Digital Image Processing

Second Edition

Problem Solutions-Students

Rafael C. Gonzalez
Richard E. Woods

Digital Image Processing

Second Edition

Problem Solutions–Student Set

Rafael C. Gonzalez
Richard E. Woods

Prentice Hall

Upper Saddle River, NJ 07458

www.prenhall.com/gonzalezwoods

or

www.imageprocessingbook.com

Revision history

10 9 8 7 6 5 4 3 2 1

Copyright ©1992-2002 by Rafael C. Gonzalez and Richard E. Woods

1 Preface

This abbreviated manual contains detailed solutions to all problems marked with a star in *Digital Image Processing*, 2nd Edition. These solutions can also be downloaded from the book web site (www.imageprocessingbook.com).

2 Solutions (Students)

Problem 2.1

The diameter, x, of the retinal image corresponding to the dot is obtained from similar triangles, as shown in Fig. P2.1. That is,

$$\frac{(d/2)}{0.2} = \frac{(x/2)}{0.014}$$

which gives $x = 0.07d$. From the discussion in Section 2.1.1, and taking some liberties of interpretation, we can think of the fovea as a square sensor array having on the order of 337,000 elements, which translates into an array of size 580×580 elements. Assuming equal spacing between elements, this gives 580 elements and 579 spaces on a line 1.5 mm long. The size of each element and each space is then $s = [(1.5\text{mm})/1,159] = 1.3 \times 10^{-6}$ m. If the size (on the fovea) of the imaged dot is less than the size of a single resolution element, we assume that the dot will be invisible to the eye. In other words, the eye will not detect a dot if its diameter, d, is such that $0.07(d) < 1.3 \times 10^{-6}$ m, or $d < 18.6 \times 10^{-6}$ m.

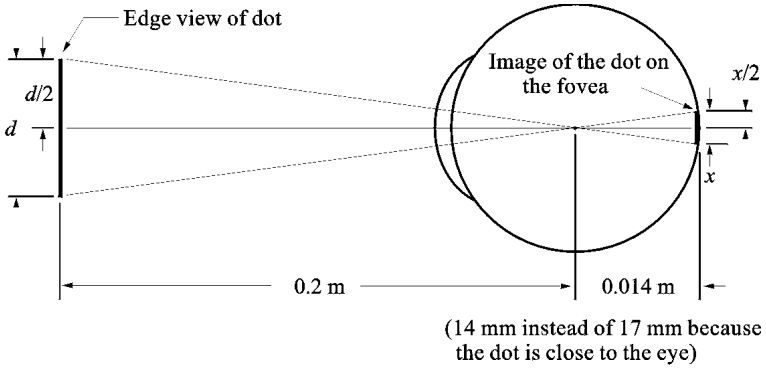


Figure P2.1

Problem 2.3

$$\lambda = c/v = 2.998 \times 10^8 \text{(m/s)}/60\text{(1/s)} = 4.99 \times 10^6 \text{m} = 5000 \text{ Km.}$$

Problem 2.6

One possible solution is to equip a monochrome camera with a mechanical device that sequentially places a red, a green, and a blue pass filter in front of the lens. The strongest camera response determines the color. If all three responses are approximately equal, the object is white. A faster system would utilize three different cameras, each equipped with an individual filter. The analysis would be then based on polling the response of each camera. This system would be a little more expensive, but it would be faster and more reliable. Note that both solutions assume that the field of view of the camera(s) is such that it is completely filled by a uniform color [i.e., the camera(s) is(are) focused on a part of the vehicle where only its color is seen. Otherwise further analysis would be required to isolate the region of uniform color, which is all that is of interest in solving this problem].

Problem 2.9

(a) The total amount of data (including the start and stop bit) in an 8-bit, 1024×1024 image, is $(1024)^2 \times [8 + 2]$ bits. The total time required to transmit this image over a At 56K baud link is $(1024)^2 \times [8 + 2]/56000 = 187.25$ sec or about 3.1 min. (b) At 750K this time goes down to about 14 sec.

Problem 2.11

Let p and q be as shown in Fig. P2.11. Then, (a) S_1 and S_2 are not 4-connected because q is not in the set $N_4(p)$; (b) S_1 and S_2 are 8-connected because q is in the set $N_8(p)$; (c) S_1 and S_2 are *m*-connected because (i) q is in $N_D(p)$, and (ii) the set $N_4(p) \cap N_4(q)$ is empty.

