4 about this and other examples of periodic noise.

5.2.4 Estimation of Noise Parameters

The parameters of periodic noise typically are estimated by inspection of the Fourier spectrum of the image. As noted in the previous section, periodic noise tends to produce frequency spikes that often can be detected even by visual analysis. Another approach is to attempt to infer the periodicity of noise components directly from the image, but this is possible only in simplistic cases. Automated analysis is possible in situations in which the noise spikes are either exceptionally pronounced, or when some knowledge is available about the general location of the frequency components of the interference.

The parameters of noise PDFs may be known partially from sensor specifications, but it is often necessary to estimate them for a particular imaging

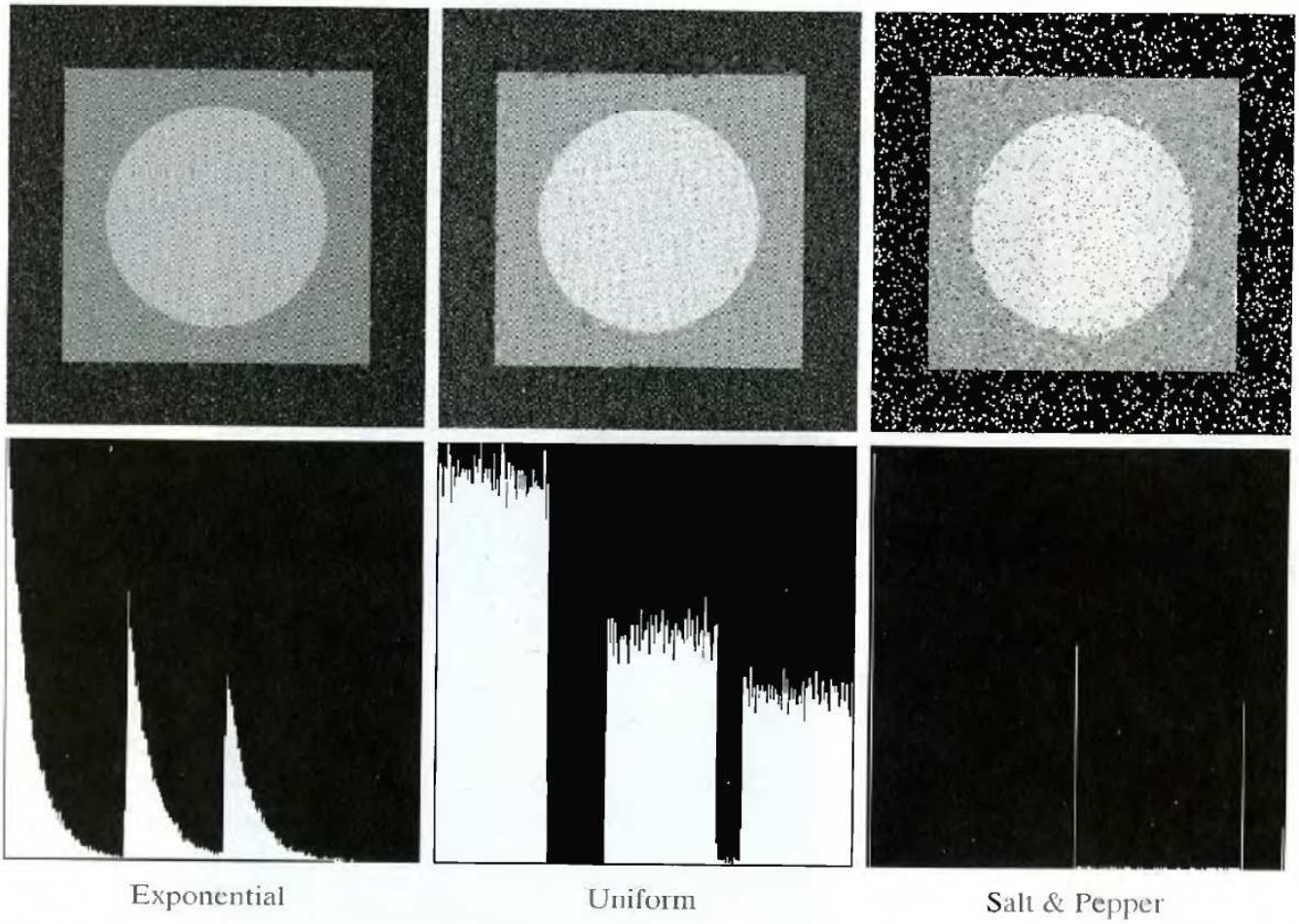


a b c
d e f

FIGURE 5.4 Images and histograms resulting from adding Gaussian, Rayleigh, and gamma noise to the image in Fig. 5.3.

arrangement. If the imaging system is available, one simple way to study the characteristics of system noise is to capture a set of images of "flat" environments. For example, in the case of an optical sensor, this is as simple as imaging a solid gray board that is illuminated uniformly. The resulting images typically are good indicators of system noise.

When only images already generated by the sensor are available, frequently it is possible to estimate the parameters of the PDF from small patches of reasonably constant gray level. For example, the vertical strips (of 150×20 pixels) shown in Fig. 5.6 were cropped from the Gaussian, Rayleigh, and uniform images in Fig. 5.4. The histograms shown were calculated using image data from these small strips. The histograms in Fig. 5.4 that correspond to the histograms in Fig. 5.6 are the ones in the middle of the group of three in Figs. 5.4(d), (e), and (k). We see that the shapes of these histograms correspond quite closely to the shapes of the histograms in Fig. 5.6. Their heights are different due to scaling, but the shapes are unmistakably similar.



g h i
j k l

FIGURE 5.4 *(Continued)* Images and histograms resulting from adding exponential, uniform, and impulse noise to the image in Fig. 5.3.

The simplest way to use the data from the image strips is for calculating the mean and variance of the gray levels. Consider a strip (subimage) denoted by S. We can use the following sample approximations from basic statistics:

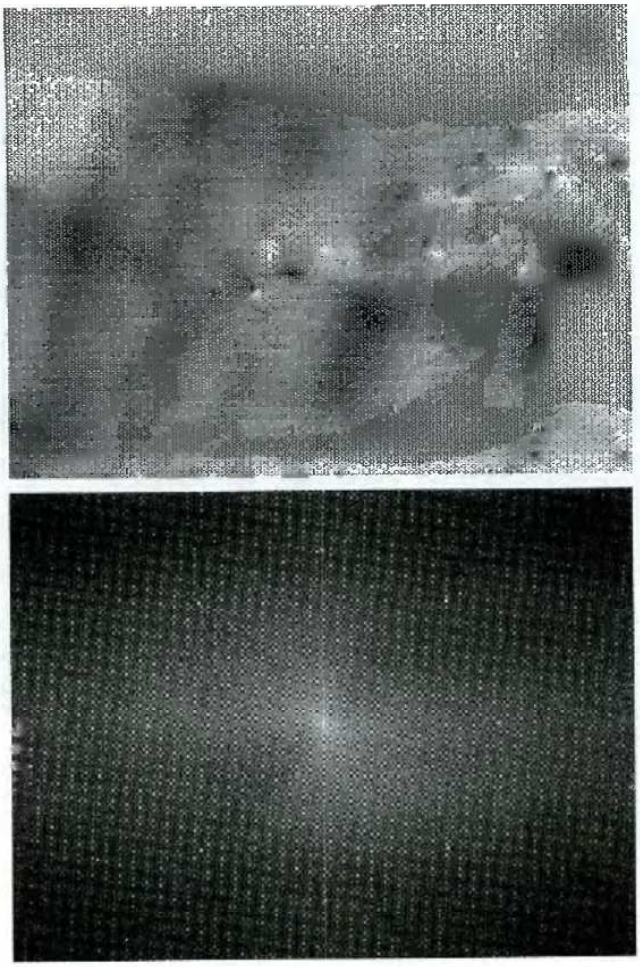
$$\mu = \sum_{z_i \in S} z_i p(z_i) \tag{5.2-15}$$

and

$$\sigma^2 = \sum_{z_i \in S} (z_i - \mu)^2 p(z_i) \tag{5.2-16}$$

where the z_i's are the gray-level values of the pixels in S, and $p(z_i)$ are the corresponding normalized histogram values.

The shape of the histogram identifies the closest PDF match. If the shape is approximately Gaussian, then the mean and variance is all we need because the Gaussian PDF is completely specified by these two parameters. For the



a
b

FIGURE 5.5 (a) Image corrupted by sinusoidal noise. (b) Spectrum (each pair of conjugate impulses corresponds to one sine wave). (Original image courtesy of NASA.)

other shapes discussed in Section 5.2.2, we use the mean and variance to solve for the parameters a and b. Impulse noise is handled differently because the estimate needed is of the actual probability of occurrence of white and black pixels. Obtaining this estimate requires that both black and white pixels be visible, so a midgray, relatively constant area is needed in the image in order to be able to compute a histogram. The heights of the peaks corresponding to black and white pixels are the estimates of P_a and P_b in Eq. (5.2-14).

5.3 Restoration in the Presence of Noise Only–Spatial Filtering

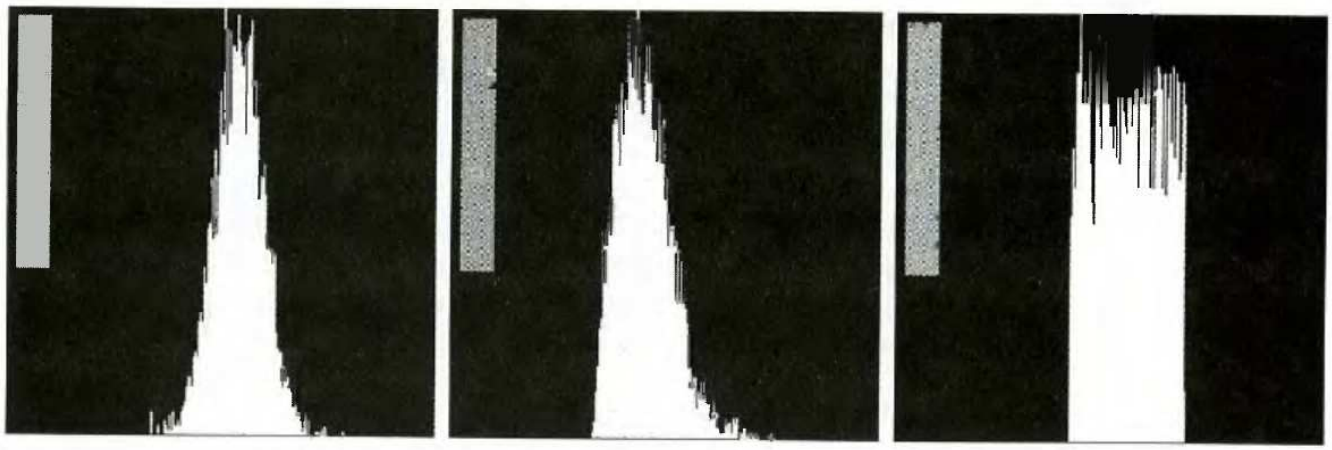
When the only degradation present in an image is noise, Eqs. (5.1-1) and (5.1-2) become

$$g(x, y) = f(x, y) + \eta(x, y) \tag{5.3-1}$$

and

$$G(u, v) = F(u, v) + N(u, v). \tag{5.3-2}$$

The noise terms are unknown, so subtracting them from $g(x, y)$ or $G(u, v)$ is not a realistic option. In the case of periodic noise, it usually is possible to



a b c

FIGURE 5.6 Histograms computed using small strips (shown as inserts) from (a) the Gaussian, (b) the Rayleigh, and (c) the uniform noisy images in Fig. 5.4.

estimate $N(u, v)$ from the spectrum of $G(u, v)$, as noted in Section 5.2.3. In this case $N(u, v)$ can be subtracted from $G(u, v)$ to obtain an estimate of the original image. In general, however, this type of knowledge is the exception, rather than the rule.

Spatial filtering is the method of choice in situations when only additive noise is present. This topic was discussed in detail in Sections 3.5 and 3.6 in connection with image enhancement. In fact, enhancement and restoration become almost indistinguishable disciplines in this particular case. With the exception of the nature of the computation performed by a specific filter, the mechanics for implementing all the filters that follow are exactly as discussed in Section 3.5.

5.3.1 Mean Filters

In this section we discuss briefly the noise-reduction spatial filters introduced in Section 3.6 and develop several other filters whose performance is in many cases superior to the filters discussed in that section.

Arithmetic mean filter

This is the simplest of the mean filters. Let S_{xy} represent the set of coordinates in a rectangular subimage window of size $m \times n$, centered at point (x, y). The arithmetic mean filtering process computes the average value of the corrupted image $g(x, y)$ in the area defined by S_{xy}. The value of the restored image $\hat{f}$ at any point (x, y) is simply the arithmetic mean computed using the pixels in the region defined by S_{xy}. In other words,

$$\hat{f}(x, y) = \frac{1}{mn} \sum_{(s,t) \in S_{xy}} g(s, t). \tag{5.3-3}$$

This operation can be implemented using a convolution mask in which all coefficients have value $1/mn$. As discussed in Section 3.6.1, a mean filter simply smoothes local variations in an image. Noise is reduced as a result of blurring.

Geometric mean filter

An image restored using a *geometric mean* filter is given by the expression

$$\hat{f}(x, y) = \left[\prod_{(s,t)\in S_{xy}} g(s, t) \right]^{\frac{1}{mn}}. \tag{5.3-4}$$

Here, each restored pixel is given by the product of the pixels in the subimage window, raised to the power $1/mn$. As shown in Example 5.2, a geometric mean filter achieves smoothing comparable to the arithmetic mean filter, but it tends to lose less image detail in the process.

Harmonic mean filter

The *harmonic mean* filtering operation is given by the expression

$$\hat{f}(x, y) = \frac{mn}{\sum_{(s,t)\in S_{xy}} \frac{1}{g(s, t)}}. \tag{5.3-5}$$

The harmonic mean filter works well for salt noise, but fails for pepper noise. It does well also with other types of noise like Gaussian noise.

Contraharmonic mean filter

The *contraharmonic* mean filtering operation yields a restored image based on the expression

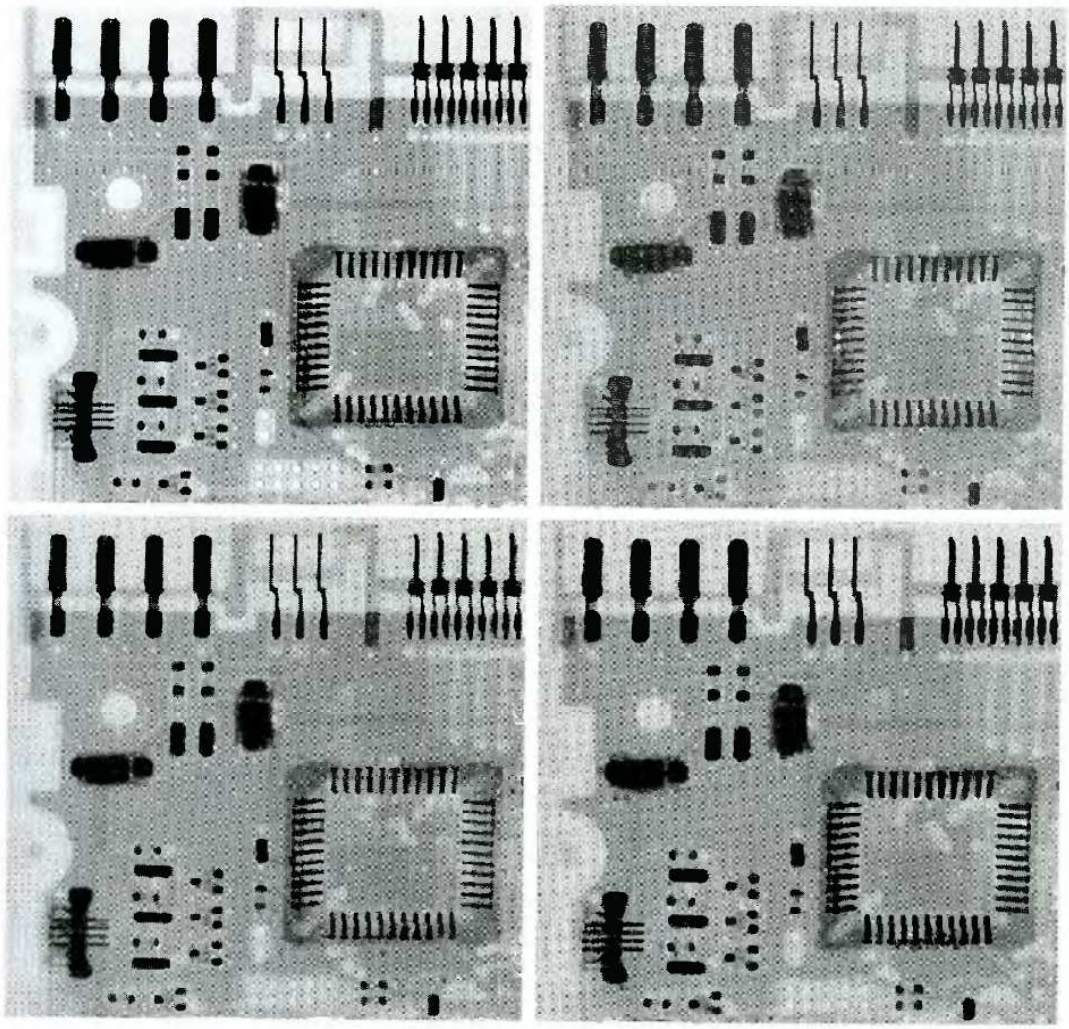
$$\hat{f}(x, y) = \frac{\sum_{(s,t)\in S_{xy}} g(s, t)^{Q+1}}{\sum_{(s,t)\in S_{xy}} g(s, t)^{Q}} \tag{5.3-6}$$

where Q is called the *order* of the filter. This filter is well suited for reducing or virtually eliminating the effects of salt-and-pepper noise. For positive values of Q, the filter eliminates pepper noise. For negative values of Q it eliminates salt noise. It cannot do both simultaneously. Note that the contraharmonic filter reduces to the arithmetic mean filter if $Q = 0$, and to the harmonic mean filter if $Q = -1$.

EXAMPLE 5.2: Illustration of mean filters.

■ Figure 5.7(a) shows an X-ray image of a circuit board, and Fig. 5.7(b) shows the same image, but corrupted with additive Gaussian noise of zero mean and variance of 400. For this type of image this is a significant level of noise. Figures 5.7(c) and (d) show, respectively, the result of filtering the noisy image with an arithmetic mean filter of size 3×3 and a geometric mean filter of the same size. Although both filters did a reasonable job of attenuating the contribution due to noise, the geometric mean filter did not blur the image as much as the arithmetic filter. For instance, the connector fingers at the top of the image are much sharper in Fig. 5.7(d) than in (c). The same is true in other parts of the image (see Problem 5.1).

Figure 5.8(a) shows the same circuit image, but corrupted now by pepper noise with probability of 0.1. Similarly, Fig. 5.8(b) shows the image corrupted by salt noise with the same probability. Figure 5.8(c) shows the result of filtering Fig. 5.8(a) using a contraharmonic mean filter with $Q = 1.5$, and Fig. 5.8(d)



a b
c d

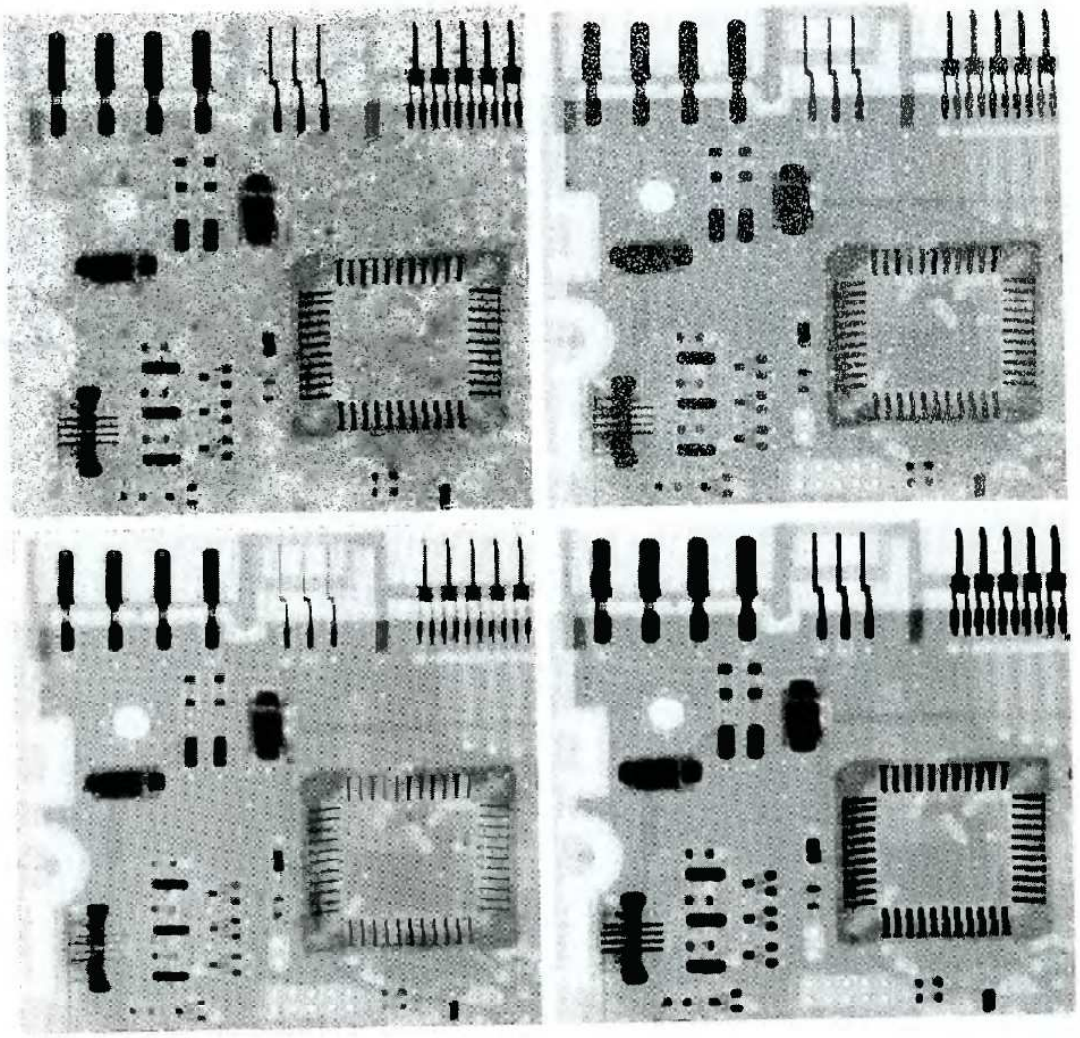
FIGURE 5.7 (a) X-ray image. (b) Image corrupted by additive Gaussian noise. (c) Result of filtering with an arithmetic mean filter of size 3×3. (d) Result of filtering with a geometric mean filter of the same size. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

shows the result of filtering Fig. 5.8(b) with $Q = -1.5$. Both filters did a good job in reducing the effect of the noise. The positive-order filter did a better job of cleaning the background, at the expense of blurring the dark areas. The opposite was true of the negative-order filter.

In general, the arithmetic and geometric mean filters (particularly the latter) are well suited for random noise like Gaussian or uniform noise. The contraharmonic filter is well suited for impulse noise, but it has the disadvantage that it must be known whether the noise is dark or light in order to select the proper sign for Q. The results of choosing the wrong sign for Q can be disastrous, as Fig. 5.9 shows. Some of the filters discussed in the following section eliminate this shortcoming. ■

5.3.2 Order-Statistics Filters

Order-statistics filters were introduced in Section 3.6.2. We now expand the discussion in that section and introduce some additional order-statistics filters. As noted in Section 3.6.2, order-statistics filters are spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter. The response of the filter at any point is determined by the ranking result.



a b
c d

FIGURE 5.8 (a) Image corrupted by pepper noise with a probability of 0.1. (b) Image corrupted by salt noise with the same probability. (c) Result of filtering (a) with a 3×3 contraharmonic filter of order 1.5. (d) Result of filtering (b) with $Q = -1.5$.

Median filter

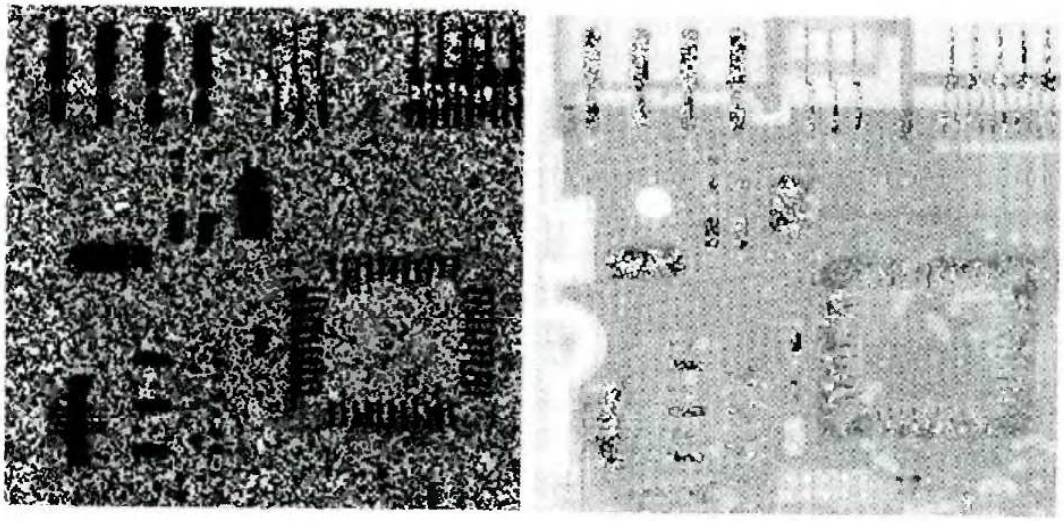
The best-known order-statistics filter is the *median filter*, which, as its name implies, replaces the value of a pixel by the median of the gray levels in the neighborhood of that pixel:

$$\hat{f}(x, y) = \underset{(s,t)\in S_{xy}}{\text{median}}\{g(s, t)\}. \tag{5.3-7}$$

The original value of the pixel is included in the computation of the median. Median filters are quite popular because, for certain types of random noise, they provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filters of similar size. Median filters are particularly effective in the presence of both bipolar and unipolar impulse noise. In fact, as Example 5.3 shows, the median filter yields excellent results for images corrupted by this type of noise. Computation of the median and implementation of this filter are discussed in detail in Section 3.6.2.

Max and min filters

Although the median filter is by far the order-statistics filter most used in image processing, it is by no means the only one. The median represents the 50th per-



a b

FIGURE 5.9 Results of selecting the wrong sign in contraharmonic filtering. (a) Result of filtering Fig. 5.8(a) with a contraharmonic filter of size 3×3 and $Q = -1.5$. (b) Result of filtering 5.8(b) with $Q = 1.5$.

centile of a ranked set of numbers, but the reader will recall from basic statistics that ranking lends itself to many other possibilities. For example, using the 100th percentile results in the so-called *max filter*, given by

$$\hat{f}(x, y) = \max_{(s,t)\in S_{xy}} \{g(s, t)\}. \tag{5.3-8}$$

This filter is useful for finding the brightest points in an image. Also, because pepper noise has very low values, it is reduced by this filter as a result of the max selection process in the subimage area S_{xy}.

The 0th percentile filter is the *min filter*:

$$\hat{f}(x, y) = \min_{(s,t)\in S_{xy}} \{g(s, t)\}. \tag{5.3-9}$$

This filter is useful for finding the darkest points in an image. Also, it reduces salt noise as a result of the min operation.

Midpoint filter

The midpoint filter simply computes the midpoint between the maximum and minimum values in the area encompassed by the filter:

$$\hat{f}(x, y) = \frac{1}{2}\Big[\max_{(s,t)\in S_{xy}} \{g(s, t)\} + \min_{(s,t)\in S_{xy}} \{g(s, t)\}\Big]. \tag{5.3-10}$$

Note that this filter combines order statistics and averaging. This filter works best for randomly distributed noise, like Gaussian or uniform noise.

Alpha-trimmed mean filter

Suppose that we delete the $d/2$ lowest and the $d/2$ highest gray-level values of $g(s, t)$ in the neighborhood S_{xy}. Let $g_r(s, t)$ represent the remaining $mn - d$ pixels. A filter formed by averaging these remaining pixels is called an *alpha-trimmed mean* filter:

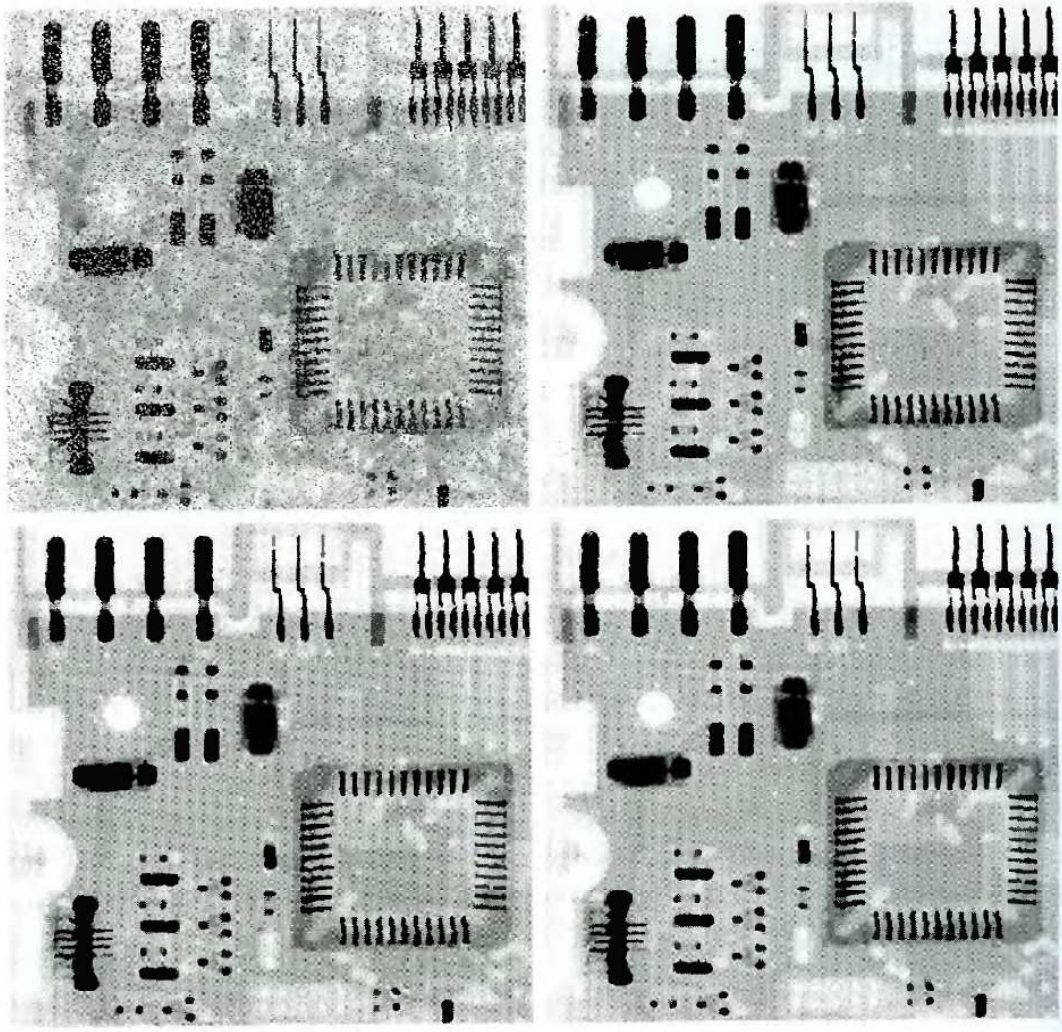
$$\hat{f}(x, y) = \frac{1}{mn - d} \sum_{(s,t)\in S_{xy}} g_r(s, t) \tag{5.3-11}$$

where the value of d can range from 0 to $mn - 1$. When $d = 0$, the alpha-trimmed filter reduces to the arithmetic mean filter discussed in the previous section. If we choose $d = (mn - 1)/2$, the filter becomes a median filter. For other values of d, the alpha-trimmed filter is useful in situations involving multiple types of noise, such as a combination of salt-and-pepper and Gaussian noise.

EXAMPLE 5.3: Illustration of order-statistics filters.

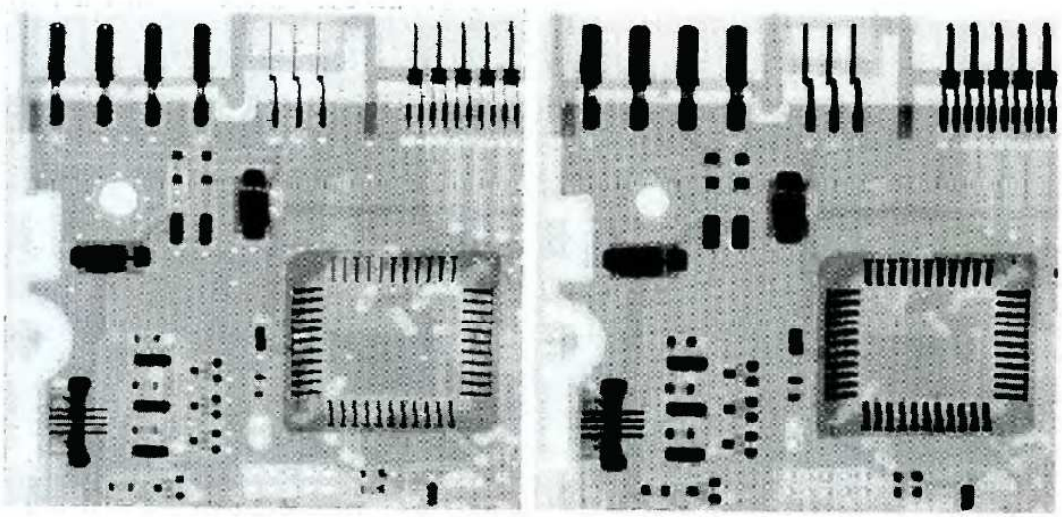
■ Figure 5.10(a) shows the circuit image corrupted by impulse noise with probabilities $P_a = P_b = 0.1$. Figure 5.10(b) shows the result of median filtering with a filter of size 3×3. The improvement over Fig. 5.10(a) is significant, but several noise points are still visible. A second pass [of the image in Fig. 5.10(b)] with the median filter removed most of these points, leaving only few, barely visible noise points. These are removed with a third pass of the filter. These results are good examples of the power of median filtering in handling impulselike additive noise. Keep in mind that repeated passes of a median filter tend to blur the image, so it is desirable to keep the number of passes as low as possible.

Figure 5.11(a) shows the result of applying the max filter to the pepper noise image of Fig. 5.8(a). The filter did a reasonable job of removing the pepper



a b
c d

FIGURE 5.10 (a) Image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.1$. (b) Result of one pass with a median filter of size 3×3. (c) Result of processing (b) with this filter. (d) Result of processing (c) with the same filter.



a b

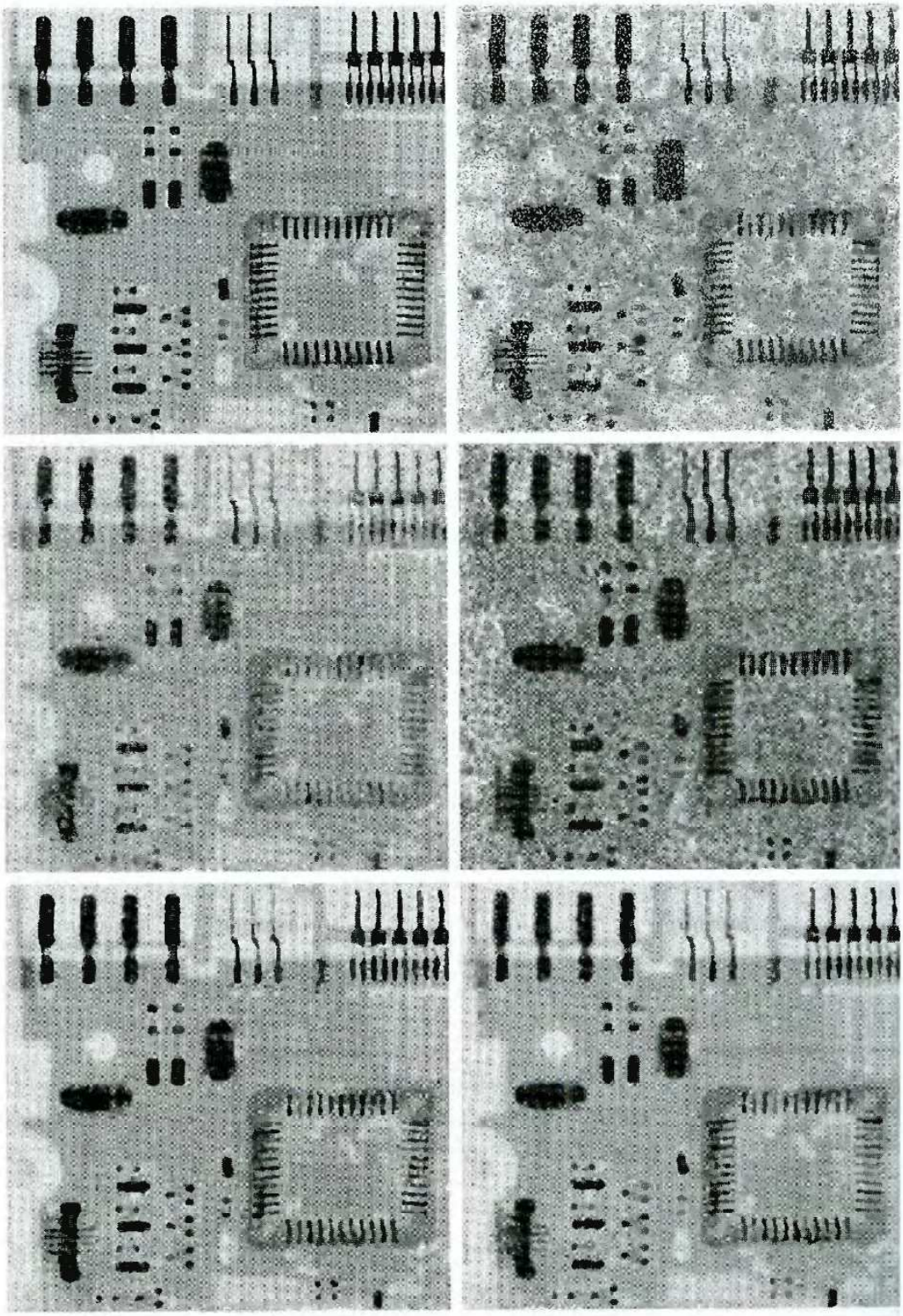
FIGURE 5.11 (a) Result of filtering Fig. 5.8(a) with a max filter of size 3×3. (b) Result of filtering 5.8(b) with a min filter of the same size.

noise, but we note that it also removed (set to a light gray level) some dark pixels from the borders of the dark objects. Figure 5.11(b) shows the result of applying the min filter to the image in Fig. 5.8(b). In this particular case, the min filter did a better job than the max filter on noise removal, but it removed some white points around the border of light objects. These made the light objects smaller and some of the dark objects larger (like the connector fingers in the top of the image) because white points around these objects were set to a dark level.

The alpha-trimmed filter is illustrated next. Figure 5.12(a) shows the circuit image corrupted this time by additive, uniform noise of variance 800 and zero mean. This is a high level of noise corruption that is made worse by further addition of salt-and-pepper noise with $P_a = P_b = 0.1$, as Fig. 5.12(b) shows. The high level of noise in this image warrants use of larger filters. Figures 5.12(c) through (f) show the results obtained using arithmetic mean, geometric mean, median, and alpha-trimmed mean (with $d = 5$) filters of size 5×5. As expected, the arithmetic and geometric mean filters (especially the latter) do not do well because of the presence of impulse noise. The median and alpha-trimmed filters performed much better, with the alpha-trimmed filter giving slightly better noise reduction. Note, for example, that the fourth connector finger from the top, left, is slightly smoother in the alpha-trimmed result. This is not unexpected because, for a high value of d, the alpha-trimmed filter approaches the performance of the median filter, but still retains some smoothing capabilities. ■

5.3.3 Adaptive Filters

Once selected, the filters discussed thus far are applied to an image without regard for how image characteristics vary from one point to another. In this section we take a look at two simple *adaptive* filters whose behavior changes based on statistical characteristics of the image inside the filter region defined by the $m \times n$ rectangular window S_{xy}. As shown in the following discussion, adaptive filters are capable of performance superior to that of the filters discussed thus far. The price paid for improved filtering power is an increase in filter complexity. Keep in mind that we still are dealing with the case in which the degraded



a b
c d
e f

FIGURE 5.12 (a) Image corrupted by additive uniform noise. (b) Image additionally corrupted by additive salt-and-pepper noise. Image in (b) filtered with a 5×5: (c) arithmetic mean filter; (d) geometric mean filter; (e) median filter; and (f) alpha-trimmed mean filter with $d = 5$.

image is equal to the original image plus noise. No other types of degradations are being considered yet.

Adaptive, local noise reduction filter

The simplest statistical measures of a random variable are its mean and variance. These are reasonable parameters on which to base an adaptive filter because they are quantities closely related to the appearance of an image. The mean gives a measure of average gray level in the region over which the mean is computed, and the variance gives a measure of average contrast in that region.

Our filter is to operate on a local region, S_{xy}. The response of the filter at any point (x, y) on which the region is centered is to be based on four quantities: (a) $g(x, y)$, the value of the noisy image at (x, y); (b) σ_η^2, the variance of the noise corrupting $f(x, y)$ to form $g(x, y)$; (c) m_L, the local mean of the pixels in S_{xy}; and (d) σ_L^2, the local variance of the pixels in S_{xy}. We want the behavior of the filter to be as follows:

1. If σ_η^2 is zero, the filter should return simply the value of $g(x, y)$. This is the trivial, zero-noise case in which $g(x, y)$ is equal to $f(x, y)$.
2. If the local variance is high relative to σ_η^2, the filter should return a value close to $g(x, y)$. A high local variance typically is associated with edges, and these should be preserved.
3. If the two variances are equal, we want the filter to return the arithmetic mean value of the pixels in S_{xy}. This condition occurs when the local area has the same properties as the overall image, and local noise is to be reduced simply by averaging.

An adaptive expression for obtaining $\hat{f}(x, y)$ based on these assumptions may be written as

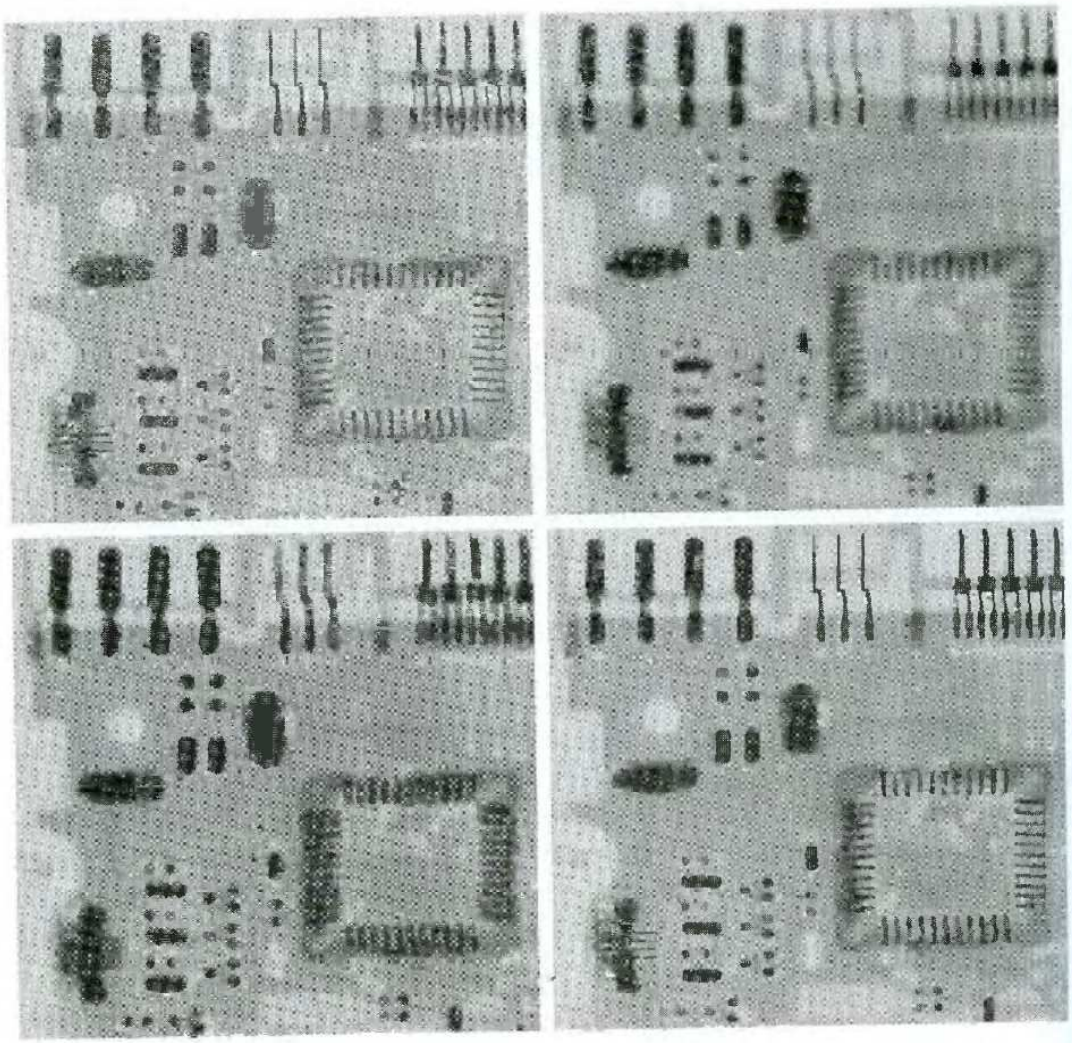
$$\hat{f}(x, y) = g(x, y) - \frac{\sigma_\eta^2}{\sigma_L^2}\left[g(x, y) - m_L\right]. \tag{5.3-12}$$

The only quantity that needs to be known or estimated is the variance of the overall noise, σ_η^2. The other parameters are computed from the pixels in S_{xy} at each location (x, y) on which the filter window is centered. A tacit assumption in Eq. (5.3-12) is that $\sigma_\eta^2 \leq \sigma_L^2$. The noise in our model is additive and position independent, so this is a reasonable assumption to make because S_{xy} is a subset of $g(x, y)$. However, we seldom have exact knowledge of σ_η^2. Therefore, it is possible for this condition to be violated in practice. For that reason, a test should be built into an implementation of Eq. (5.3-12) so that the ratio is set to 1 if the condition $\sigma_\eta^2 > \sigma_L^2$ occurs. This makes this filter nonlinear. However, it prevents nonsensical results (i.e., negative gray levels, depending on the value of m_L) due to a potential lack of knowledge about the variance of the image noise. Another approach is to allow the negative values to occur, and then rescale the gray level values at the end. The result then would be a loss of dynamic range in the image.