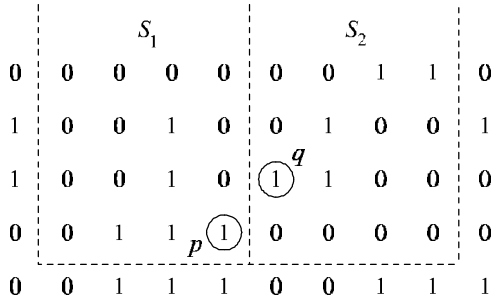


Figure P2.11

Problem 2.12

The solution to this problem consists of defining all possible neighborhood shapes to go from a diagonal segment to a corresponding 4-connected segment, as shown in Fig. P2.12. The algorithm then simply looks for the appropriate match every time a diagonal segment is encountered in the boundary.

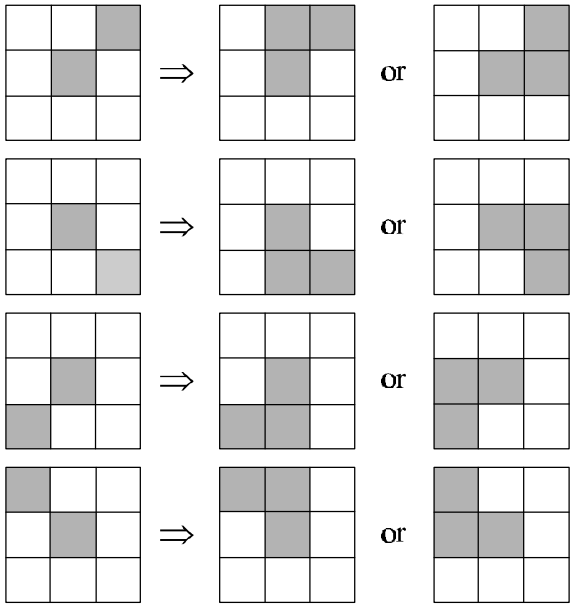


Figure P2.12

Problem 2.15

(a) When $V = \{0, 1\}$, 4-path does not exist between p and q because it is impossible to

get from p to q by traveling along points that are both 4-adjacent and also have values from V. Figure P2.15(a) shows this condition; it is not possible to get to q. The shortest 8-path is shown in Fig. P2.15(b); its length is 4. The length of shortest m- path (shown dashed) is 5. Both of these shortest paths are unique in this case. (b) One possibility for the shortest 4-path when $V = \{1, 2\}$ is shown in Fig. P2.15(c); its length is 6. It is easily verified that another 4-path of the same length exists between p and q. One possibility for the shortest 8-path (it is not unique) is shown in Fig. P2.15(d); its length is 4. The length of a shortest *m*-path (shoen dashed) is 6. This path is not unique.

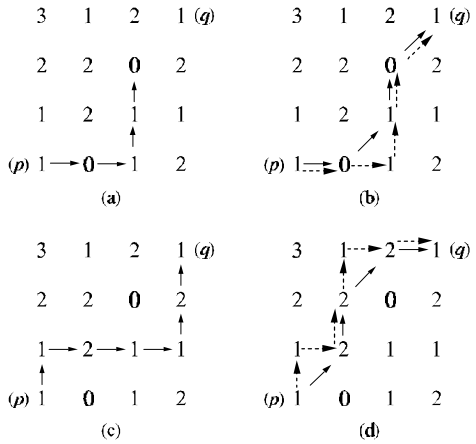


Figure P2.15

Problem 2.16

(a) A shortest 4-path between a point p with coordinates (x, y) and a point q with coordinates (s, t) is shown in Fig. P2.16, where the assumption is that all points along the path are from V. The length of the segments of the path are $|x - s|$ and $|y - t|$, respectively. The total path length is $|x - s| + |y - t|$, which we recognize as the definition of the D_4 distance, as given in Eq. (2.5-16). (Recall that this distance is independent of any paths that may exist between the points.) The D_4 distance obviously is equal to the length of the shortest 4-path when the length of the path is $|x - s| + |y - t|$. This occurs whenever we can get from p to q by following a path whose elements (1) are from V, and (2) are arranged in such a way that we can traverse the path from p to q by making turns in at most two directions (e.g., right and up). (b) The path may of may not be unique, depending on V and the values of the points along the way.