EXAMPLE 5.4: Illustration of adaptive, local noise-reduction filtering.

■ Figure 5.13(a) shows the circuit image, corrupted this time by additive Gaussian noise of zero mean and a variance of 1000. This is a significant level of noise corruption, but it makes an ideal test bed on which to compare relative filter performance. Figure 5.13(b) is the result of processing the noisy image with an arithmetic mean filter of size 7×7. The noise is smoothed out, but at the cost of significant blurring in the image. Similar comments are applicable to Fig. 5.13(c), which shows the result of processing the noisy image with a geometric mean filter, also of size 7×7. The differences between these two filtered images are analogous to those we discussed in Example 5.2; only the degree of blurring is different.

Figure 5.13(d) shows the result of using the adaptive filter of Eq. (5.3-12) with $\sigma_\eta^2 = 1000$. The improvements in this result compared to the two previous filters are significant. In terms of overall noise reduction, the adaptive filter achieved results similar to the arithmetic and geometric mean filters. However, the image filtered with the adaptive filter is much sharper. For example, the connector fingers at the top of the image are sharper in Fig. 5.13(d). Other features, such as holes and the eight legs of the dark component on the



a b
c d

FIGURE 5.13 (a) Image corrupted by additive Gaussian noise of zero mean and variance 1000. (b) Result of arithmetic mean filtering. (c) Result of geometric mean filtering. (d) Result of adaptive noise reduction filtering. All filters were of size 7×7.

lower left-hand side of the image, are significantly clearer in Fig. 5.13(d). These results are typical of what can be achieved with an adaptive filter. As mentioned at the beginning of this section, the price paid for the improved performance is additional filter complexity.

The preceding results used a value for σ_η^2 that matched the variance of the noise exactly. If this quantity is not known and an estimate is used that is too low, the algorithm will return an image that closely resembles the original because the corrections will be smaller than they should be. Estimates that are too high will cause the ratio of the variances to be clipped at 1.0, and the algorithm will subtract the mean from the image more frequently than it would normally do so. If negative values are allowed and the image is rescaled at the end, the result will be a loss of dynamic range, as mentioned previously. ■

Adaptive median filter

The median filter discussed in Section 5.3.2 performs well as long as the spatial density of the impulse noise is not large (as a rule of thumb, P_a and P_b less than 0.2). It is shown in this section that adaptive median filtering can handle impulse noise with probabilities even larger than these. An additional benefit of the adaptive median filter is that it seeks to preserve detail while smoothing nonimpulse noise, something that the "traditional" median filter does not do. As in all the filters discussed in the preceding sections, the adaptive median filter also works in a rectangular window area S_{xy}. Unlike those filters, however, the adaptive median filter changes (increases) the size of S_{xy} during filter operation, depending on certain conditions listed in this section. Keep in mind that the output of the filter is a single value used to replace the value of the pixel at (x, y), the particular point on which the window S_{xy} is centered at a given time.

Consider the following notation:

$$
\begin{aligned}
z_{\min} &= \text{minimum gray level value in } S_{xy} \\
z_{\max} &= \text{maximum gray level value in } S_{xy} \\
z_{\text{med}} &= \text{median of gray levels in } S_{xy} \\
z_{xy} &= \text{gray level at coordinates } (x, y) \\
S_{\max} &= \text{maximum allowed size of } S_{xy}.
\end{aligned}
$$

The adaptive median filtering algorithm works in two levels, denoted level A and level B, as follows:

Level A:
$A1 = z_{\text{med}} - z_{\min}$
$A2 = z_{\text{med}} - z_{\max}$
If $A1 > 0$ AND $A2 < 0$, Go to level B
Else increase the window size
If window size $\leq S_{\max}$ repeat level A
Else output z_{xy}.

Level B:
$B1 = z_{xy} - z_{\min}$
$B2 = z_{xy} - z_{\max}$
If $B1 > 0$ AND $B2 < 0$, output z_{xy}
Else output z_{med}.

The key to understanding the mechanics of this algorithm is to keep in mind that it has three main purposes: to remove salt-and-pepper (impulse) noise, to provide smoothing of other noise that may not be impulsive, and to reduce distortion, such as excessive thinning or thickening of object boundaries. The values $z_{\min}$ and $z_{\max}$ are considered statistically by the algorithm to be "impulselike" noise components, even if these are not the lowest and highest possible pixel values in the image.

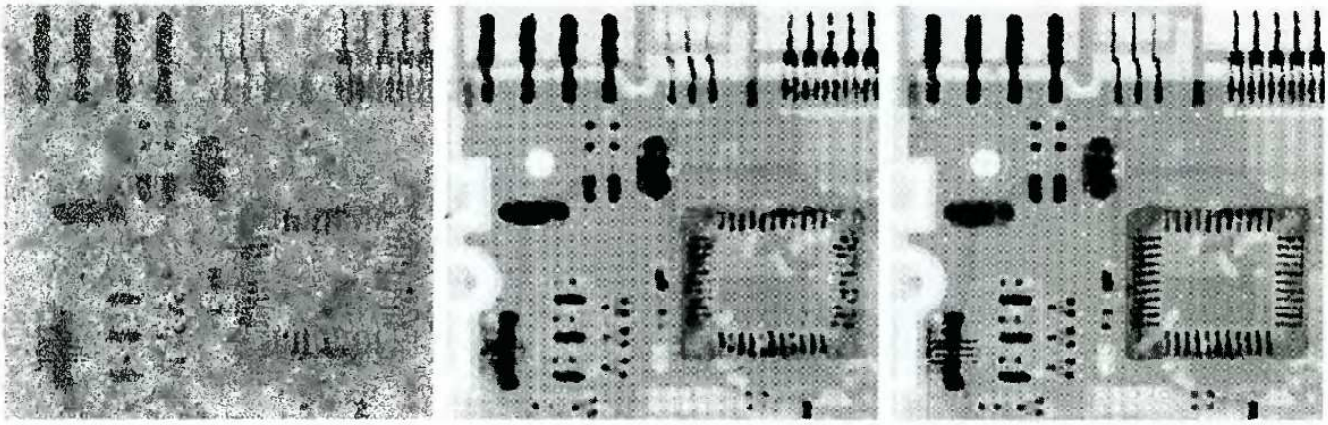
With these observations in mind, we see that the purpose of level A is to determine if the median filter output, z_{med}, is an impulse (black *or* white) or not. If the condition $z_{\min} < z_{\text{med}} < z_{\max}$ holds, then z_{med} cannot be an impulse for the reason mentioned in the previous paragraph. In this case, we go to level B and test to see if the point in the center of the window, z_{xy}, is itself an impulse (recall that z_{xy} is the point being processed). If the condition $B1 > 0$ AND $B2 < 0$ is true, then $z_{\min} < z_{xy} < z_{\max}$, and z_{xy} cannot be an impulse for the same reason that z_{med} was not. In this case, the algorithm outputs the unchanged pixel value, z_{xy}. By not changing these "intermediate-level" points, distortion is reduced in the image. If the condition $B1 > 0$ AND $B2 < 0$ is false, then either $z_{xy} = z_{\min}$ or $z_{xy} = z_{\max}$. In either case, the value of the pixel is an extreme value and the algorithm outputs the median value z_{med}, which we know from level A is not a noise impulse. The last step is what the standard median filter does. The problem is that the standard median filter replaces every point in the image by the median of the corresponding neighborhood. This causes unnecessary loss of detail.

Continuing with the explanation, suppose that level A *does* find an impulse (i.e., it fails the test that would cause it to branch to level B). The algorithm then increases the size of the window and repeats level A. This looping continues until the algorithm either finds a median value that is not an impulse (and branches to level B), or the maximum window size is reached. If the maximum window size is reached, the algorithm returns the value of z_{xy}. Note that there is no guarantee that this value is not an impulse. The smaller the noise probabilities P_a and/or P_b are, or the larger $S_{\max}$ is allowed to be, the less likely it is that a premature exit condition will occur. This is plausible. As the density of the impulses increases, it stands to reason that we would need a larger window to "clean up" the noise spikes.

Every time the algorithm outputs a value, the window S_{xy} is moved to the next location in the image. The algorithm then is reinitialized and applied to the pixels in the new location. As indicated in Problem 3.20, the median value can be updated iteratively using only the new pixels, thus reducing computational overhead.

EXAMPLE 5.5: Illustration of adaptive median filtering.

■ Figure 5.14(a) shows the circuit image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.25$, which is 2.5 times the noise level used in Fig. 5.10(a). Here the noise level is high enough to obscure most of the detail in the image. As a basis for comparison, the image was filtered first using the smallest median filter required to remove most visible traces of impulse noise. A 7×7 median filter was required to do this, and the result is shown in



a b c

FIGURE 5.14 (a) Image corrupted by salt-and-pepper noise with probabilities $P_a = P_b = 0.25$. (b) Result of filtering with a 7×7 median filter. (c) Result of adaptive median filtering with $S_{max} = 7$.

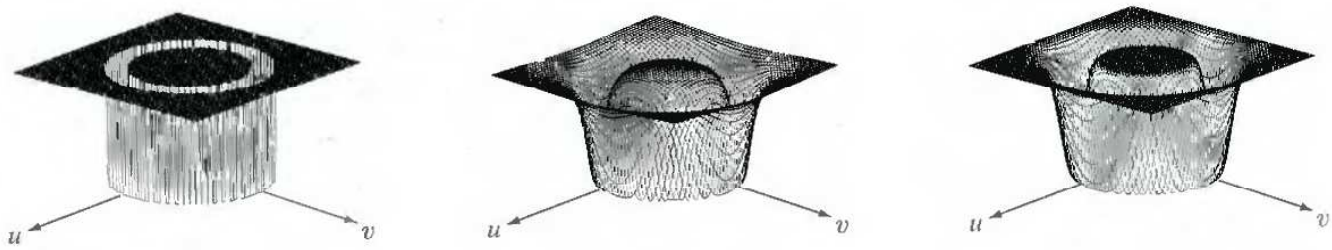
Fig. 5.14(b). Although the noise was effectively removed, the filter caused significant loss of detail in the image. For instance, some of the connector fingers at the top of the image appear distorted or broken. Other image details are similarly distorted.

Figure 5.14(c) shows the result of using the adaptive median filter with $S_{max} = 7$. The level of noise removal was similar to the median filter. However, the adaptive filter preserved sharpness and detail to the point where improvements over Fig. 5.14(b) are quite significant. The connector fingers are less distorted, and some other features that were either obscured or distorted beyond recognition by the median filter appear sharper and better defined in Fig. 5.14(c). Two notable examples are the feed-through small white holes throughout the board, and the dark component with eight legs in the bottom, left quadrant of the image.

Considering the high level of noise in Fig. 5.14(a), the adaptive algorithm performed quite well. The choice of maximum allowed window size depends on the application, but a reasonable starting value can be estimated by experimenting with various sizes of the standard median filter first. This will establish a visual baseline regarding expectations on the performance of the adaptive algorithm. ■

5.4 Periodic Noise Reduction by Frequency Domain Filtering

In Chapter 4 we discussed lowpass and highpass frequency domain filters as fundamental tools for image enhancement. In this section we discuss the more specialized bandreject, bandpass, and notch filters as tools for periodic noise reduction or removal.



a b c

FIGURE 5.15 From left to right, perspective plots of ideal, Butterworth (of order 1), and Gaussian bandreject filters.

5.4.1 Bandreject Filters

Bandreject filters remove or attenuate a band of frequencies about the origin of the Fourier transform. An ideal bandreject filter is given by the expression

$$H(u,v) = \begin{cases} 1 & \text{if } D(u,v) < D_0 - \dfrac{W}{2} \\ 0 & \text{if } D_0 - \dfrac{W}{2} \le D(u,v) \le D_0 + \dfrac{W}{2} \\ 1 & \text{if } D(u,v) > D_0 + \dfrac{W}{2} \end{cases} \tag{5.4-1}$$

where $D(u,v)$ is the distance from the origin of the centered frequency rectangle, as given in Eq. (4.3-3), W is the width of the band, and D_0 is its radial center.

Similarly, a Butterworth bandreject filter of order n is given by the expression

$$H(u,v) = \frac{1}{1 + \left[\dfrac{D(u,v)W}{D^2(u,v) - D_0^2}\right]^{2n}} \tag{5.4-2}$$

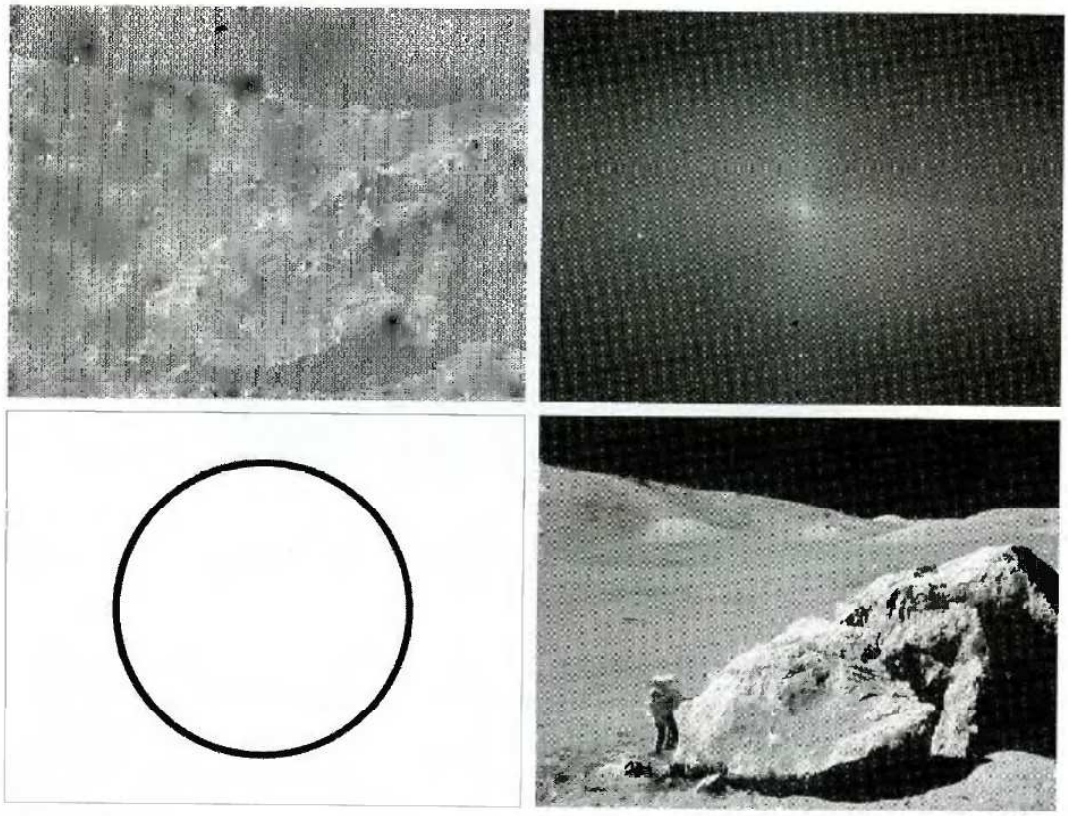
and a Gaussian bandreject filter is given by

$$H(u,v) = 1 - e^{-\frac{1}{2}\left[\frac{D^2(u,v) - D_0^2}{D(u,v)W}\right]^2}. \tag{5.4-3}$$

Figure 5.15 shows perspective plots of these three filters.

EXAMPLE 5.6: Use of bandreject filtering for periodic noise removal.

■ One of the principal applications of bandreject filtering is for noise removal in applications where the general location of the noise component(s) in the frequency domain is approximately known. A good example is an image corrupted by additive periodic noise that can be approximated as two-dimensional sinusoidal functions. It is not difficult to show that the Fourier transform of a sine consists of two impulses that are mirror images of each other about the origin of the transform. Their locations are given in Table 4.1. The impulses are both imaginary (the real part of the Fourier transform of a sine is zero) and are complex conjugates of each other. We will have more to say about this topic in Sections 5.4.3 and 5.4.4. Our purpose at the moment is to illustrate bandreject filtering.



a b
c d

FIGURE 5.16 (a) Image corrupted by sinusoidal noise. (b) Spectrum of (a). (c) Butterworth bandreject filter (white represents 1). (d) Result of filtering. (Original image courtesy of NASA.)

Figure 5.16(a), which is the same as Fig. 5.5(a), shows an image heavily corrupted by sinusoidal noise of various frequencies. The noise components are easily seen as symmetric pairs of bright dots in the Fourier spectrum shown in Fig. 5.16(b). In this example, the components lie on an approximate circle about origin of the transform, so a circularly symmetric bandreject filter is a good choice. Figure 5.16(c) shows a Butterworth bandreject filter of order 4, with the appropriate radius and width to enclose completely the noise impulses. Since it is desirable in general to remove as little as possible from the transform, sharp, narrow filters are common in bandreject filtering. The result of filtering Fig. 5.16(a) with this filter is shown in Fig. 5.16(d). The improvement is quite evident. Even small details and textures were restored effectively by this simple filtering approach. It is worth noting also that it would not be possible to get equivalent results by a direct spatial domain filtering approach using small convolution masks. ■

5.4.2 Bandpass Filters

A *bandpass* filter performs the opposite operation of a bandreject filter. In Section 4.4 we showed how a highpass filter can be obtained from a corresponding lowpass filter by using Eq. (4.4-1). Similarly, the transfer function $H_{bp}(u, v)$ of a bandpass filter is obtained from a corresponding bandreject filter with transfer function $H_{br}(u, v)$ by using the equation

$$H_{bp}(u, v) = 1 - H_{br}(u, v). \tag{5.4-4}$$

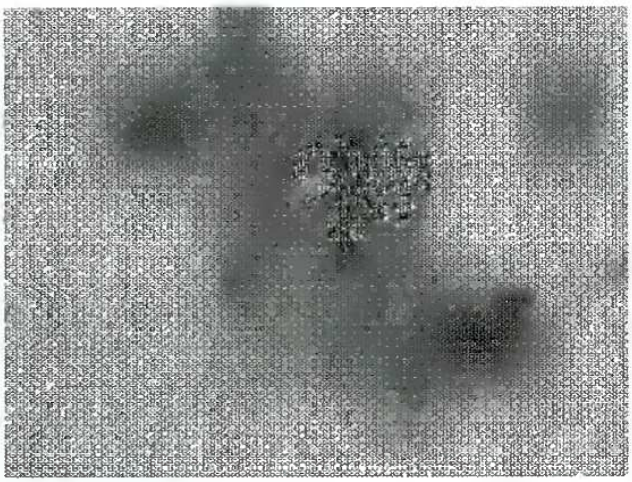


FIGURE 5.17 Noise pattern of the image in Fig. 5.16(a) obtained by bandpass filtering.

It is left as an exercise for the reader (Problem 5.12) to derive expressions for the bandpass filters corresponding to Eqs. (5.4-1) through (5.4-3).

EXAMPLE 5.7: Bandpass filtering for extracting noise patterns.

■ Performing straight bandpass filtering on an image is not a common procedure because it generally removes too much image detail. However, bandpass filtering is quite useful in isolating the effect on an image of selected frequency bands. This is illustrated in Fig. 5.17. This image was generated by (1) using Eq. (5.4-4) to obtain the bandpass filter corresponding to the bandreject filter used in the previous example; and (2) taking the inverse transform of the bandpass-filtered transform. Most image detail was lost, but the information that remains is most useful, as it is clear that the noise pattern recovered using this method is quite close to the noise that corrupted the image in Fig. 5.16(a). In other words, bandpass filtering helped isolate the noise pattern. This is a useful result because it simplifies analysis of the noise, reasonably independently of image content. ■

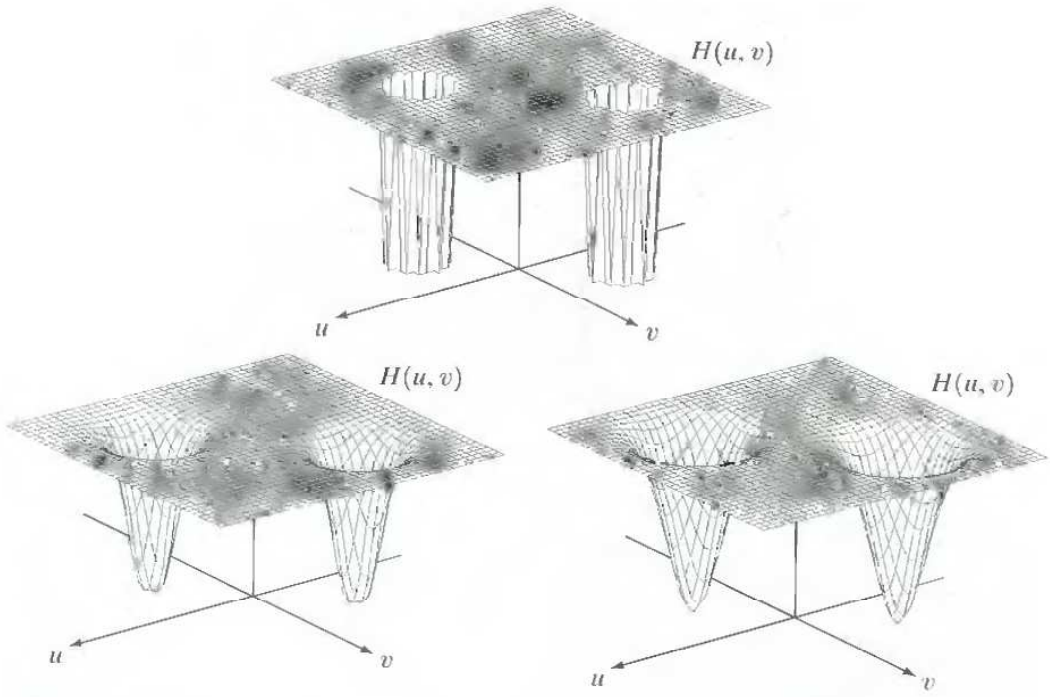
5.4.3 Notch Filters

A *notch* filter rejects (or passes) frequencies in predefined neighborhoods about a center frequency. Figure 5.18 shows 3-D plots of ideal, Butterworth, and Gaussian notch (reject) filters. Due to the symmetry of the Fourier transform, notch filters must appear in symmetric pairs about the origin in order to obtain meaningful results. The one exception to this rule is if the notch filter is located at the origin, in which case it appears by itself. Although we show only one pair for illustrative purposes, the number of pairs of notch filters that can be implemented is arbitrary. The shape of the notch areas also can be arbitrary (e.g., rectangular).

The transfer function of an ideal notch reject filter of radius D_0, with centers at (u_0, v_0) and, by symmetry, at $(-u_0, -v_0)$, is

$$H(u,v) = \begin{cases} 0 & \text{if } D_1(u,v) \le D_0 \quad \text{or} \quad D_2(u,v) \le D_0 \\ 1 & \text{otherwise} \end{cases} \tag{5.4-5}$$

where



a
b c

FIGURE 5.18 Perspective plots of (a) ideal, (b) Butterworth (of order 2), and (c) Gaussian notch (reject) filters.

$$D_1(u, v) = \left[(u - M/2 - u_0)^2 + (v - N/2 - v_0)^2\right]^{1/2} \tag{5.4-6}$$

and

$$D_2(u, v) = \left[(u - M/2 + u_0)^2 + (v - N/2 + v_0)^2\right]^{1/2}. \tag{5.4-7}$$

As usual, the assumption is that the center of the frequency rectangle has been shifted to the point $(M/2, N/2)$, according to the filtering procedure outlined in Section 4.2.3. Therefore, the values of (u_0, v_0) are with respect to the shifted center.

The transfer function of a Butterworth notch reject filter of order n is given by

$$H(u, v) = \frac{1}{1 + \left[\dfrac{D_0^2}{D_1(u, v)D_2(u, v)}\right]^n} \tag{5.4-8}$$

where $D_1(u, v)$ and $D_2(u, v)$ are given in Eqs. (5.4-6) and (5.4-7), respectively. A Gaussian notch reject filter has the form

$$H(u, v) = 1 - e^{-\frac{1}{2}\left[\frac{D_1(u,v)D_2(u,v)}{D_0^2}\right]} \tag{5.4-9}$$

It is interesting to note that these three filters become highpass filters if $u_0 = v_0 = 0$.

As shown in the previous section for bandpass filters, we can obtain notch filters that *pass*, rather than suppress, the frequencies contained in the notch areas. Since these filters perform exactly the opposite function as the notch reject filters given in Eqs. (5.4-5), (5.4-8), and (5.4-9), their transfer functions are given by

$$H_{np}(u, v) = 1 - H_{nr}(u, v) \tag{5.4-10}$$

where $H_{np}(u, v)$ is the transfer function of the notch pass filter corresponding to the notch reject filter with transfer function $H_{nr}(u, v)$. It is left as an exercise for the reader (Problem 5.13) to derive equations for the notch pass filters corresponding to the reject filters just discussed, and to show that they become lowpass filters when $u_0 = v_0 = 0$.

EXAMPLE 5.8: Removal of periodic noise by notch filtering.

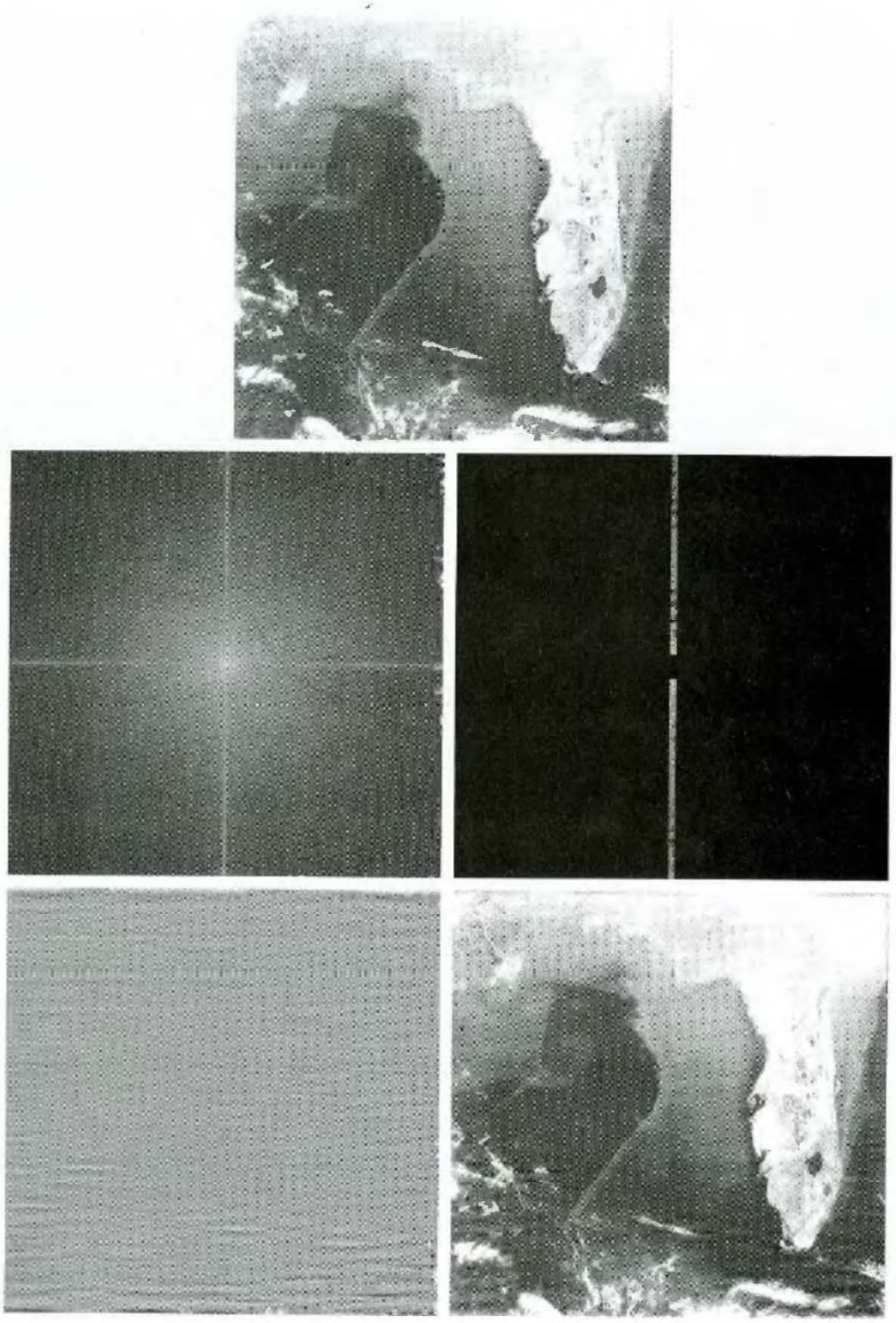
■ Figure 5.19(a) shows the same image as Fig. 4.21(a). When we discussed lowpass filtering of that image in Section 4.3.4, we indicated that there were better ways to reduce the effect of the scan lines. The notch filtering approach that follows reduces the noise in this image, without introducing appreciable blurring. Unless blurring is desirable for reasons we discussed in Section 4.3, notch filtering is preferable if a suitable filter can be found.

Just by looking at the nearly horizontal lines of the noise pattern in Fig. 5.19(a), we expect its contribution in the frequency domain to be concentrated along the vertical axis. However, the noise is not dominant enough to have a clear pattern along this axis, as is evident from the spectrum shown in Fig. 5.19(b). We can get an idea of what the noise contribution looks like by constructing a simple ideal notch pass filter along the vertical axis of the Fourier transform, as shown in Fig. 5.19(c). The spatial representation of the noise pattern (inverse transform of the notch pass–filtered result) is shown in Fig. 5.19(d). This noise pattern corresponds closely to the pattern in Fig. 5.19(a). Having thus constructed a suitable notch pass filter that isolates the noise to a reasonable degree, we can obtain the corresponding notch reject filter from Eq. (5.4-10). The result of processing the image with the notch reject filter is shown in Fig. 5.19(e). This image contains significantly fewer visible noise scan lines than Fig. 5.19(a). ■

5.4.4 Optimum Notch Filtering

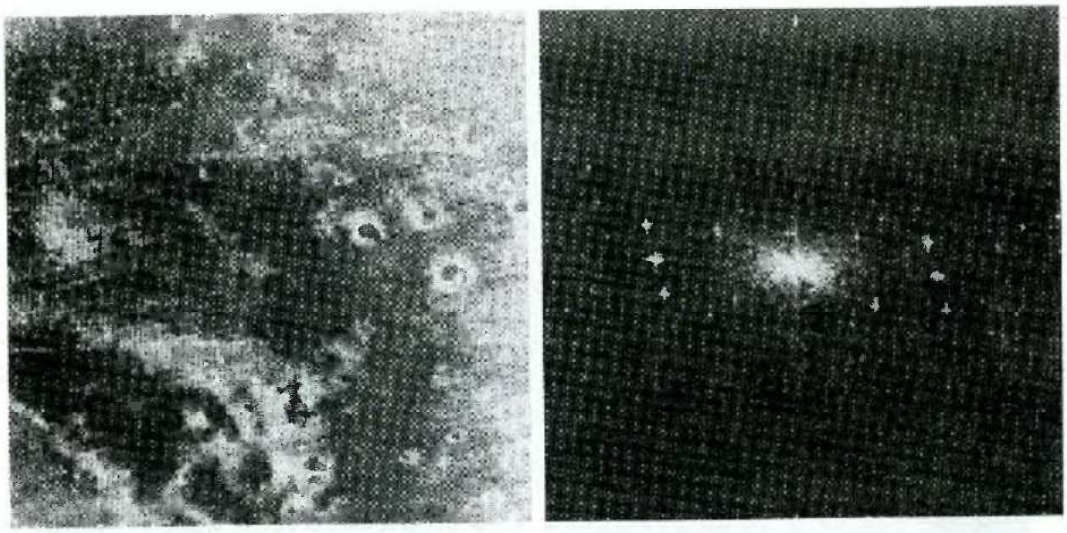
Clearly defined interference patterns are not common. Images derived from electro-optical scanners, such as those used in space and aerial imaging, sometimes are corrupted by coupling and amplification of low-level signals in the scanners' electronic circuitry. The resulting images tend to contain pronounced, 2-D periodic structures superimposed on the scene data with patterns more complex than those we have studied thus far.

Figure 5.20(a), an example of this type of periodic image degradation, shows a digital image of the Martian terrain taken by the *Mariner 6* spacecraft. The interference pattern is quite similar to the one shown in Fig. 5.16(a), but the former pattern is considerably more subtle and, consequently, harder to detect in the frequency plane. Figure 5.20(b) shows the Fourier spectrum of the image in question. The starlike components were caused by the interference, and several pairs of components are present, indicating that the pattern contained more than just one sinusoidal component.



a
b c
d e

FIGURE 5.19 (a) Satellite image of Florida and the Gulf of Mexico (note horizontal sensor scan lines). (b) Spectrum of (a). (c) Notch pass filter shown superimposed on (b). (d) Inverse Fourier transform of filtered image, showing noise pattern in the spatial domain. (e) Result of notch reject filtering. (Original image courtesy of NOAA.)



a b

FIGURE 5.20 (a) Image of the Martian terrain taken by *Mariner 6*. (b) Fourier spectrum showing periodic interference. (Courtesy of NASA.)

When several interference components are present, the methods discussed in the preceding sections are not always acceptable because they may remove too much image information in the filtering process (a highly undesirable feature when images are unique and/or expensive to acquire). In addition, the interference components generally are not single-frequency bursts. Instead, they tend to have broad skirts that carry information about the interference pattern. These skirts are not always easily detectable from the normal transform background. Alternative filtering methods that reduce the effect of these drawbacks are quite useful in many applications. The method discussed here is optimum, in the sense that it minimizes local variances of the restored estimate $\hat{f}(x, y)$.

The procedure consists of first isolating the principal contributions of the interference pattern and then subtracting a variable, weighted portion of the pattern from the corrupted image. Although we develop the procedure in the context of a specific application, the basic approach is quite general and can be applied to other restoration tasks in which multiple periodic interference is a problem.

The first step is to extract the principal frequency components of the interference pattern. This can be done by placing a notch pass filter, $H(u, v)$, at the location of each spike. If $H(u, v)$ is constructed to pass only components associated with the interference pattern, then, from the discussion in Sections 5.4.2 and 5.4.3, it follows that the Fourier transform of the interference noise pattern is given by the expression

$$N(u, v) = H(u, v)G(u, v) \tag{5.4-11}$$

where, as usual, $G(u, v)$, denotes the Fourier transform of the corrupted image.

Formation of $H(u, v)$ requires considerable judgment about what is or is not an interference spike. For this reason, the notch pass filter generally is constructed interactively by observing the spectrum of $G(u, v)$ on a display. After a particular filter has been selected, the corresponding pattern in the spatial domain is obtained from the expression

$$\eta(x, y) = \Im^{-1}\{H(u, v)G(u, v)\}. \tag{5.4-12}$$

Because the corrupted image is assumed to be formed by the addition of the uncorrupted image $f(x, y)$ and the interference, if $\eta(x, y)$ were known completely, subtracting the pattern from $g(x, y)$ to obtain $f(x, y)$ would be a simple matter, as discussed earlier in this chapter. The problem, of course, is that this filtering procedure usually yields only an approximation of the true pattern. The effect of components not present in the estimate of $\eta(x, y)$ can be minimized instead by subtracting from $g(x, y)$ a *weighted* portion of $\eta(x, y)$ to obtain an estimate of $f(x, y)$:

$$\hat{f}(x, y) = g(x, y) - w(x, y)\eta(x, y) \tag{5.4-13}$$

where, as before, $\hat{f}(x, y)$ is the estimate of $f(x, y)$ and $w(x, y)$ is to be determined. The function $w(x, y)$ is called a *weighting* or *modulation* function, and the objective of the procedure is to select this function so that the result is optimized in some meaningful way. One approach is to select $w(x, y)$ so that the variance of the estimate $\hat{f}(x, y)$ is minimized over a specified neighborhood of every point (x, y).

Consider a neighborhood of size $(2a + 1)$ by $(2b + 1)$ about a point (x, y). The "local" variance of $\hat{f}(x, y)$ at coordinates (x, y) can be estimated from the samples as follows:

$$\sigma^2(x, y) = \frac{1}{(2a + 1)(2b + 1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} [\hat{f}(x + s, y + t) - \bar{\hat{f}}(x, y)]^2 \tag{5.4-14}$$

where $\bar{\hat{f}}(x, y)$ is the average value of $\hat{f}$ in the neighborhood; that is,

$$\bar{\hat{f}}(x, y) = \frac{1}{(2a + 1)(2b + 1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} \hat{f}(x + s, y + t). \tag{5.4-15}$$

Points on or near the edge of the image can be treated by considering partial neighborhoods.

Substituting Eq. (5.4-13) into Eq. (5.4-14) yields

$$\begin{aligned}\sigma^2(x, y) = \frac{1}{(2a + 1)(2b + 1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} \{[g(x + s, y + t) \\ - w(x + s, y + t)\eta(x + s, y + t)] \\ - [\bar{g}(x, y) - \overline{w(x, y)\eta(x, y)}]\}^2.\end{aligned} \tag{5.4-16}$$

Assuming that $w(x, y)$ remains essentially constant over the neighborhood gives the approximation

$$w(x + s, y + t) = w(x, y) \tag{5.4-17}$$

for $-a \le s \le a$ and $-b \le t \le b$. This assumption also results in the expression

$$\overline{w(x, y)\eta(x, y)} = w(x, y)\bar{\eta}(x, y) \tag{5.4-18}$$

in the neighborhood. With these approximations, Eq. (5.4-16) becomes

$$\begin{aligned}\sigma^2(x, y) = \frac{1}{(2a + 1)(2b + 1)} \sum_{s=-a}^{a} \sum_{t=-b}^{b} \{[g(x + s, y + t) \\ - w(x, y)\eta(x + s, y + t)] \\ - [\bar{g}(x, y) - w(x, y)\bar{\eta}(x, y)]\}^2.\end{aligned} \tag{5.4-19}$$

To minimize $\sigma^2(x, y)$, we solve

$$\frac{\partial \sigma^2(x, y)}{\partial w(x, y)} = 0 \tag{5.4-20}$$

for $w(x, y)$. The result is

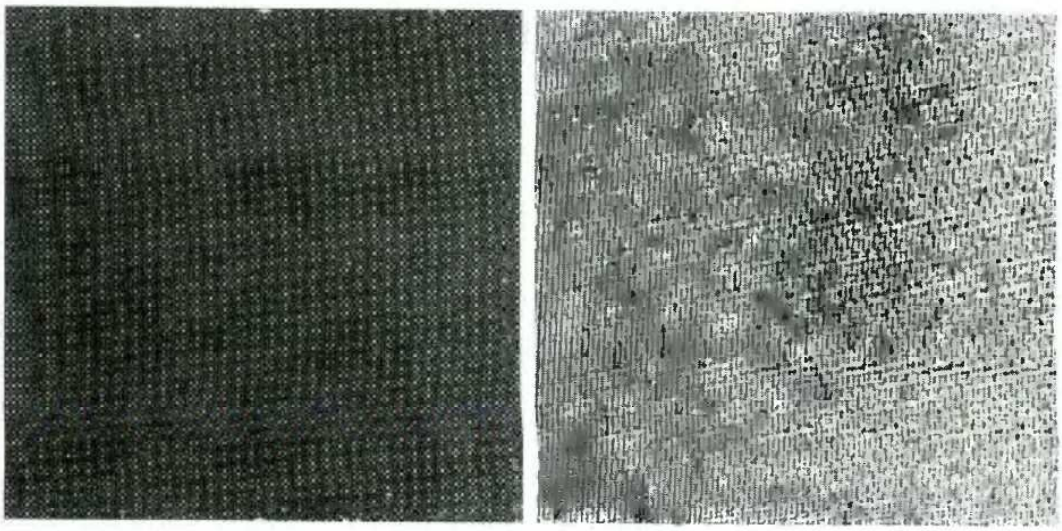
$$w(x, y) = \frac{\overline{g(x, y)\eta(x, y)} - \bar{g}(x, y)\bar{\eta}(x, y)}{\overline{\eta^2}(x, y) - \bar{\eta}^2(x, y)}. \tag{5.4-21}$$

To obtain the restored image $\hat{f}(x, y)$, we compute $w(x, y)$ from Eq. (5.4-21) and then make use of Eq. (5.4-13). As $w(x, y)$ is assumed to be constant in a neighborhood, computing this function for every value of x and y in the image is unnecessary. Instead, $w(x, y)$ is computed for *one* point in each nonoverlapping neighborhood (preferably the center point) and then used to process all the image points contained in that neighborhood.

EXAMPLE 5.9: Illustration of optimum notch filtering.

■ Figures 5.21 through 5.23 show the result of applying the preceding technique to the image shown in Fig. 5.20(a). This image is of size 512×512 pixels, and a neighborhood with $a = b = 15$ was selected. Figure 5.21 shows the Fourier spectrum of the corrupted image. The origin was not shifted to the center of the frequency plane in this particular case, so $u = v = 0$ is at the top, left corner of the transform image shown in Fig. 5.21. Figure 5.22(a) shows the spectrum of $N(u, v)$, where only the noise spikes are present. Figure 5.22(b) shows the

FIGURE 5.21 Fourier spectrum (without shifting) of the image shown in Fig. 5.20(a). (Courtesy of NASA.)



a b

FIGURE 5.22 (a) Fourier spectrum of $N(u, v)$, and (b) corresponding noise interference pattern $\eta(x, y)$. (Courtesy of NASA.)

interference pattern $\eta(x, y)$ obtained by taking the inverse Fourier transform of $N(u, v)$. Note the similarity between this pattern and the structure of the noise present in Fig. 5.20(a). Finally, Fig. 5.23 shows the processed image obtained by using Eq. (5.4-13). The periodic interference was removed for all practical purposes. ■

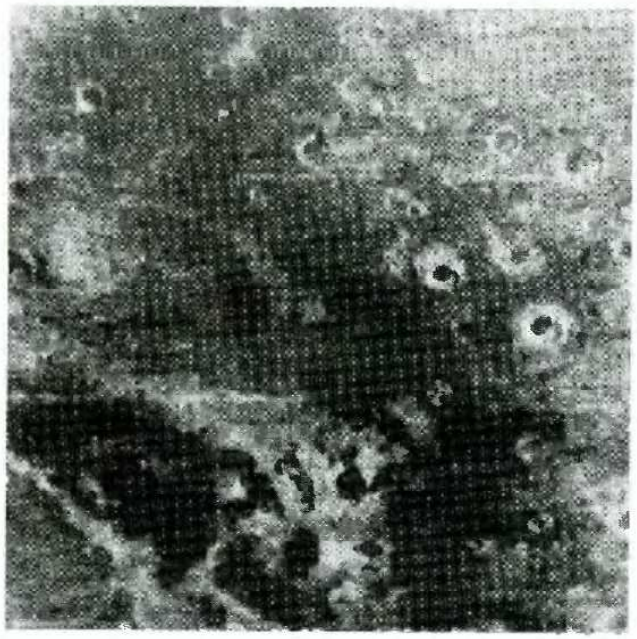


FIGURE 5.23 Processed image. (Courtesy of NASA.)