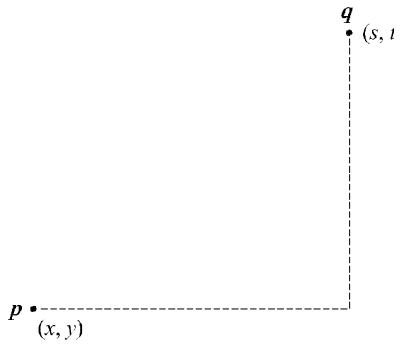


Figure P2.16

Problem 2.18

With reference to Eq. (2.6-1), let H denote the neighborhood sum operator, let S_1 and S_2 denote two different small subimage areas of the same size, and let S_1+S_2 denote the corresponding pixel-by-pixel sum of the elements in S_1 and S_2, as explained in Section 2.5.4. Note that the size of the neighborhood (i.e., number of pixels) is not changed by this pixel-by-pixel sum. The operator H computes the sum of pixel values is a given neighborhood. Then, $H(aS_1 + bS_2)$ means: (1) multiplying the pixels in each of the subimage areas by the constants shown, (2) adding the pixel-by-pixel values from S_1 and S_2 (which produces a single subimage area), and (3) computing the sum of the values of all the pixels in that single subimage area. Let ap_1 and bp_2 denote two arbitrary (but *corresponding*) pixels from $aS_1 + bS_2$. Then we can write

$$\begin{aligned} H(aS_1 + bS_2) &= \sum_{p_1 \in S_1 \text{ and } p_2 \in S_2} ap_1 + bp_2 \\ &= \sum_{p_1 \in S_1} ap_1 + \sum_{p_2 \in S_2} bp_2 \\ &= a \sum_{p_1 \in S_1} p_1 + b \sum_{p_2 \in S_2} p_2 \\ &= aH(S_1) + bH(S_2) \end{aligned}$$

which, according to Eq. (2.6-1), indicates that H is a linear operator.

3 Solutions (Students)

Problem 3.2

(a)

$$s = T(r) = \frac{1}{1 + (m/r)^E}.$$

Problem 3.4

(a) The number of pixels having different gray level values would decrease, thus causing the number of components in the histogram to decrease. Since the number of pixels would not change, this would cause the height some of the remaining histogram peaks to increase in general. Typically, less variability in gray level values will reduce contrast.

Problem 3.5

All that histogram equalization does is remap histogram components on the intensity scale. To obtain a uniform (flat) histogram would require in general that pixel intensities be actually redistributed so that there are L groups of n/L pixels with the same intensity, where L is the number of allowed discrete intensity levels and n is the total number of pixels in the input image. The histogram equalization method has no provisions for this type of (artificial) redistribution process.

Problem 3.8

We are interested in just one example in order to satisfy the statement of the problem. Consider the probability density function shown in Fig. P3.8(a). A plot of the transformation $T(r)$ in Eq. (3.3-4) using this particular density function is shown in Fig. P3.8(b). Because $p_r(r)$ is a probability density function we know from the discussion

in Section 3.3.1 that the transformation $T(r)$ satisfies conditions (a) and (b) stated in that section. However, we see from Fig. P3.8(b) that the inverse transformation from s back to r is not single valued, as there are an infinite number of possible mappings from $s = 1/2$ back to r. It is important to note that the reason the inverse transformation function turned out not to be single valued is the gap in $p_r(r)$ in the interval $[1/4, 3/4]$.

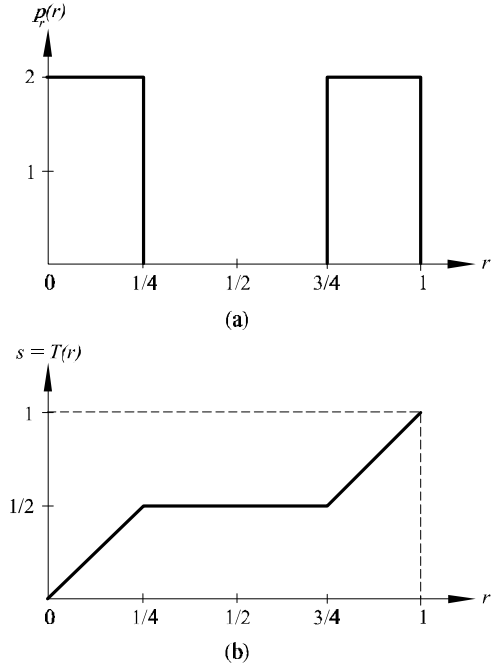


Figure P3.8.