5.5 Linear, Position-Invariant Degradations

The input-output relationship in Fig. 5.1 before the restoration stage is expressed as

$$g(x, y) = H[f(x, y)] + \eta(x, y). \tag{5.5-1}$$

For the moment, let us assume that $\eta(x, y) = 0$ so that $g(x, y) = H[f(x, y)]$. Based on the discussion in Section 2.6, H is *linear* if

$$H[af_1(x, y) + bf_2(x, y)] = aH[f_1(x, y)] + bH[f_2(x, y)] \tag{5.5-2}$$

where a and b are scalars and $f_1(x, y)$ and $f_2(x, y)$ are any two input images.

See inside front cover
Consult the book web site for a brief review of linear system theory.

If $a = b = 1$, Eq. (5.5-2) becomes

$$H[f_1(x, y) + f_2(x, y)] = H[f_1(x, y)] + H[f_2(x, y)] \tag{5.5-3}$$

which is called the property of *additivity*. This property simply says that, if H is a linear operator, the response to a sum of two inputs is equal to the sum of the two responses.

With $f_2(x, y) = 0$, Eq. (5.5-2) becomes

$$H[af_1(x, y)] = aH[f_1(x, y)] \tag{5.5-4}$$

which is called the property of *homogeneity*. It says that the response to a constant multiple of any input is equal to the response to that input multiplied by the same constant. Thus a linear operator possesses both the property of additivity and the property of homogeneity.

An operator having the input-output relationship $g(x, y) = H[f(x, y)]$ is said to be *position* (or *space*) *invariant* if

$$H[f(x - \alpha, y - \beta)] = g(x - \alpha, y - \beta) \tag{5.5-5}$$

for any $f(x, y)$ and any α and β. This definition indicates that the response at any point in the image depends only on the *value* of the input at that point, not on its *position*.

With a slight (but equivalent) change in notation in the definition of the discrete impulse function in Eq. (4.2-33), $f(x, y)$ can be expressed in terms of a continuous impulse function:

$$f(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta)\delta(x - \alpha, y - \beta)\, d\alpha\, d\beta. \tag{5.5-6}$$

This, in fact, is the *definition* using continuous variables of a unit impulse located at coordinates (x, y).

Assume again for a moment that $\eta(x, y) = 0$. Then, substitution of Eq. (5.5-6) into Eq. (5.5-1) results in the expression

$$g(x, y) = H[f(x, y)] = H\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta)\delta(x - \alpha, y - \beta)\, d\alpha\, d\beta\right]. \tag{5.5-7}$$

If H is a linear operator and we extend the additivity property to integrals, then

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} H[f(\alpha, \beta)\delta(x - \alpha, y - \beta)]\, d\alpha\, d\beta. \tag{5.5-8}$$

Because $f(\alpha, \beta)$ is independent of x and y, and using the homogeneity property, it follows that

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta) H[\delta(x - \alpha, y - \beta)]\, d\alpha\, d\beta. \tag{5.5-9}$$

The term

$$h(x, \alpha, y, \beta) = H[\delta(x - \alpha, y - \beta)] \tag{5.5-10}$$

is called the *impulse response* of H. In other words, if $\eta(x, y) = 0$ in Eq. (5.5-1), then $h(x, \alpha, y, \beta)$ is the response of H to an impulse of strength 1 at coordinates (x, y). In optics, the impulse becomes a point of light and $h(x, \alpha, y, \beta)$ is commonly referred to as the *point spread function* (PSF). This name arises from the fact that all physical optical systems blur (spread) a point of light to some degree, with the amount of blurring being determined by the quality of the optical components.

Substituting Eq. (5.5-10) into Eq. (5.5-9) yields the expression

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta) h(x, \alpha, y, \beta)\, d\alpha\, d\beta \tag{5.5-11}$$

which is called the *superposition* (or *Fredholm*) *integral of the first kind*. This expression is a fundamental result that is at the core of linear system theory. It states that if the response of H to an impulse is known, the response to *any* input $f(\alpha, \beta)$ can be calculated by means of Eq. (5.5-11). In other words, a linear system H is completely characterized by its impulse response.

If H is position invariant, then, from Eq. (5.5-5),

$$H[\delta(x - \alpha, y - \beta)] = h(x - \alpha, y - \beta). \tag{5.5-12}$$

Equation (5.5-11) reduces in this case to

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta) h(x - \alpha, y - \beta)\, d\alpha\, d\beta. \tag{5.5-13}$$

This expression is called the *convolution integral*; it is the continuous-variable equivalent of the discrete convolution expression in Eq. (4.2-30). This integral tells us that knowing the impulse response of a linear system allows us to compute its response, g, to any input f. The result is simply the convolution of the impulse response and the input function.

In the presence of additive noise, the expression of the linear degradation model [Eq. (5.5-11)] becomes

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta) h(x, \alpha, y, \beta)\, d\alpha\, d\beta + \eta(x, y). \tag{5.5-14}$$

If H is position invariant, Eq. (5.5-14) becomes

$$g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha, \beta) h(x - \alpha, y - \beta)\, d\alpha\, d\beta + \eta(x, y). \tag{5.5-15}$$

The values of the noise term $\eta(x, y)$ are random, and are assumed to be independent of position. Using the familiar notation for convolution, we can write Eq. (5.5-15) as

$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y) \tag{5.5-16}$$

or, based on the convolution theorem, we can express it in the frequency domain, as

$$G(u, v) = H(u, v)F(u, v) + N(u, v). \tag{5.5-17}$$

These two expressions agree with Eqs. (5.1-1) and (5.1-2). Keep in mind that, for discrete quantities, all products are term by term. For example, term ij of $H(u, v)F(u, v)$ is the product of term ij of $H(u, v)$ and term ij of $F(u, v)$.

In summary, the preceding discussion indicates that a linear, spatially-invariant degradation system with additive noise can be modeled in the spatial domain as the convolution of the degradation (point spread) function with an image, followed by the addition of noise. Based on the convolution theorem (Sections 4.2.4 and 4.6.4), the same process can be expressed in the frequency domain as the product of the transforms of the image and degradation, followed by the addition of the transform of the noise. When working in the frequency domain, we make use of an FFT algorithm, as discussed in Section 4.6. Keep in mind also the need for function padding in the implementation of discrete Fourier transforms, as outlined in Section 4.6.3.

Many types of degradations can be approximated by linear, position-invariant processes. The advantage of this approach is that the extensive tools of linear system theory then become available for the solution of image restoration problems. Nonlinear and position-dependent techniques, although more general (and usually more accurate), introduce difficulties that often have no known solution or are very difficult to solve computationally. This chapter focuses on linear, space-invariant restoration techniques. Because degradations are modeled as being the result of convolution, and restoration seeks to find filters that apply the process in reverse, the term *image deconvolution* is used frequently to signify linear image restoration. Similarly, the filters used in the restoration process often are called *deconvolution filters.*

5.6 Estimating the Degradation Function

There are three principal ways to estimate the degradation function for use in image restoration: (1) observation, (2) experimentation, and (3) mathematical modeling. These methods are discussed in the following sections. The process of restoring an image by using a degradation function that has been estimated in some way sometimes is called *blind deconvolution,* due to the fact that the true degradation function is seldom known completely.

5.6.1 Estimation by Image Observation

Suppose that we are given a degraded image without any knowledge about the degradation function H. One way to estimate this function is to gather information from the image itself. For example, if the image is blurred, we can look at a small section of the image containing simple structures, like part of an object and the background. In order to reduce the effect of noise in our observation, we

would look for areas of strong signal content. Using sample gray levels of the object and background, we can construct an unblurred image of the same size and characteristics as the observed subimage. Let the observed subimage be denoted by $g_s(x, y)$, and let the constructed subimage (which in reality is our estimate of the original image in that area) be denoted by $\hat{f}_s(x, y)$. Then, assuming that the effect of noise is negligible because of our choice of a strong-signal area, it follows from Eq. (5.5-17) that

$$H_s(u, v) = \frac{G_s(u, v)}{\hat{F}_s(u, v)}. \tag{5.6-1}$$

From the characteristics of this function we then deduce the complete function $H(u, v)$ by making use of the fact that we are assuming position invariance. For example, suppose that a radial plot of $H_s(u, v)$ turns out to have the shape of Butterworth lowpass filter. We can use that information to construct a function $H(u, v)$ on a larger scale, but having the same shape.

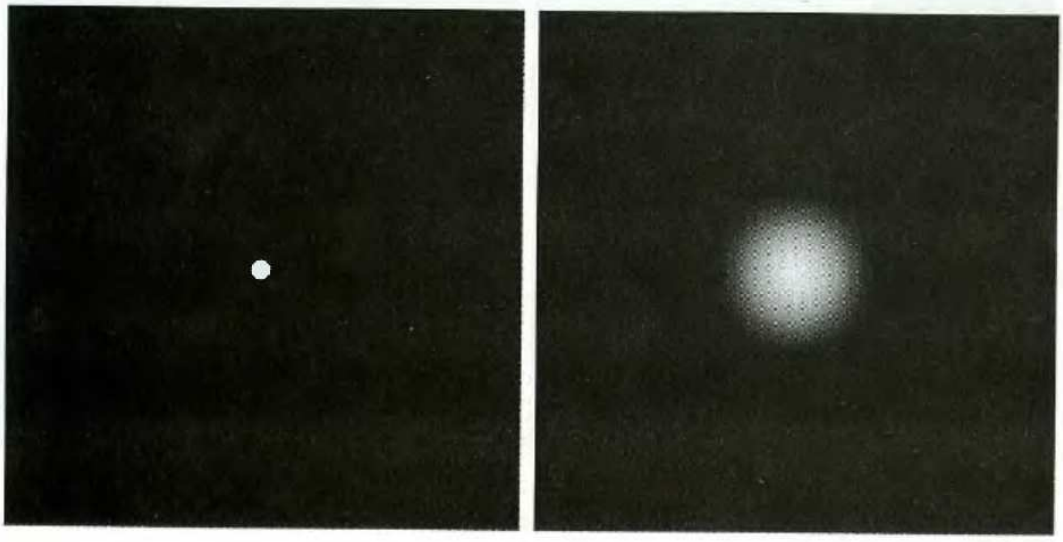
5.6.2 Estimation by Experimentation

If equipment similar to the equipment used to acquire the degraded image is available, it is possible in principle to obtain an accurate estimate of the degradation. Images similar to the degraded image can be acquired with various system settings until they are degraded as closely as possible to the image we wish to restore. Then the idea is to obtain the impulse response of the degradation by imaging an impulse (small dot of light) using the same system settings. As noted in Section 5.5, a linear, space-invariant system is described completely by its impulse response.

An impulse is simulated by a bright dot of light, as bright as possible to reduce the effect of noise. Then, recalling that the Fourier transform of an impulse is a constant, it follows from Eq. (5.5-17) that

$$H(u, v) = \frac{G(u, v)}{A} \tag{5.6-2}$$

where, as before, $G(u, v)$ is the Fourier transform of the observed image and A is a constant describing the strength of the impulse. Figure 5.24 shows an example.



a b

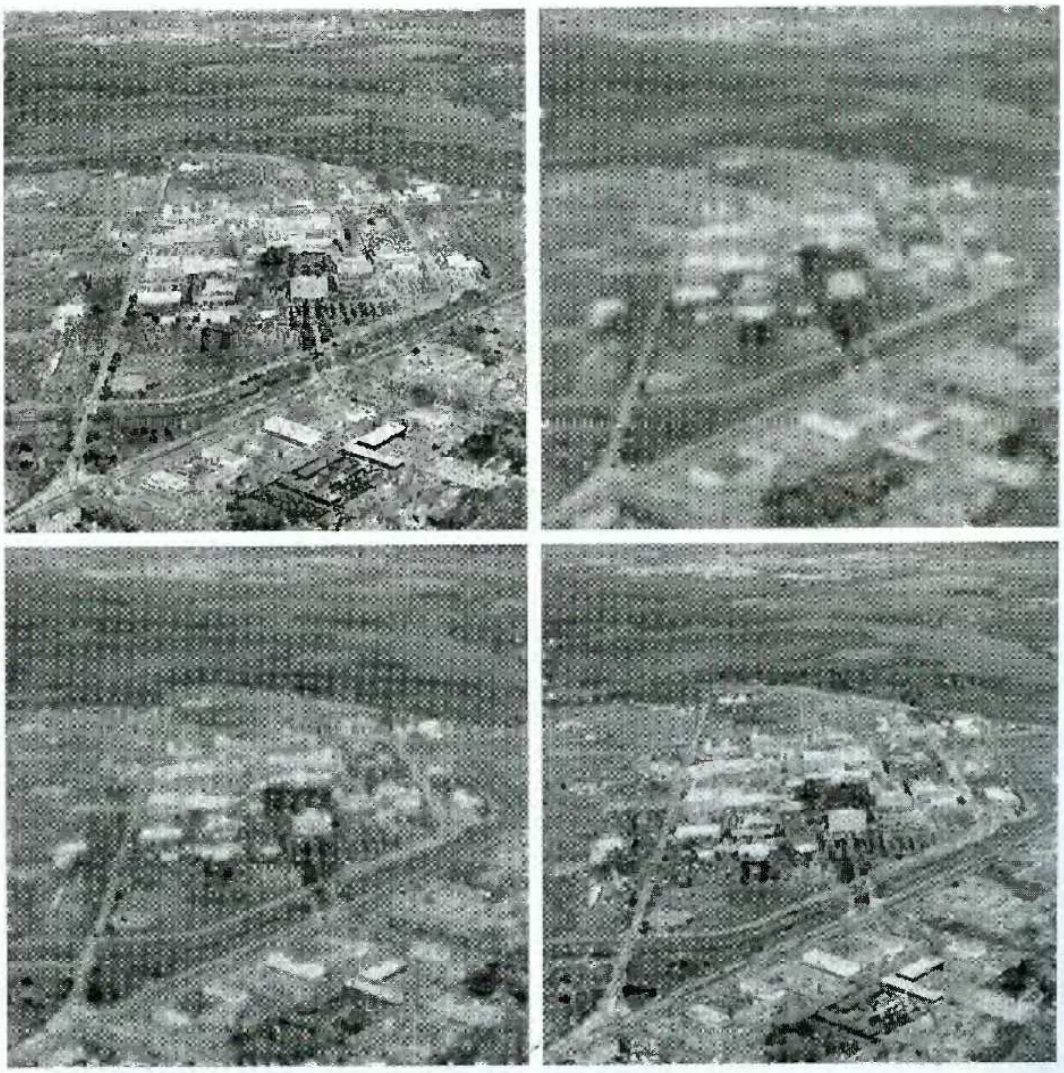
FIGURE 5.24 Degradation estimation by impulse characterization. (a) An impulse of light (shown magnified). (b) Imaged (degraded) impulse.

5.6.3 Estimation by Modeling

Degradation modeling has been used for many years because of the insight it affords into the image restoration problem. In some cases, the model can even take into account environmental conditions that cause degradations. For example, a degradation model proposed by Hufnagel and Stanley [1964] is based on the physical characteristics of atmospheric turbulence. This model has a familiar form:

$$H(u, v) = e^{-k(u^2+v^2)^{5/6}} \tag{5.6-3}$$

where k is a constant that depends on the nature of the turbulence. With the exception of the 5/6 power on the exponent, this equation has the same form as the Gaussian lowpass filter discussed in Section 4.3.3. In fact, the Gaussian LPF is used sometimes to model mild, uniform blurring. Figure 5.25 shows examples obtained by simulating blurring an image using Eq. (5.6-3) with values $k = 0.0025$ (severe turbulence in this case), $k = 0.001$ (mild turbu-



a b
c d

FIGURE 5.25 Illustration of the atmospheric turbulence model. (a) Negligible turbulence. (b) Severe turbulence, $k = 0.0025$. (c) Mild turbulence, $k = 0.001$. (d) Low turbulence, $k = 0.00025$. (Original image courtesy of NASA.)

lence), and $k = 0.00025$ (low turbulence). All images are of size 480×480 pixels.

Another major approach in modeling is to derive a mathematical model starting from basic principles. We illustrate this procedure by treating in some detail the case in which an image has been blurred by uniform linear motion between the image and the sensor during image acquisition. Suppose that an image $f(x, y)$ undergoes planar motion and that $x_0(t)$ and $y_0(t)$ are the time varying components of motion in the x- and y-directions, respectively. The total exposure at any point of the recording medium (say, film or digital memory) is obtained by integrating the instantaneous exposure over the time interval during which the imaging system shutter is open.

Assuming that shutter opening and closing takes place instantaneously, and that the optical imaging process is perfect, isolates the effect of image motion. Then, if T is the duration of the exposure, it follows that

$$g(x, y) = \int_0^T f[x - x_0(t), y - y_0(t)]\, dt \tag{5.6-4}$$

where $g(x, y)$ is the blurred image.

From Eq. (4.2-3), the Fourier transform of Eq. (5.6-4) is

$$\begin{aligned} G(u, v) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x, y) e^{-j2\pi(ux+vy)}\, dx\, dy \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[\int_0^T f[x - x_0(t), y - y_0(t)]\, dt\right] e^{-j2\pi(ux+vy)}\, dx\, dy. \end{aligned} \tag{5.6-5}$$

Reversing the order of integration allows Eq. (5.6-5) to be expressed in the form

$$G(u, v) = \int_0^T \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f[x - x_0(t), y - y_0(t)] e^{-j2\pi(ux+vy)}\, dx\, dy\right] dt. \tag{5.6-6}$$

The term inside the outer brackets is the Fourier transform of the displaced function $f[x - x_0(t), y - y_0(t)]$. Using Eq. (4.6-2) then yields the expression

$$\begin{aligned} G(u, v) &= \int_0^T F(u, v) e^{-j2\pi[ux_0(t)+vy_0(t)]}\, dt \\ &= F(u, v)\int_0^T e^{-j2\pi[ux_0(t)+vy_0(t)]}\, dt \end{aligned} \tag{5.6-7}$$

where the last step follows from the fact that $F(u, v)$ is independent of t.

By defining

$$H(u, v) = \int_0^T e^{-j2\pi[ux_0(t)+vy_0(t)]}\, dt \tag{5.6-8}$$

Eq. (5.6-7) may be expressed in the familiar form

$$G(u, v) = H(u, v)F(u, v). \tag{5.6-9}$$

If the motion variables $x_0(t)$ and $y_0(t)$ are known, the transfer function $H(u, v)$ can be obtained directly from Eq. (5.6-8). As an illustration, suppose that the image in question undergoes uniform linear motion in the x-direction only, at a rate given by $x_0(t) = at/T$. When $t = T$, the image has been displaced by a total distance a. With $y_0(t) = 0$, Eq. (5.6-8) yields

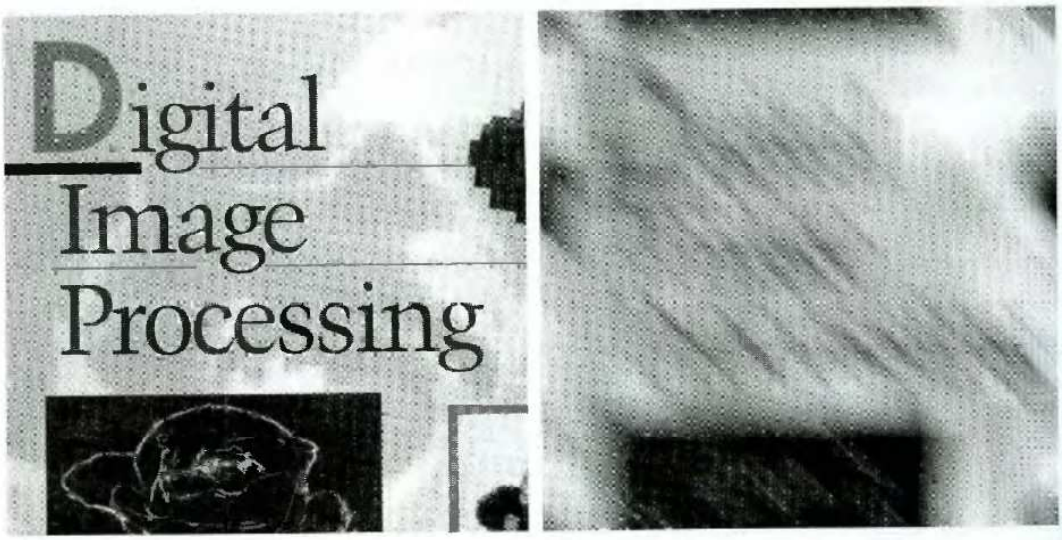
$$\begin{aligned} H(u, v) &= \int_0^T e^{-j2\pi u x_0(t)}\, dt \\ &= \int_0^T e^{-j2\pi u a t/T}\, dt \\ &= \frac{T}{\pi u a} \sin(\pi u a) e^{-j\pi u a}. \end{aligned} \tag{5.6-10}$$

It is noted that H vanishes at values of u given by $u = n/a$, where n is an integer. If we allow the y-component to vary as well, with the motion given by $y_0 = bt/T$, then the degradation function becomes

$$H(u, v) = \frac{T}{\pi(ua + vb)} \sin\left[\pi(ua + vb)\right] e^{-j\pi(ua+vb)}. \tag{5.6-11}$$

EXAMPLE 5.10: Image blurring due to motion.

■ The blurring characteristics of Eq. (5.6-11) are illustrated next. Figure 5.26(b) is an image blurred by computing the Fourier transform of the image in Fig. 5.26(a), multiplying the transform by $H(u, v)$ from Eq. (5.6-11), and taking the inverse transform. The images are of size 688×688 pixels, and the parameters used in Eq. (5.6-11) were $a = b = 0.1$ and $T = 1$. As discussed in Sections 5.8 and 5.9, recovery of the original image from its blurred counterpart presents some interesting challenges, particularly when noise is present in the degraded image. ■



a b

FIGURE 5.26 (a) Original image. (b) Result of blurring using the function in Eq. (5.6-11) with $a = b = 0.1$ and $T = 1$.