Problem 3.9

(c) If none of the gray levels r_k, $k = 1, 2, \ldots, L-1$, are 0, then $T(r_k)$ will be strictly monotonic. This implies that the inverse transformation will be of finite slope and this will be single-valued.

Problem 3.11

The value of the histogram component corresponding to the *k*th intensity level in a neighborhood is

$$p_r(r_k) = \frac{n_k}{n}$$

for $k = 1, 2, \ldots, K-1$,where n_k is the number of pixels having gray level value r_k, n is the total number of pixels in the neighborhood, and K is the total number of possible gray levels. Suppose that the neighborhood is moved one pixel to the right. This deletes the leftmost column and introduces a new column on the right. The updated histogram then becomes

$$p'_r(r_k) = \frac{1}{n}[n_k - n_{L_k} + n_{R_k}]$$

for $k = 0, 1, \ldots, K-1$, where n_{L_k} is the number of occurrences of level r_k on the left column and n_{R_k} is the similar quantity on the right column. The preceding equation can be written also as

$$p'_r(r_k) = p_r(r_k) + \frac{1}{n}[n_{R_k} - n_{L_k}]$$

for $k = 0, 1, \ldots, K-1$. The same concept applies to other modes of neighborhood motion:

$$p'_r(r_k) = p_r(r_k) + \frac{1}{n}[b_k - a_k]$$

for $k = 0, 1, \ldots, K-1$, where a_k is the number of pixels with value r_k in the neighborhood area deleted by the move, and b_k is the corresponding number introduced by the move.

$$\sigma^2_{\bar{g}} = \sigma^2_f + \frac{1}{K^2}[\sigma^2_{\eta_1} + \sigma^2_{\eta_2} + \cdots + \sigma^2_{\eta_K}]$$

The first term on the right side is 0 because the elements of f are constants. The various $\sigma^2_{\eta_i}$ are simply samples of the noise, which is has variance σ^2_η. Thus, $\sigma^2_{\eta_i} = \sigma^2_\eta$ and we have

$$\sigma^2_{\bar{g}} = \frac{K}{K^2}\sigma^2_\eta = \frac{1}{K}\sigma^2_\eta$$

which proves the validity of Eq. (3.4-5).

Problem 3.14

Let $g(x, y)$ denote the golden image, and let $f(x, y)$ denote any input image acquired during routine operation of the system. Change detection via subtraction is based on computing the simple difference $d(x, y) = g(x, y) - f(x, y)$. The resulting image $d(x, y)$ can be used in two fundamental ways for change detection. One way is use a pixel-by-pixel analysis. In this case we say that $f(x, y)$ is "close enough" to the golden image if all the pixels in $d(x, y)$ fall within a specified threshold band $[T_{min}, T_{max}]$ where T_{min} is negative and T_{max} is positive. Usually, the same value of threshold is

used for both negative and positive differences, in which case we have a band $[-T, T]$ in which all pixels of $d(x, y)$ must fall in order for $f(x, y)$ to be declared acceptable. The second major approach is simply to sum all the pixels in $|d(x, y)|$ and compare the sum against a threshold S. Note that the absolute value needs to be used to avoid errors cancelling out. This is a much cruder test, so we will concentrate on the first approach.

There are three fundamental factors that need tight control for difference-based inspection to work: (1) proper registration, (2) controlled illumination, and (3) noise levels that are low enough so that difference values are not affected appreciably by variations due to noise. The first condition basically addresses the requirement that comparisons be made between corresponding pixels. Two images can be identical, but if they are displaced with respect to each other, comparing the differences between them makes no sense. Often, special markings are manufactured into the product for mechanical or image-based alignment

Controlled illumination (note that "illumination" is not limited to visible light) obviously is important because changes in illumination can affect dramatically the values in a difference image. One approach often used in conjunction with illumination control is intensity scaling based on actual conditions. For example, the products could have one or more small patches of a tightly controlled color, and the intensity (and perhaps even color) of each pixels in the entire image would be modified based on the actual versus expected intensity and/or color of the patches in the image being processed.

Finally, the noise content of a difference image needs to be low enough so that it does not materially affect comparisons between the golden and input images. Good signal strength goes a long way toward reducing the effects of noise. Another (sometimes complementary) approach is to implement image processing techniques (e.g., image averaging) to reduce noise.