5.7 Inverse Filtering

The material in this section is our first step in studying restoration of images degraded by a degradation function H, which is given or obtained by a method such as those discussed in the previous section. The simplest approach to restoration is direct inverse filtering, where we compute an estimate, $\hat{F}(u, v)$, of the transform of the original image simply by dividing the transform of the degraded image, $G(u, v)$, by the degradation function:

$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)}. \tag{5.7-1}$$

The divisions are between individual elements of the functions, as explained in connection with Eq. (5.5-17). Substituting the right side of Eq. (5.5-17) for $G(u, v)$ in Eq. (5.7-1) yields

$$\hat{F}(u, v) = F(u, v) + \frac{N(u, v)}{H(u, v)}. \tag{5.7-2}$$

This is an interesting expression. It tells us that even if we know the degradation function we cannot recover the undegraded image [the inverse Fourier transform of $F(u, v)$] exactly because $N(u, v)$ is a random function whose Fourier transform is not known. There is more bad news. If the degradation has zero or very small values, then the ratio $N(u, v)/H(u, v)$ could easily dominate the estimate $\hat{F}(u, v)$. This, in fact, is frequently the case, as will be demonstrated shortly.

One approach to get around the zero or small-value problem is to limit the filter frequencies to values near the origin. From Eq. (4.2-22) we know that $H(0, 0)$ is equal to the average value of $h(x, y)$ and that this is usually the highest value of $H(u, v)$ in the frequency domain. Thus, by limiting the analysis to frequencies near the origin, we reduce the probability of encountering zero values. This approach is illustrated in the following example.

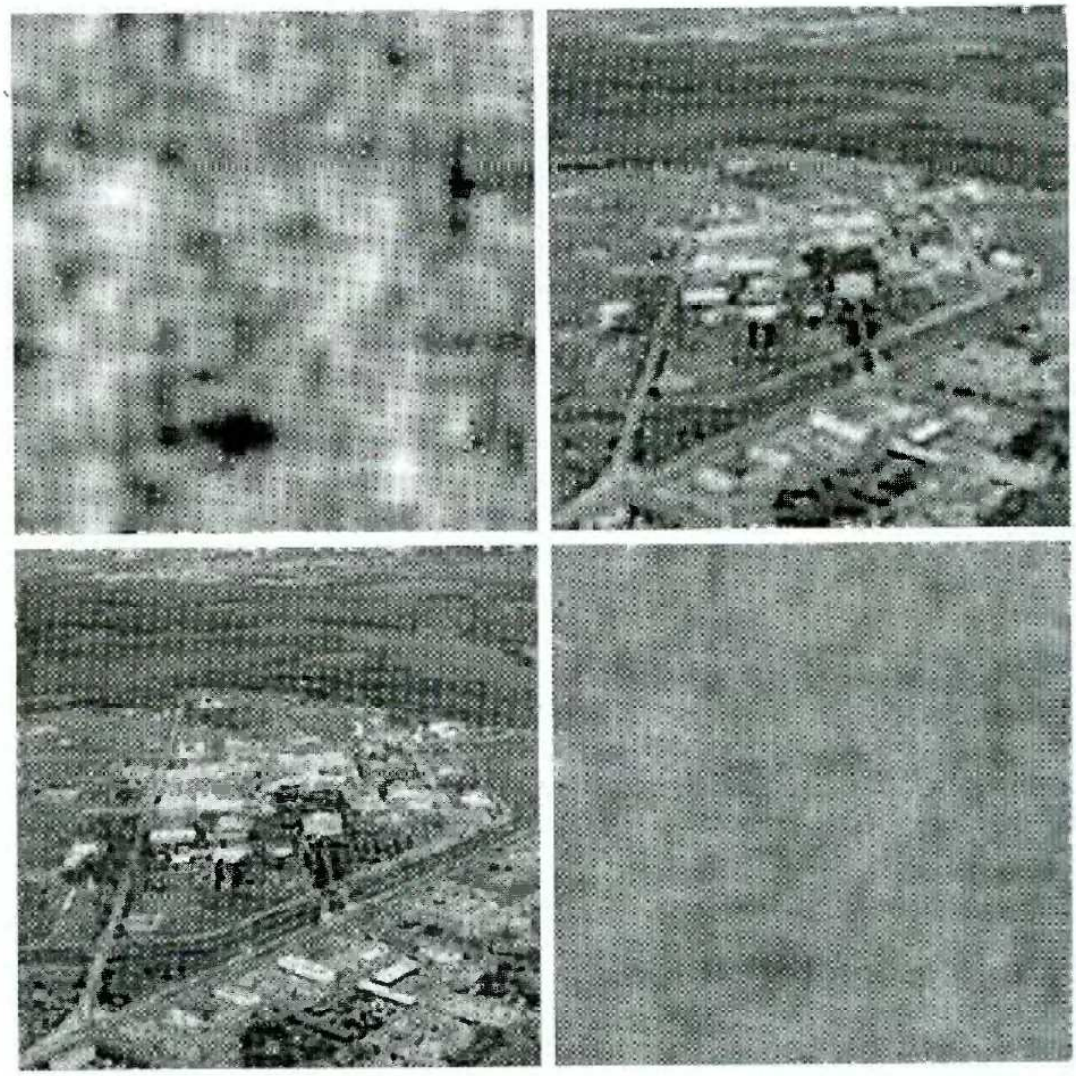
EXAMPLE 5.11: Inverse filtering.

■ The image shown in Fig. 5.25(b) was inverse filtered with Eq. (5.7-1) using the exact inverse of the degradation function that generated that image. That is, the degradation function used was

$$H(u, v) = e^{-k[(u-M/2)^2+(v-N/2)^2]^{5/6}}$$

with $k = 0.0025$. The $M/2$ and $N/2$ constants are offset values; they center the function so that it will correspond with the centered Fourier transform, as discussed on numerous occasions in the previous chapter. In this case, $M = N = 480$. We know that a Gaussian-shape function has no zeros, so that will not be a concern here. However, in spite of this, the degradation values became so small that the result of full inverse filtering [Fig. 5.27(a)] is useless. The reasons for this poor result are as discussed in connection with Eq. (5.7-2).

Figures 5.27(b) through (d) show the results of cutting off values of the ratio $G(u, v)/H(u, v)$ outside a radius of 40, 70, and 85, respectively. The cut off was implemented by applying to the ratio a Butterworth lowpass function of order 10. This provided a sharp (but smooth) transition at the desired radius. Radii near 70 yielded the best visual results [Fig. 5.27(c)]. Radius values below that



a b
c d

FIGURE 5.27 Restoring Fig. 5.25(b) with Eq. (5.7-1). (a) Result of using the full filter. (b) Result with H cut off outside a radius of 40; (c) outside a radius of 70; and (d) outside a radius of 85.

tended toward blurred images, as illustrated in Fig. 5.27(b), which was obtained using a radius of 40. Values above 70 started to produce degraded images, as illustrated in Fig. 5.27(d), which was obtained using a radius of 85. The image content is almost visible behind a "curtain" of noise, but the noise definitely dominates the result. Further increases in radius values produced images that looked more and more like Fig. 5.27(a). ■

The results in the preceding example are illustrative of the poor performance of direct inverse filtering in general. The basic theme of the sections that follow is how to improve on direct inverse filtering.

5.8 Minimum Mean Square Error (Wiener) Filtering

The inverse filtering approach discussed in the previous section makes no explicit provision for handling noise. In this section we discuss an approach that incorporates both the degradation function and statistical characteristics of noise into the restoration process. The method is founded on considering images and noise as random processes, and the objective is to find an estimate $\hat{f}$ of the

uncorrupted image f such that the mean square error between them is minimized. This error measure is given by

$$e^2 = E\{(f - \hat{f})^2\} \tag{5.8-1}$$

where $E\{\cdot\}$ is the expected value of the argument. It is assumed that the noise and the image are uncorrelated; that one or the other has zero mean; and that the gray levels in the estimate are a linear function of the levels in the degraded image. Based on these conditions, the minimum of the error function in Eq. (5.8-1) is given in the frequency domain by the expression

$$\begin{aligned}\hat{F}(u, v) &= \left[\frac{H^*(u, v)S_f(u, v)}{S_f(u, v)|H(u, v)|^2 + S_\eta(u, v)}\right]G(u, v) \\ &= \left[\frac{H^*(u, v)}{|H(u, v)|^2 + S_\eta(u, v)/S_f(u, v)}\right]G(u, v) \\ &= \left[\frac{1}{H(u, v)}\frac{|H(u, v)|^2}{|H(u, v)|^2 + S_\eta(u, v)/S_f(u, v)}\right]G(u, v)\end{aligned} \tag{5.8-2}$$

where we used the fact that the product of a complex quantity with its conjugate is equal to the magnitude of the complex quantity squared. This result is known as the *Wiener filter*, after N. Wiener [1942], who first proposed the concept in the year shown. The filter, which consists of the terms inside the brackets, also is commonly referred to as the *minimum mean square error filter* or the *least square error filter*. We include references at the end of the chapter to sources containing detailed derivations of the Wiener filter. Note from the first line in Eq. (5.8-2) that the Wiener filter does not have the same problem as the inverse filter with zeros in the degradation function, unless both $H(u, v)$ and $S_\eta(u, v)$ are zero for the same value(s) of u and v.

The terms in Eq. (5.8-2) are as follows:

$H(u, v)$ = degradation function
$H^*(u, v)$ = complex conjugate of $H(u, v)$
$|H(u, v)|^2 = H^*(u, v)H(u, v)$
$S_\eta(u, v) = |N(u, v)|^2$ = power spectrum of the noise [see Eq. (4.2-20)]
$S_f(u, v) = |F(u, v)|^2$ = power spectrum of the undegraded image.

As before, $H(u, v)$ is the transform of the degradation function and $G(u, v)$ is the transform of the degraded image. The restored image in the spatial domain is given by the inverse Fourier transform of the frequency-domain estimate $\hat{F}(u, v)$. Note that if the noise is zero, then the noise power spectrum vanishes and the Wiener filter reduces to the inverse filter.

When we are dealing with spectrally white noise, the spectrum $|N(u, v)|^2$ is a constant, which simplifies things considerably. However, the power spectrum of the undegraded image seldom is known. An approach used frequently when these quantities are not known or cannot be estimated is to approximate Eq. (5.8-2) by the expression

$$\hat{F}(u, v) = \left[\frac{1}{H(u, v)}\frac{|H(u, v)|^2}{|H(u, v)|^2 + K}\right]G(u, v) \tag{5.8-3}$$

a b c

FIGURE 5.28 Comparison of inverse- and Wiener filtering. (a) Result of full inverse filtering of Fig. 5.25(b). (b) Radially limited inverse filter result. (c) Wiener filter result.

where K is a specified constant. The following examples illustrate the use of this expression.

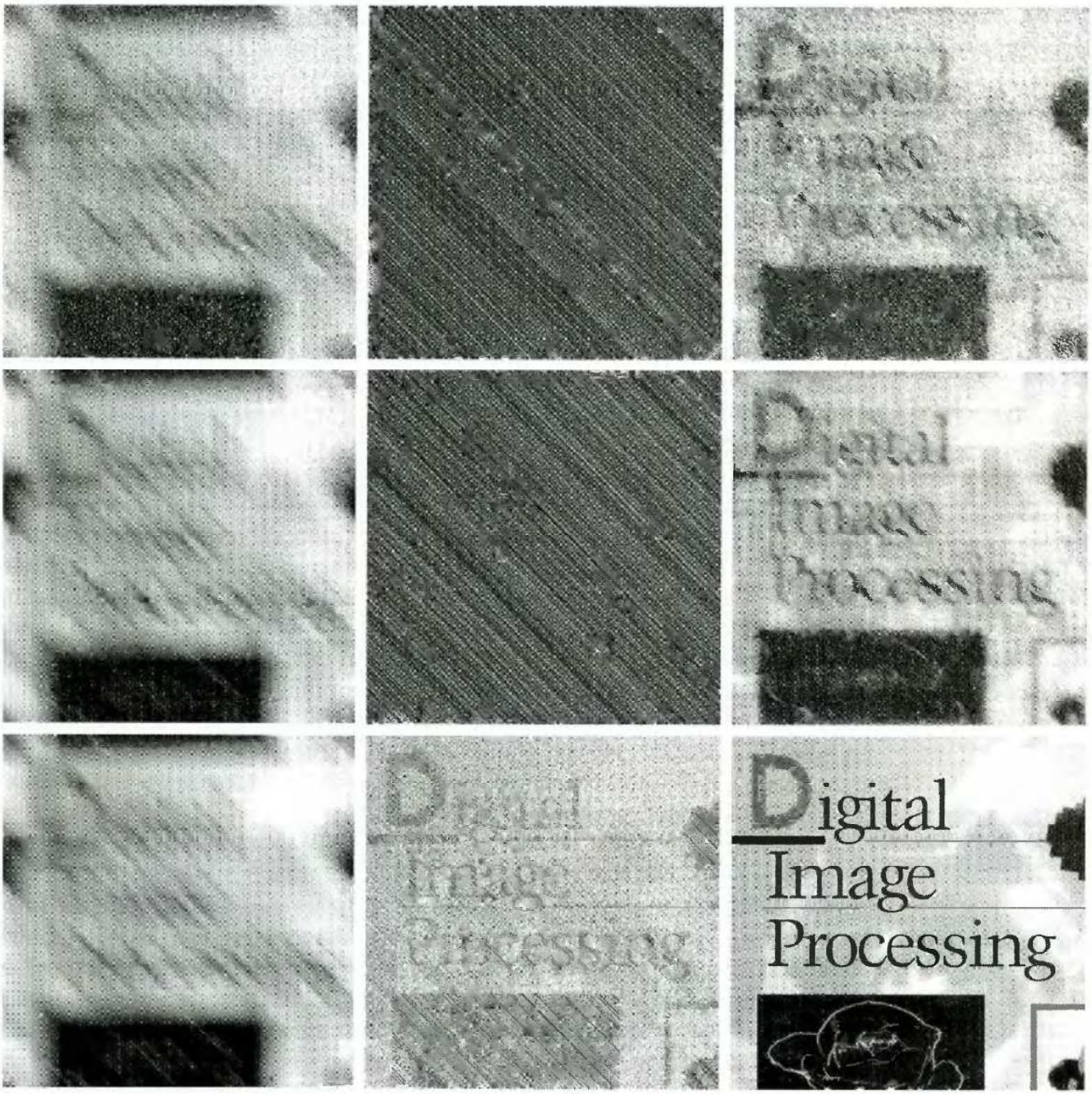
EXAMPLE 5.12: Comparison of inverse and Wiener filtering.

■ Figure 5.28 illustrates the power of Wiener filtering over direct inverse filtering. Figure 5.28(a) is the full inverse-filtered result shown in Fig. 5.27(a). Similarly, Fig. 5.28(b) is the radially limited inverse filter result of Fig. 5.27(c). These images are duplicated here for convenience in making comparisons. Figure 5.28(c) shows the result obtained using Eq. (5.8-3) with the degradation function used in Example 5.11. The value of K was chosen interactively to yield the best visual results. The power of Wiener filtering over the direct inverse approach is evident in this example. By comparing Figs. 5.25(a) and 5.28(c), we see that the Wiener filter yielded a result very close in appearance to the original image. ■

EXAMPLE 5.13: Further comparisons of Wiener filtering.

■ The first row of Fig. 5.29 shows, from left to right, the blurred image of Fig. 5.26(b) heavily corrupted by additive Gaussian noise of zero mean and variance of 650; the result of direct inverse filtering; and the result of Wiener filtering. The Wiener filter of Eq. (5.8-3) was used, with $H(u, v)$ from Example 5.10, and with K chosen interactively to give the best possible visual result. As expected, the inverse filter produced an unusable image. Note that the noise in the inverse filter image is so strong that its structure is in the direction of the *deblurring* filter. The Wiener filter result is by no means perfect, but it does give us a hint as to image content. With some difficulty, the text is readable.

The second row of Fig. 5.29 shows the same sequence, but with the level of noise variance reduced by one order of magnitude. This reduction had little effect on the inverse filter, but the Wiener results are considerably improved. The text now is much easier to read. In the third row of Fig. 5.29, the noise variance has decreased more than five orders of magnitude from the first row. In fact, image 5.29(g) has no visible noise. The inverse filter result is interesting in this case. The noise is still quite visible, but the text can be seen through a "curtain" of noise. This is a good example of the comments made regarding Eq. (5.7-2). In other words, as is evident in Fig. 5.29(h), the inverse filter was



a b c
d e f
g h i

FIGURE 5.29 (a) Image corrupted by motion blur and additive noise. (b) Result of inverse filtering. (c) Result of Wiener filtering. (d)–(f) Same sequence, but with noise variance one order of magnitude less. (g)–(i) Same sequence, but noise variance reduced by five orders of magnitude from (a). Note in (h) how the deblurred image is quite visible through a "curtain" of noise.

quite capable of essentially eliminating the blur in the image. However, the noise still dominates the result. If we could "look" behind the noise in Figs. 5.29(b) and (e), the characters also would show with little blurring. The Wiener filter result in Fig. 5.29(i) is excellent, being quite close visually to the original image shown in Fig. 5.26(a). These types of results are representative of what is possible with Wiener filtering, as long as a reasonable estimate of the degradation function is available. ■

5.9 Constrained Least Squares Filtering

The problem of having to know something about the degradation function H is common to all methods discussed in this chapter. However, the Wiener filter presents an additional difficulty: The power spectra of the undegraded image and noise must be known. We showed in the previous section that it is possible to achieve excellent results using the approximation given in Eq. (5.8-3). However, a constant estimate of the ratio of the power spectra is not always a suitable solution.

The method discussed in this section requires knowledge of only the mean and variance of the noise. As discussed in Section 5.2.4, these parameters usually can be calculated from a given degraded image, so this is an important advantage. Another difference is that the Wiener filter is based on minimizing a statistical criterion and, as such, it is optimal in an average sense. The algorithm presented in this section has the notable feature that it yields an optimal result for *each* image to which it is applied. Of course, it is important to keep in mind that these optimality criteria, while satisfying from a theoretical point of view, are not related to the dynamics of visual perception. As a result, the choice of one algorithm over the other will almost always be determined (at least partially) by the perceived visual quality of the resulting images.

By using the definition of convolution given in Eq. (4.2-30), we can express Eq. (5.5-16) in vector-matrix form, as follows:

$$\mathbf{g} = \mathbf{Hf} + \boldsymbol{\eta}. \tag{5.9-1}$$

See inside front cover
Consult the book web site for a brief review of vectors and matrices.

For example, suppose that $g(x, y)$ is of size $M \times N$. Then we can form the first N elements of the vector $\mathbf{g}$ by using the image elements in first row of $g(x, y)$, the next N elements from the second row, and so on. The resulting vector will have dimensions $MN \times 1$. These are also the dimensions of $\mathbf{f}$ and $\boldsymbol{\eta}$, as these vectors are formed in the same manner. The matrix $\mathbf{H}$ then has dimensions $MN \times MN$. Its elements are given by the elements of the convolution given in Eq. (4.2-30).

It would be reasonable to come to the conclusion that the restoration problem can now be reduced to simple matrix manipulations. Unfortunately, this is not the case. For instance, suppose that we are working with images of medium size; say $M = N = 512$. Then the vectors in Eq. (5.9-1) would be of dimension $262{,}144 \times 1$, and matrix $\mathbf{H}$ would be of dimensions $262{,}144 \times 262{,}144$. Manipulating vectors and matrices of these sizes is not a trivial task. The problem is complicated further by the fact $\mathbf{H}$ is highly sensitive to noise (after the experiences we had with the effect of noise in the previous two sections, this should not be a surprise). However, formulating the restoration problem in matrix form does facilitate derivation of restoration techniques.

Although we do not fully derive the method of constrained least squares that we are about to present, this method has its roots in a matrix formulation. We give references at the end of the chapter to sources where such derivations are covered in detail. Central to the method is the issue of the sensitivity of **H** to noise. One way to alleviate the noise sensitivity problem is to base optimality of restoration on a measure of smoothness, such as the second derivative of an image (our old friend the Laplacian). To be meaningful, the restoration must be constrained by the parameters of the problems at hand. Thus, what is desired is to find the minimum of a criterion function, C, defined as

$$C = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \left[\nabla^2 f(x, y) \right]^2 \tag{5.9-2}$$

subject to the constraint

$$\|\mathbf{g} - \mathbf{H}\hat{\mathbf{f}}\|^2 = \|\boldsymbol{\eta}\|^2 \tag{5.9-3}$$

where $\|\mathbf{w}\|^2 \triangleq \mathbf{w}^T\mathbf{w}$ is the Euclidean vector norm,† and $\hat{\mathbf{f}}$ is the estimate of the undegraded image. The Laplacian operator ∇^2 is defined in Eq. (3.7-1).

The frequency domain solution to this optimization problem is given by the expression

$$\hat{F}(u, v) = \left[\frac{H^*(u, v)}{|H(u, v)|^2 + \gamma |P(u, v)|^2} \right] G(u, v) \tag{5.9-4}$$

where γ is a parameter that must be adjusted so that the constraint in Eq. (5.9-3) is satisfied, and $P(u, v)$ is the Fourier transform of the function

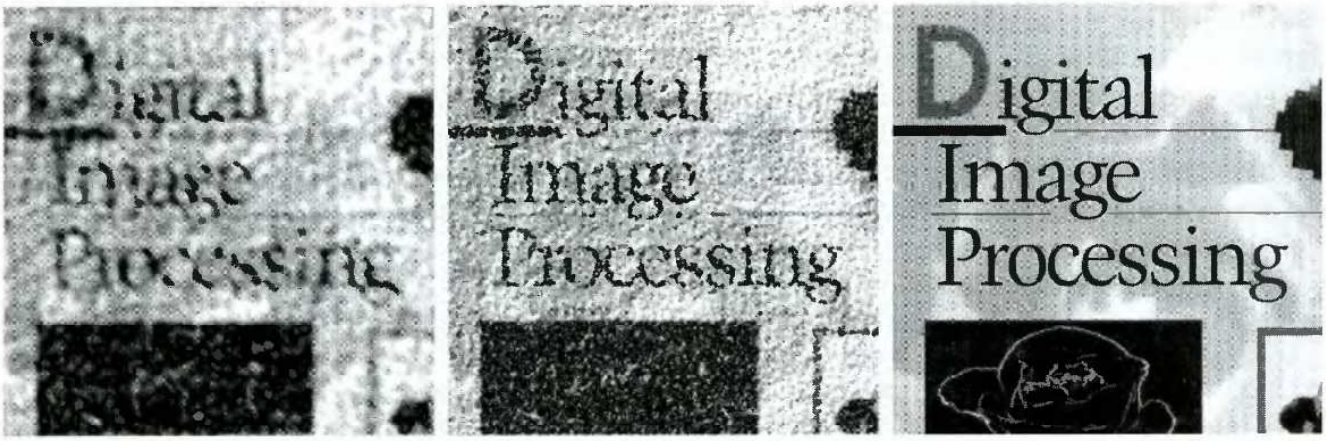
$$p(x, y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}. \tag{5.9-5}$$

We recognize this function as the Laplacian operator introduced in Section 3.7.2. As noted earlier, it is important to keep in mind that $p(x, y)$, as well as all other relevant spatial domain functions, must be properly padded with zeros prior to computing their Fourier transforms for use in Eq. (5.9-4), as discussed in Section 4.6.3. Note that Eq. (5.9-4) reduces to inverse filtering if γ is zero.

EXAMPLE 5.14: Comparison of Wiener and constrained least squares filtering.

■ Figure 5.30 shows the result of processing Figs. 5.29(a), (d), and (g) with constrained least squares filters, in which the values of γ were selected manually to yield the best visual results. This is the same procedure we used to generate the Wiener filtered results in Fig. 5.29(c), (f), and (i). By comparing the constrained least squares and Wiener results, it is noted that the former yielded slightly better results for the high- and medium-noise cases, with both filters generating essentially equal results for the low-noise case. It is not unexpected that the constrained least squares filter would outperform the Wiener filter when selecting the parameters manually for better visual results. The parameter γ in Eq. (5.9-4) is a scalar, while the value of K in Eq. (5.8-3) is an approximation to

†Recall that, for a vector $\mathbf{w}$ with n components, $\mathbf{w}^T\mathbf{w} = \sum_{k=1}^{n} w_k^2$, where w_k is the kth component of $\mathbf{w}$.



a b c

FIGURE 5.30 Results of constrained least squares filtering. Compare (a), (b), and (c) with the Wiener filtering results in Figs. 5.29(c), (f), and (i), respectively.

the ratio of two unknown frequency domain functions, whose ratio seldom is constant. Thus, it stands to reason that a result based on manually selecting γ would be a more accurate estimate of the undegraded image. ■

As shown in the preceding example, it is possible to adjust the parameter γ interactively until acceptable results are achieved. If we are interested in optimality, however, then the parameter γ must be adjusted so that the constraint in Eq. (5.9-3) is satisfied. A procedure for computing γ by iteration is as follows.

Define a "residual" vector $\mathbf{r}$ as

$$\mathbf{r} = \mathbf{g} - \mathbf{H}\hat{\mathbf{f}}. \tag{5.9-6}$$

Since, from the solution in Eq. (5.9-4), $\hat{F}(u, v)$ (and by implication $\hat{\mathbf{f}}$) is a function of γ, then $\mathbf{r}$ also is a function of this parameter. It can be shown (Hunt [1973]) that

$$\begin{aligned} \phi(\gamma) &= \mathbf{r}^T\mathbf{r} \\ &= \|\mathbf{r}\|^2 \end{aligned} \tag{5.9-7}$$

is a monotonically increasing function of γ. What we want to do is adjust gamma so that

$$\|\mathbf{r}\|^2 = \|\boldsymbol{\eta}\|^2 \pm a \tag{5.9-8}$$

where a is an accuracy factor. In view of Eq. (5.9-6), if $\|\mathbf{r}\|^2 = \|\boldsymbol{\eta}\|^2$, the constraint in Eq. (5.9-3) will be strictly satisfied.

Because $\phi(\gamma)$ is monotonic, finding the desired value of γ is not difficult. One approach is to

1. Specify an initial value of γ.
2. Compute $\|\mathbf{r}\|^2$.
3. Stop if Eq. (5.9-8) is satisfied; otherwise return to Step 2 after increasing γ if $\|\mathbf{r}\|^2 < \|\boldsymbol{\eta}\|^2 - a$ or decreasing γ if $\|\mathbf{r}\|^2 > \|\boldsymbol{\eta}\|^2 + a$. Use the new value of γ in Eq. (5.9-4) to recompute the optimum estimate $\hat{F}(u, v)$.

Other procedures, such as a Newton-Raphson algorithm, can be used to improve the speed of convergence.

In order to use this algorithm, we need the quantities $\|\mathbf{r}\|^2$ and $\|\boldsymbol{\eta}\|^2$. To compute $\|\mathbf{r}\|^2$, we note from Eq. (5.9-6) that

$$R(u, v) = G(u, v) - H(u, v)\hat{F}(u, v) \tag{5.9-9}$$

from which we obtain $r(x, y)$ by computing the inverse transform of $R(u, v)$. Then

$$\|\mathbf{r}\|^2 = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} r^2(x, y). \tag{5.9-10}$$

Computation of $\|\boldsymbol{\eta}\|^2$ leads to an interesting result. First, consider the variance of the noise over the entire image, which we estimate by the sample-average method:

$$\sigma_\eta^2 = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\eta(x, y) - m_\eta]^2 \tag{5.9-11}$$

where

$$m_\eta = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \eta(x, y) \tag{5.9-12}$$

is the sample mean. With reference to the *form* of Eq. (5.9-10), we note that the double summation in Eq. (5.9-11) is equal to $\|\boldsymbol{\eta}\|^2$. This gives us the expression

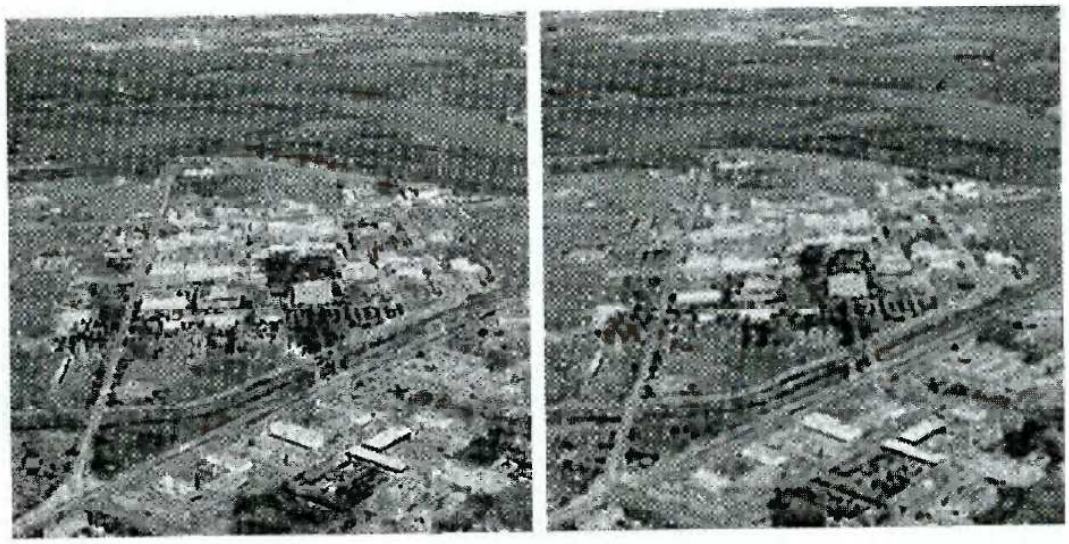
$$\|\boldsymbol{\eta}\|^2 = MN[\sigma_\eta^2 + m_\eta^2]. \tag{5.9-13}$$

This is a most useful result. It tells us that we can implement an optimum restoration algorithm by having knowledge of only the mean and variance of the noise. These quantities are not difficult to estimate (Section 5.2.4), assuming that the noise and image gray-level values are not correlated. This is a basic assumption of all the methods discussed in this chapter.

EXAMPLE 5.15: Iterative estimation of the optimum constrained least squares filter.

■ Figure 5.31(a) shows the result obtained by using the algorithm just described to estimate the optimum filter for restoring Fig. 5.25(b). The initial value used for γ was 10^{-5}, the correction factor for adjusting γ was 10^{-6}, and the value for a was 0.25. The noise parameters specified were the same used to generate Fig. 5.25(a): a noise variance of 10^{-5}, and zero mean. The restored result is almost as good as Fig. 5.28(c), which was obtained by Wiener filtering with K manually specified for best visual results. Figure 5.31(b) shows what can happen if the wrong estimate of noise parameters are used. In this case, the noise variance specified was 10^{-2} and the mean was left at a value of 0. The result in this case is considerably more blurred. ■

As stated a the beginning of this section, it is important to keep in mind that optimum restoration in the sense of constrained least squares does not necessarily imply "best" in the visual sense. Depending on the nature and magnitude of the degradation and noise, the other parameters in the algorithm for iteratively determining the optimum estimate also play a role in the final result. In general, automatically determined restoration filters yield inferior results to manual adjustment of filter parameters. This is particularly true of the constrained least squares filter, which is completely specified by a single, scalar parameter.



a b

FIGURE 5.31 (a) Iteratively determined constrained least squares restoration of Fig. 5.16(b), using correct noise parameters. (b) Result obtained with wrong noise parameters.

5.10 Geometric Mean Filter

It is possible to generalize slightly the Wiener filter discussed in the Section 5.8. The generalization is in the form of the so-called *geometric mean filter*:

$$\hat{F}(u,v) = \left[\frac{H^*(u,v)}{|H(u,v)|^2}\right]^{\alpha}\left[\frac{H^*(u,v)}{|H(u,v)|^2 + \beta\left[\dfrac{S_\eta(u,v)}{S_f(u,v)}\right]}\right]^{1-\alpha} G(u,v) \qquad (5.10\text{-}1)$$

with α and β being positive, real constants. The geometric mean filter consists of the two expressions in brackets raised to the powers α and $1 - \alpha$, respectively.

When $\alpha = 1$ this filter reduces to the inverse filter. With $\alpha = 0$ the filter becomes the so-called *parametric Wiener filter*, which reduces to the standard Wiener filter when $\beta = 1$. If $\alpha = 1/2$, the filter becomes a product of the two quantities raised to the same power, which is the definition of the geometric mean, thus giving the filter its name. With $\beta = 1$, as α decreases below $1/2$, the filter performance will tend more toward the inverse filter. Similarly, when α increases above $1/2$, the filter will behave more like the Wiener filter. When $\alpha = 1/2$ and $\beta = 1$, the filter also is commonly referred to as the *spectrum equalization filter*. Equation (5.10-1) is quite useful when implementing restoration filters because it really represents a family of filters combined into a single expression.

5.11 Geometric Transformations

We conclude this chapter with an introductory discussion on the use of geometric transformations for image restoration. Unlike the techniques discussed so far, geometric transformations modify the spatial relationships between pixels in an image. Geometric transformations often are called *rubber-sheet transformations*, because they may be viewed as the process of "printing" an

image on a sheet of rubber and then stretching this sheet according to some predefined set of rules.

In terms of digital image processing, a geometric transformation consists of two basic operations: (1) a *spatial transformation*, which defines the "rearrangement" of pixels on the image plane; and (2) *gray-level interpolation*, which deals with the assignment of gray levels to pixels in the spatially transformed image. We discuss in the following sections the fundamental ideas underlying these concepts, and their use in the context of image restoration.

5.11.1 Spatial Transformations

Suppose that an image f with pixel coordinates (x, y) undergoes geometric distortion to produce an image g with coordinates (x', y'). This transformation may be expressed as

$$x' = r(x, y) \tag{5.11-1}$$

and

$$y' = s(x, y) \tag{5.11-2}$$

where $r(x, y)$ and $s(x, y)$ are the spatial transformations that produced the geometrically distorted image $g(x', y')$. For example, if $r(x, y) = x/2$ and $s(x, y) = y/2$, the "distortion" is simply a shrinking of the size of $f(x, y)$ by one-half in both spatial directions.

If $r(x, y)$ and $s(x, y)$ were known analytically, recovering $f(x, y)$ from the distorted image $g(x', y')$ by applying the transformations in reverse might be possible theoretically. In practice, however, formulating a single set of analytical functions $r(x, y)$ and $s(x, y)$ that describe the geometric distortion process over the entire image plane generally is not possible. The method used most frequently to overcome this difficulty is to formulate the spatial relocation of pixels by the use of *tiepoints*, which are a subset of pixels whose location in the input (distorted) and output (corrected) images is known precisely.

Figure 5.32 shows quadrilateral regions in a distorted and corresponding corrected image. The vertices of the quadrilaterals are corresponding tiepoints.

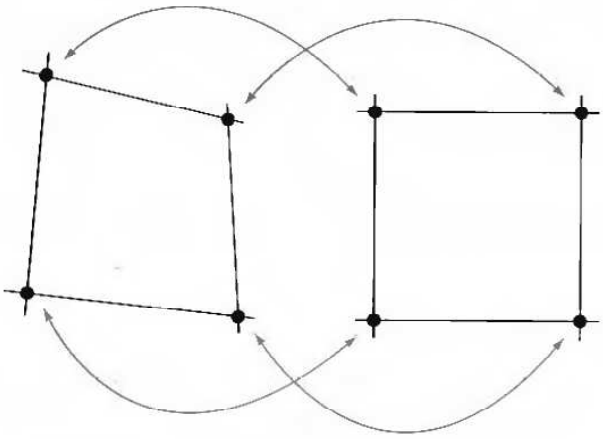


FIGURE 5.32 Corresponding tiepoints in two image segments.

Suppose that the geometric distortion process within the quadrilateral regions is modeled by a pair of bilinear equations so that

$$r(x, y) = c_1 x + c_2 y + c_3 xy + c_4 \tag{5.11-3}$$

and

$$s(x, y) = c_5 x + c_6 y + c_7 xy + c_8. \tag{5.11-4}$$

Then, from Eqs. (5.11-1) and (5.11-2),

$$x' = c_1 x + c_2 y + c_3 xy + c_4 \tag{5.11-5}$$

and

$$y' = c_5 x + c_6 y + c_7 xy + c_8. \tag{5.11-6}$$

Since there are a total of eight known tiepoints, these equations can be solved for the eight coefficients c_i, $i = 1, 2, \ldots, 8$. The coefficients constitute the geometric distortion model used to transform *all* pixels within the quadrilateral region defined by the tiepoints used to obtain the coefficients. In general, enough tiepoints are needed to generate a set of quadrilaterals that cover the entire image, with each quadrilateral having its own set of coefficients.

Once we have the coefficients, the procedure used to generate the corrected (i.e., restored) image is not difficult. If we want to find the value of the undistorted image at any point (x_0, y_0), we simply need to know where in the distorted image $f(x_0, y_0)$ was mapped. This we find out by substituting (x_0, y_0) into Eqs. (5.11-5) and (5.11-6) to obtain the geometrically distorted coordinates (x_0', y_0'). The value of the point in the undistorted image that was mapped to (x_0', y_0') is $g(x_0', y_0')$. So we obtain the restored image value simply by letting $\hat{f}(x_0, y_0) = g(x_0', y_0')$. For example, to generate $\hat{f}(0, 0)$, we substitute $(x, y) = (0, 0)$ into Eqs. (5.11-5) and (5.11-6) to obtain a pair of coordinates (x', y') from those equations. Then we let $\hat{f}(0, 0) = g(x', y')$, where x' and y' are the coordinate values just obtained. Next, we substitute $(x, y) = (0, 1)$ into Eqs. (5.11-5) and (5.11-6), obtain another pair of values (x', y'), and let $f(0, 1) = g(x', y')$ for those coordinate values. The procedure continues pixel by pixel and row by row until an array whose size does not exceed the size of image g is obtained. A column (rather than a row) scan would yield identical results. Also, a bookkeeping procedure is needed to keep track of which quadrilaterals apply at a given pixel location in order to use the proper coefficients.

Tiepoints are established by a number of different techniques, depending on the application. For instance, some image generation systems having physical artifacts (such as metallic points) embedded on the imaging sensor itself. These produce a *known* set of points (called *reseau marks*) directly on the image as it is acquired. If the image is distorted later by some other process (such as an image display or image reconstruction process), then the image can be geometrically corrected using the technique just described.

5.11.2 Gray-Level Interpolation

The method discussed in the preceding section steps through integer values of the coordinates (x, y) to yield the restored image $\hat{f}(x, y)$. However, depending on the values of the coefficients c_i, Eqs. (5.11-5) and (5.11-6) can yield noninteger val-

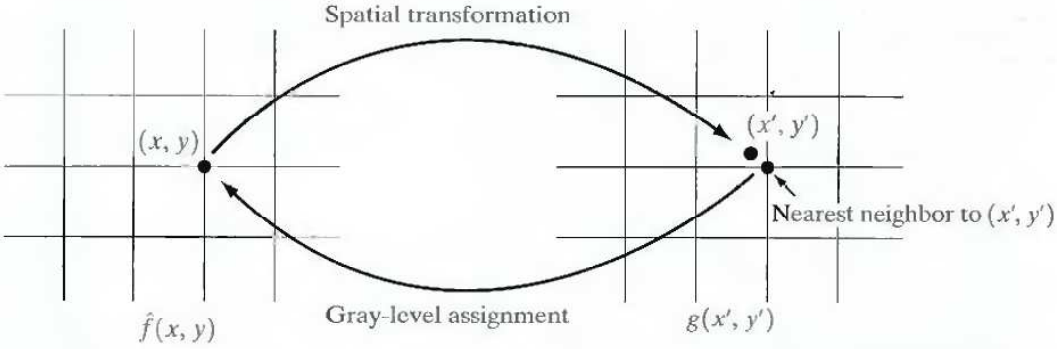


FIGURE 5.33 Gray-level interpolation based on the nearest neighbor concept.

ues for x' and y'. Because the distorted image g is digital, its pixel values are defined only at integer coordinates. Thus using noninteger values for x' and y' causes a mapping into locations of g for which no gray levels are defined. Inferring what the gray-level values at those locations should be, based only on the pixel values at integer coordinate locations, then becomes necessary. The technique used to accomplish this is called *gray-level interpolation*.

The simplest scheme for gray-level interpolation is based on a nearest neighbor approach. This method, also called *zero-order interpolation*, is illustrated in Fig. 5.33. This figure shows (1) the mapping of integer (x, y) coordinates into fractional coordinates (x', y') by means of Eqs. (5.11-5) and (5.11-6); (2) the selection of the closest integer coordinate neighbor to (x', y'); and (3) the assignment of the gray level of this nearest neighbor to the pixel located at (x, y).

Although nearest neighbor interpolation is simple to implement, this method often has the drawback of producing undesirable artifacts, such as distortion of straight edges in images of high resolution. Smoother results can be obtained by using more sophisticated techniques, such as *cubic convolution interpolation*, which fits a surface of the $\sin(z)/z$ type through a much larger number of neighbors (say, 16) in order to obtain a smooth estimate of the gray level at any desired point. Typical areas in which smoother approximations generally are required include 3-D graphics (Watt [1993]) and medical imaging (Lehman et al. [1999]). The price paid for smoother approximations is additional computational burden. For general-purpose image processing a *bilinear interpolation* approach that uses the gray levels of the four nearest neighbors usually is adequate. This approach is straightforward. Because the gray level of each of the four integral nearest neighbors of a nonintegral pair of coordinates (x', y') is known, the gray-level value at these coordinates, denoted $v(x', y')$, can be interpolated from the values of its neighbors by using the relationship

$$v(x', y') = ax' + by' + cx'y' + d \tag{5.11-7}$$

where the four coefficients are easily determined from the four equations in four unknowns that can be written using the four known neighbors of (x', y'). When these coefficients have been determined, $v(x', y')$ is computed and this

value is assigned to the location in $f(x, y)$ that yielded the spatial mapping into location (x', y'). It is easy to visualize this procedure with the aid of Fig. 5.33. The exception is that, instead of using the gray-level value of the nearest neighbor to (x', y'), we actually interpolate a value at location (x', y') and use this value for the gray-level assignment at (x, y).

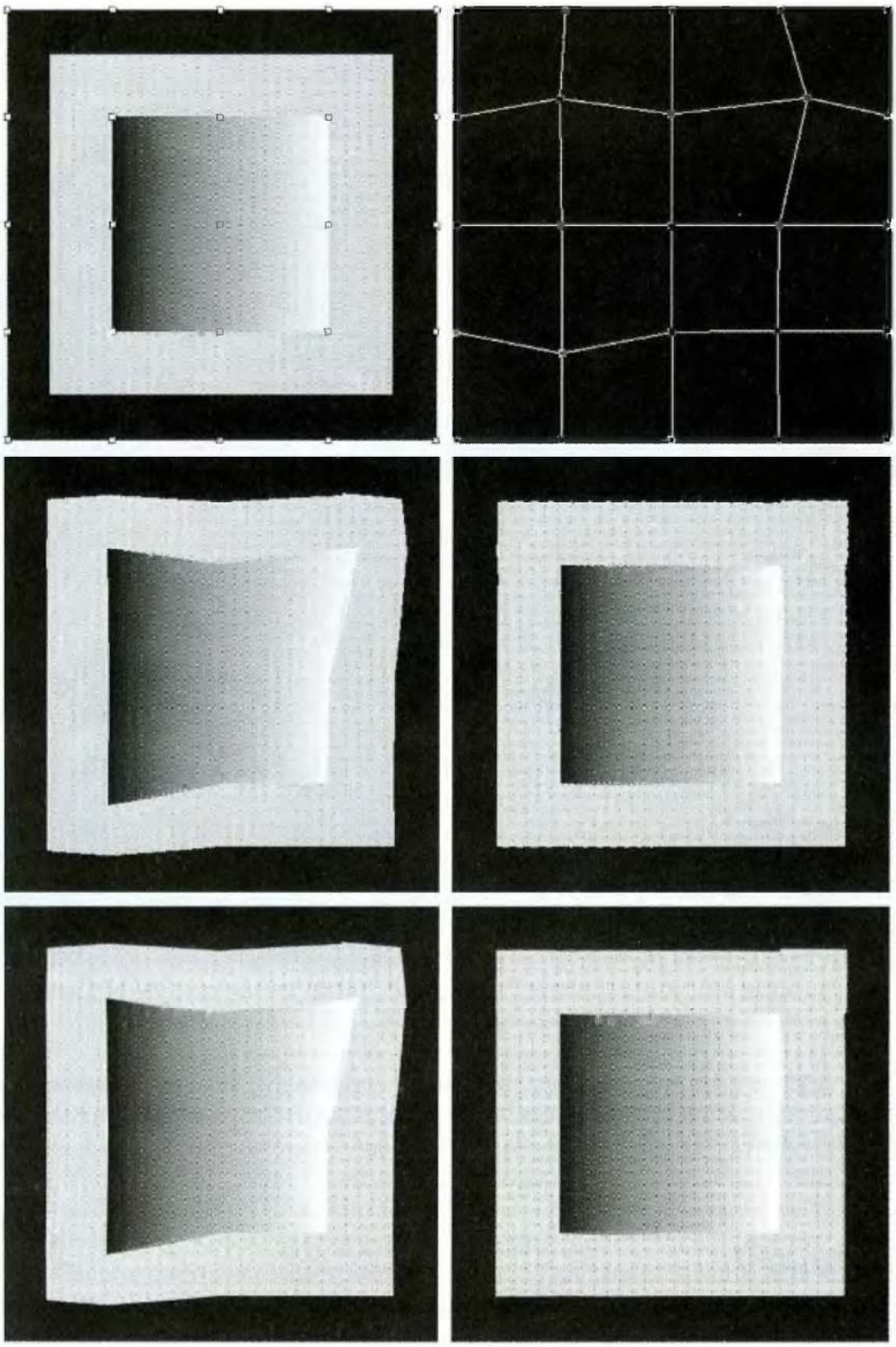
EXAMPLE 5.16: Illustration of geometric transformations.

■ Figure 5.34(a) shows an image with 25 regularly spaced tiepoints (highlighted to enhance visibility of the points in the picture). Figure 5.34(b) shows a simple rearrangement of the tiepoints to create geometric distortion. With reference to the procedure discussed in connection with Eqs. (5.11-5) and (5.11-6), the coefficients of these equations are a result of the mapping from the undistorted to the distorted coordinates. Once the coefficients are known, we have the model, and we can either distort an image (for demonstration purposes) or we can recover an image that was geometrically distorted under the set of conditions defined by the coefficients.

Suppose that we want to distort the image in Fig. 5.34(a). We simply substitute the value of any pixel (x_0, y_0) from that image into Eqs. (5.11-5) and (5.11-6) and generate the corresponding coordinates (x_0', y_0'), which we round off to the closest integer values. The value of the distorted image at that point is given by letting $g(x_0', y_0') = f(x_0, y_0)$, or we can use gray-level interpolation on the values of f in the neighborhood of (x_0, y_0). This is the same process described in connection with Eqs. (5.11-5) and (5.11-6). We are simply applying it in reverse.

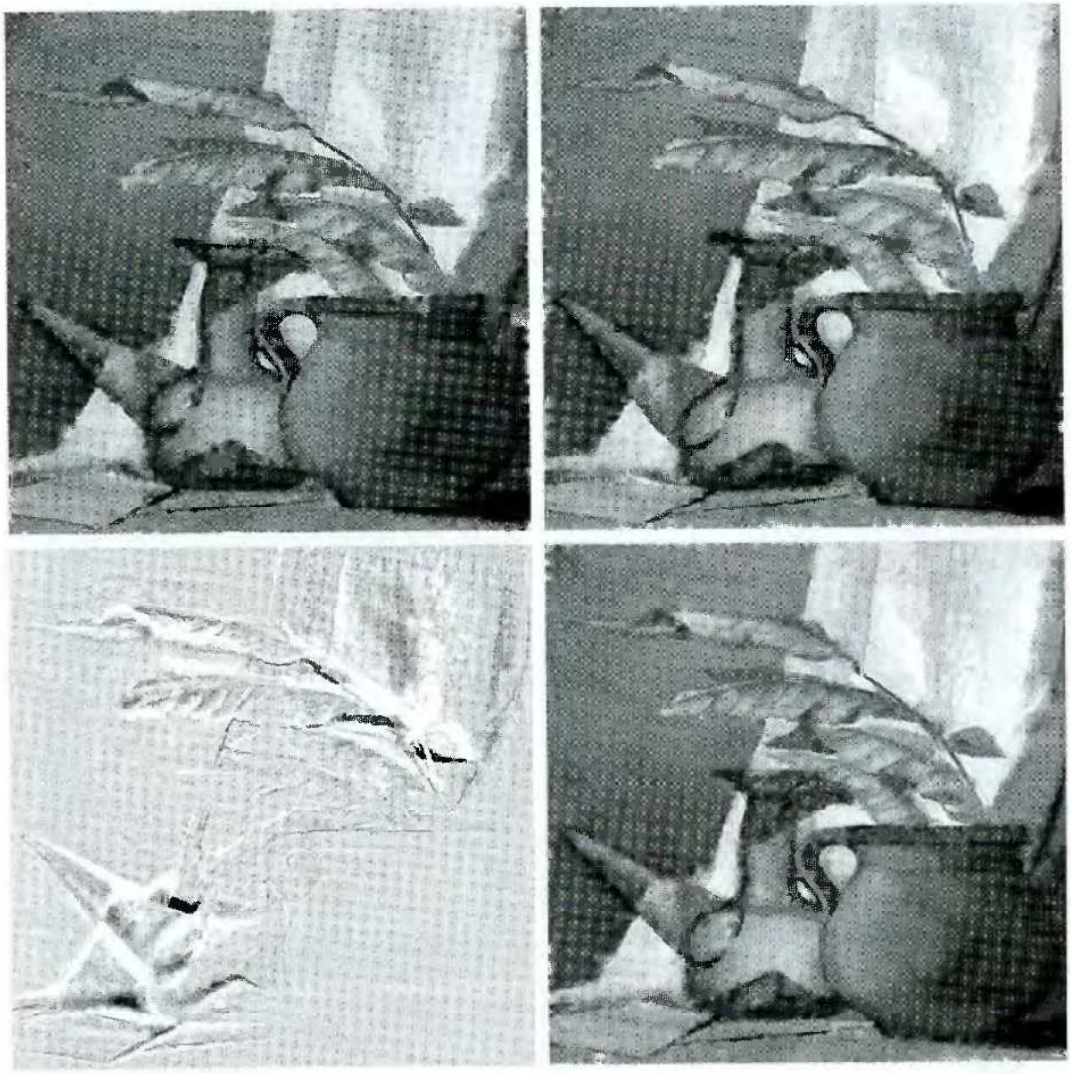
The result of distorting Fig. 5.34(a) by the method just discussed is shown in Fig. 5.34(c), where the nearest neighbor gray-level assignment scheme was used. Note that this is fairly severe distortion. If this were the given image, we would use the method discussed in connection with Eqs. (5.11-5) and (5.11-6), and one of the gray-level assignment techniques discussed in Section 5.11.2. The result of this procedure is shown in Fig. 5.34(d). The nearest neighbor gray-level assignment method was employed again. Note that the geometric correction was reasonable, but there is a significant number of errors in gray-level assignments, especially along the boundaries between the gray and black regions. Figures 5.34(e) and (f) show the same sequence of experiments, but using bilinear gray-level interpolation instead. The improvements are particularly visible in the boundaries between the gray and black regions.

The images just discussed are so regular and have such few gray levels in the sharp boundaries that almost any type of geometric distortion will cause significant degradation. When images have more texture, geometric correction errors tend to be less noticeable. For example, consider Fig. 5.35. Figure 5.35(b) is the result of geometrically distorting Fig. 5.35(a) in the same manner as Fig. 5.34(e). This distortion in Fig. 5.35(b) is not nearly as noticeable. The differences between Figs. 5.35(a) and (b) are not insignificant, as the difference image in Fig. 5.35(c) shows. They simply are not as visible because of the variety of texture in this image. Finally, Fig. 5.35(d) shows the geometrically corrected image. For all practical purposes, this image is of the same quality as the original. ■



a b
c d
e f

FIGURE 5.34 (a) Image showing tiepoints. (b) Tiepoints after geometric distortion. (c) Geometrically distorted image, using nearest neighbor interpolation. (d) Restored result. (e) Image distorted using bilinear interpolation. (f) Restored image.



a b
c d

FIGURE 5.35 (a) An image before geometric distortion. (b) Image geometrically distorted using the same parameters as in Fig. 5.34(e). (c) Difference between (a) and (b). (d) Geometrically restored image.

Summary

The principal results in this chapter are based on the assumption that image degradation can be modeled as a linear, position-invariant process followed by additive noise that is not correlated with image values. Even when these assumptions are not entirely valid, it often is possible to obtain useful results by using the methods developed in the preceding sections.

Some of the restoration techniques derived in this chapter are based on various criteria of optimality. The use of the word *optimal* in this context refers strictly to a mathematical concept, not to optimal response of the human visual system. In fact, the present lack of knowledge about visual perception precludes a general formulation of the image restoration problem that takes into account observer preferences and capabilities. In

view of these limitations, the advantage of the concepts introduced in this chapter is the development of fundamental approaches that have reasonably predictable behavior and are supported by a solid body of knowledge.

As in Chapters 3 and 4, certain restoration tasks, such as random noise reduction, are carried out in the spatial domain using convolution masks. The frequency domain was found ideal for reducing periodic noise and for modeling some important degradations, such as blur caused by motion during image acquisition. We also found the frequency domain to be a useful tool for formulating restoration filters, such as the Wiener and constrained least squares filters.

As mentioned in Chapter 4, the frequency domain offers an intuitive, solid base for experimentation. Once an approach (filter) has been found to perform satisfactorily for a given application, implementation usually is carried out via the design of a digital filter that approximates the frequency-domain solution, but runs much faster in a computer or in a dedicated hardware/firmware system. Digital filter design is beyond the scope of this book, but references relevant to this topic are included in the section that follows.

References and Further Reading

For additional reading on the linear model of degradation presented in Section 5.1, see Castleman [1996] and Pratt [1991]. The book by Peebles [1993] provides an intermediate-level coverage of noise probability density functions and their properties (Section 5.2). The book by Papoulis [1991] is more advanced and covers these concepts in more detail. References for Section 5.3 are Umbaugh [1998], Boie and Cox [1992], Hwang and Haddad [1995], Wilburn [1998], and Eng and Ma [2001]. The general area of adaptive filter design is good background for the adaptive filters discussed in Section 5.3. The book by Haykin [1996] is a good introduction to this topic. The filters in Section 5.4 are direct extensions of the material in Chapter 4. For additional reading on the material of Section 5.5, see Rosenfeld and Kak [1982] and Pratt [1991].

The topic of estimating the degradation function (Section 5.6) is an area of considerable current interest. Some of the early techniques for estimating the degradation function are given in Andrews and Hunt [1977], Rosenfeld and Kak [1982], Bates and McDonnell [1986], and Stark [1987]. Since the degradation function seldom is known exactly, a number of techniques have been proposed over the years, in which specific aspects of restoration are emphasized. For example, Geman and Reynolds [1992], and Hurn and Jennison [1996], deal with issues of preserving sharp transitions in gray levels in an attempt to emphasize sharpness, while Boyd and Meloche [1998] are concerned with restoring thin objects in degraded images. Examples of techniques that deal with image blur are Yitzhaky et al. [1998], Harikumar and Bresler [1999], Mesarović [2000], and Giannakis and Heath [2000]. Restoration of sequences of images also is of considerable interest. The book by Kokaram [1998] provides a good foundation in this area.

The filtering approaches discussed in Sections 5.7 through 5.10 have been explained in various way over the years in numerous books and articles on image processing. There are two major approaches underpinning the development of these filters. One is based on a general formulation using matrix theory, as introduced by Andrews and Hunt [1977]. This approach is elegant and general, but it is difficult for newcomers to the field because it lacks intuitiveness. Approaches based directly on frequency domain filtering (the approach we followed in this chapter) usually are easier to follow by those who first encounter restoration, but lack the unifying mathematical rigor of the matrix approach. Both approaches arrive at the same results, but our experience in teaching this material in a variety of settings indicates that students first entering this field favor the latter approach by a significant margin. Complementary readings for our coverage of the

filtering concepts presented in Sections 5.7 through 5.10 are Castleman [1996], Umbaugh [1998], and Petrou and Bosdogianni [1999]. This last reference also presents a nice tie between two-dimensional frequency domain filters and the corresponding digital filters. On the design of 2-D digital filters, see Lu and Antoniou [1992]. Although we do not cover it in this chapter, the area of computerized axial tomography sometimes is included as a topic in restoration. A good introduction to this field is given by Kak and Slaney [2001]. For further basic reading on the material of Section 5.11, see Sonka et al. [1999]. The papers by Unser et al. [1995] and by Carey et al. [1999] also are of interest.

Problems

See inside front cover

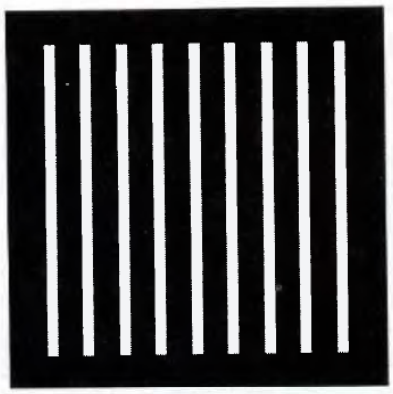
Detailed solutions to the problems marked with a star can be found in the book web site. The site also contains suggested projects based on the material in this chapter.

★5.1 The white bars in the test pattern shown are 7 pixels wide and 210 pixels high. The separation between bars is 17 pixels. What would this image look like after application of

(a) A 3×3 arithmetic mean filter?

(b) A 7×7 arithmetic mean filter?

(c) A 9×9 arithmetic mean filter?



Note: This problem and the ones that follow it, related to filtering this image, may seem a bit tedious. However, they are worth the effort, as they help develop a real understanding of how these filters work. After you understand how a particular filter affects the image, your answer can be a brief verbal description of the result. For example, "the resulting image will consist of vertical bars 3 pixels wide and 206 pixels high." Be sure to describe any deformation of the bars, such as rounded corners. You may ignore image border effects, in which the masks only partially contain image pixels.

5.2 Repeat Problem 5.1 using a geometric mean filter.

★5.3 Repeat Problem 5.1 using a harmonic mean filter.

5.4 Repeat Problem 5.1 using a contraharmonic mean filter with $Q = 1$.

★5.5 Repeat Problem 5.1 using a contraharmonic mean filter with $Q = -1$.

5.6 Repeat Problem 5.1 using a median filter.

★5.7 Repeat Problem.5.1 using a max filter.

5.8 Repeat Problem 5.1 using a min filter.

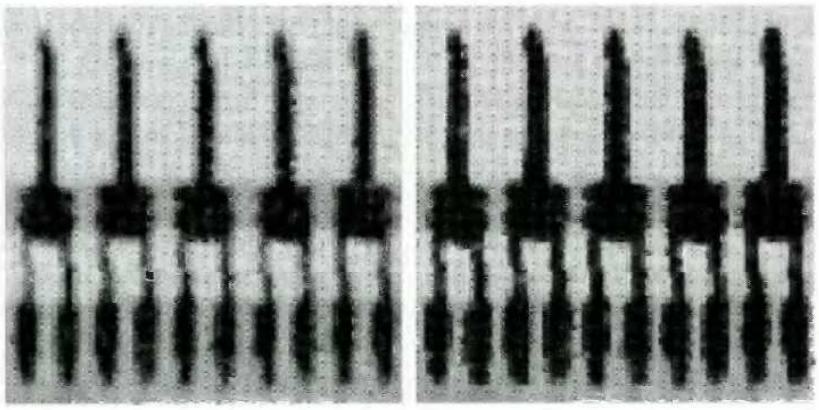
★5.9 Repeat Problem 5.1 using a midpoint filter.

5.10 The two subimages shown were extracted from the top, right corners of Figs. 5.7(c) and (d), respectively. Thus, the subimage on the left is the result of using an

arithmetic mean filter of size 3×3; the other subimage is the result of using a geometric mean filter of the same size.

★ **(a)** Explain why the subimage obtained with geometric mean filtering is less blurred. *Hint:* Start your analysis by examining a 1-D step edge profile (see Fig. 3.38 for an example of a step edge).

(b) Explain why the black components in the right image are thicker.



5.11 Refer to the contraharmonic filter given in Eq. (5.3-6).

(a) Explain why the filter is effective in elimination pepper noise when Q is positive.

(b) Explain why the filter is effective in eliminating salt noise when Q is negative.

(c) Explain why the filter gives poor results (such as the results shown in Fig. 5.9) when the wrong polarity is chosen for Q.

(d) Discuss the behavior of the filter when $Q = -1$.

(e) Discuss (for positive and negative Q) the behavior of the filter in areas of constant gray levels.

★ **5.12** Obtain equations for the bandpass filters corresponding to the bandreject filters in Eqs. (5.4-1) through (5.4-3).

5.13 Obtain equations for the notch pass filters corresponding to the notch reject filters discussed in Section 5.4.3. Show that they become lowpass filters when $u_0 = v_0 = 0$.

★ **5.14** Show that the Fourier transform of the 2-D continuous sine function

$$f(x, y) = A \sin(u_0 x + v_0 y)$$

is the pair of conjugate impulses

$$F(u, v) = -j\frac{A}{2}\left[\delta\left(u - \frac{u_0}{2\pi}, v - \frac{v_0}{2\pi}\right) - \delta\left(u + \frac{u_0}{2\pi}, v + \frac{v_0}{2\pi}\right)\right].$$

Hint: Use the continuous version of the Fourier transform in Eq. (4.2-3), and express the sine in terms of exponentials.

5.15 Start with Eq. (5.4-19) and derive Eq. (5.4-21).

★ **5.16** Consider a linear, position-invariant image degradation system with impulse response

$$h(x - \alpha, y - \beta) = e^{-[(x-\alpha)^2 + (y-\beta)^2]}.$$

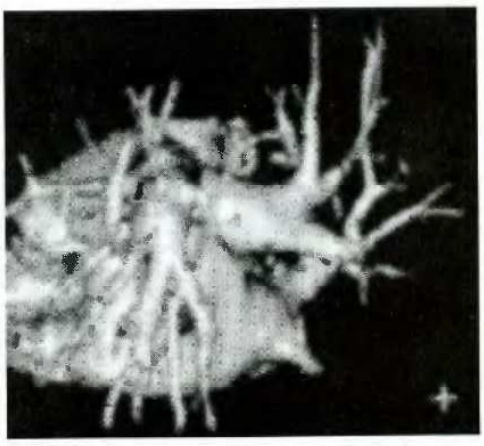
Suppose that the input to the system is an image consisting of a line of infinitesimal width located at $x = a$, and modeled by $f(x, y) = \delta(x - a)$, where δ is the impulse function. Assuming no noise, what is the output image $g(x, y)$?

5.17 During acquisition, an image undergoes uniform linear motion in the vertical direction for a time T_1. The direction of motion then switches to the horizontal direction for a time interval T_2. Assuming that the time it takes the image to change directions is negligible, and that shutter opening and closing times are negligible also, give an expression for the blurring function, $H(u, v)$.

★5.18 Consider the problem of image blurring caused by uniform acceleration in the x-direction. If the image is at rest at time $t = 0$ and accelerates with a uniform acceleration $x_0(t) = at^2/2$ for a time T, find the blurring function $H(u, v)$. You may assume that shutter opening and closing times are negligible.

5.19 A space probe is designed to transmit images from a planet as it approaches it for landing. During the last stages of landing, one of the control thrusters fails, resulting in rapid rotation of the craft about its vertical axis. The images sent during the last two seconds prior to landing are blurred as a consequence of this circular motion. The camera is located in the bottom of the probe, along its vertical axis, and pointing down. Fortunately, the rotation of the craft is also about its vertical axis, so the images are blurred by uniform rotational motion. During the acquisition time of each image the craft rotation was limited to $\pi/8$ radians. The image acquisition process can be modeled as an ideal shutter that is open only during the time the craft rotated the $\pi/8$ radians. You may assume that vertical motion was negligible during image acquisition. Formulate a solution for restoring the images.

★5.20 The image shown is a blurred, 2-D projection of a volumetric rendition of a heart. It is known that each of the cross hairs on the right, bottom part of the image was 3 pixels wide, 30 pixels long, and had gray-level values of 255 before blurring. Provide a step-by-step procedure indicating how you would use the information just given to you obtain the blurring function $H(u, v)$.



Original image courtesy of G.E. Medical Systems.

5.21 A certain X-ray imaging geometry produces a blurring degradation that can be modeled as the convolution of the sensed image with the spatial, circularly symmetric function

$$h(r) = \left[(r^2 - \sigma^2)/\sigma^4\right]e^{-r^2/2\sigma^2}$$

where $r^2 = x^2 + y^2$. Show that the degradation in the frequency domain is given by the expression

$$H(u, v) = -\sqrt{2\pi}\sigma(u^2 + v^2)e^{-2\pi^2\sigma^2(u^2+v^2)}.$$

★ **5.22** Using the transfer function in Problem 5.21, give the expression for a Wiener filter, assuming that the ratio of power spectra of the noise and undegraded signal is a constant.

5.23 Using the transfer function in Problem 5.21, give the resulting expression for the constrained least squares filter.

5.24 Assume that the model in Fig. 5.1 is linear and position invariant and show that the power spectrum of the output is given by

$$|G(u, v)|^2 = |H(u, v)|^2 |F(u, v)|^2 + |N(u, v)|^2.$$

Refer to Eqs. (5.5-17) and (4.2-20).

5.25 Cannon [1974] suggested a restoration filter $R(u, v)$ satisfying the condition

$$|\hat{F}(u, v)|^2 = |R(u, v)|^2 |G(u, v)|^2$$

and based on the premise of forcing the power spectrum of the restored image, $|\hat{F}(u, v)|^2$, to equal the power spectrum of the original image, $|F(u, v)|^2$.

★ **(a)** Find $R(u, v)$ in terms of $|F(u, v)|^2$, $|H(u, v)|^2$, and $|N(u, v)|^2$. *Hint:* Refer to Fig. 5.1, Eq. (5.5-17), and Problem 5.24.

(b) Use your result in (a) to state a result in the form of Eq. (5.8-2).

5.26 An astronomer working with a large-scale telescope observes that her images are a little blurry. The manufacturer tells the astronomer that the unit is operating within specifications. The telescope lenses focus images onto a high-resolution, CCD imaging array, and the images are then converted by the telescope electronics into digital images. Trying to improve the situation by conducting controlled lab experiments with the lenses and imaging sensors is not possible due to the size and weight of the telescope components. The astronomer, having heard about your success as an image processing expert, calls you to help her formulate a digital image processing solution for sharpening the images a little more. How would you go about solving this problem, given that the only images you can obtain are images of stellar bodies?

★ **5.27** A professor of archeology doing research on currency exchange practices during the Roman Empire recently became aware that four Roman coins crucial to his research are listed in the holdings of the British Museum in London. Unfortunately, he was told after arriving there that the coins recently had been stolen. Further research on his part revealed that the museum keeps photographs of every item for which it is responsible. Unfortunately, the photos of the coins in question are blurred to the point where the date and other small markings are not readable. The cause of the blurring was the camera being out of focus when the pictures were taken. As an image processing expert and friend of the professor, you are asked as a favor to determine whether computer processing can be utilized to restore the images to the point where the professor can read the markings. You are told that the original camera used to take the photos is still available, as are other representative coins of the same era. Propose a step-by-step solution to this problem.

5.28 Suppose that, instead of using quadrilaterals, you use triangular regions in Section 5.11 to establish a spatial transformation and gray-level interpolation. What would be the equations analogous to Eqs. (5.11-5), (5.11-6), and (5.11-7) for triangular regions?