Obviously there are a number if variations of the basic theme just described. For example, additional intelligence in the form of tests that are more sophisticated than pixel-by-pixel threshold comparisons can be implemented. A technique often used in this regard is to subdivide the golden image into different regions and perform different (usually more than one) tests in each of the regions, based on expected region content.

Problem 3.17

(a) Consider a 3×3 mask first. Since all the coefficients are 1 (we are ignoring the 1/9

scale factor), the net effect of the lowpass filter operation is to add all the gray levels of pixels under the mask. Initially, it takes 8 additions to produce the response of the mask. However, when the mask moves one pixel location to the right, it picks up only one new column. The new response can be computed as

$$R_{\text{new}} = R_{\text{old}} - C_1 + C_3$$

where C_1 is the sum of pixels under the first column of the mask before it was moved, and C_3 is the similar sum in the column it picked up after it moved. This is the basic box-filter or moving-average equation. For a 3×3 mask it takes 2 additions to get C_3 (C_1 was already computed). To this we add one subtraction and one addition to get R_{new}. Thus, a total of 4 arithmetic operations are needed to update the response after one move. This is a recursive procedure for moving from left to right along one row of the image. When we get to the end of a row, we move down one pixel (the nature of the computation is the same) and continue the scan in the opposite direction.

For a mask of size $n \times n$, $(n - 1)$ additions are needed to obtain C_3, plus the single subtraction and addition needed to obtain R_{new}, which gives a total of $(n + 1)$ arithmetic operations after each move. A brute-force implementation would require $n^2 - 1$ additions after each move.

Problem 3.19

(a) There are n^2 points in an $n \times n$ median filter mask. Since n is odd, the median value, ζ, is such that there are $(n^2 - 1)/2$ points with values less than or equal to ζ and the same number with values greater than or equal to ζ. However, since the area A (number of points) in the cluster is less than one half n^2, and A and n are integers, it follows that A is always less than or equal to $(n^2 - 1)/2$. Thus, even in the extreme case when all cluster points are encompassed by the filter mask, there are not enough points in the cluster for any of them to be equal to the value of the median (remember, we are assuming that all cluster points are lighter or darker than the background points). Therefore, if the center point in the mask is a cluster point, it will be set to the median value, which is a background shade, and thus it will be "eliminated" from the cluster. This conclusion obviously applies to the less extreme case when the number of cluster points encompassed by the mask is less than the maximum size of the cluster.

Problem 3.20

(a) Numerically sort the n^2 values. The median is

$$\zeta = [(n^2 + 1)/2]\text{-th largest value.}$$

(b) Once the values have been sorted one time, we simply delete the values in the trailing edge of the neighborhood and insert the values in the leading edge in the appropriate locations in the sorted array.

Problem 3.22

From Fig. 3.35, the vertical bars are 5 pixels wide, 100 pixels high, and their separation is 20 pixels. The phenomenon in question is related to the horizontal separation between bars, so we can simplify the problem by considering a single scan line through the bars in the image. The key to answering this question lies in the fact that the distance (in pixels) between the onset of one bar and the onset of the next one (say, to its right) is 25 pixels. Consider the scan line shown in Fig. P3.22. Also shown is a cross section of a 25×25 mask. The response of the mask is the average of the pixels that it encompasses. We note that when the mask moves one pixel to the right, it loses on value of the vertical bar on the left, but it picks up an identical one on the right, so the response doesn't change. In fact, the number of pixels belonging to the vertical bars and contained within the mask does not change, regardless of where the mask is located (as long as it is contained within the bars, and not near the edges of the set of bars). The fact that the number of bar pixels under the mask does not change is due to the peculiar separation between bars and the width of the lines in relation to the 25-pixel width of the mask This constant response is the reason no white gaps is seen in the image shown in the problem statement. Note that this constant response does not happen with the 23×23 or the 45×45 masks because they are not "synchronized" with the width of the bars and their separation.

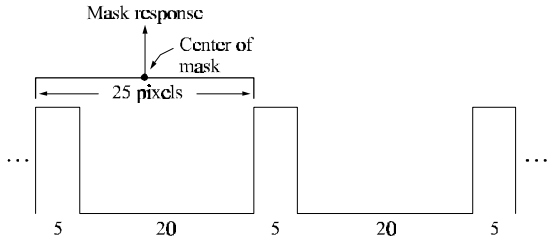


Figure P3.22

Problem 3.25

The Laplacian operator is defined as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

for the unrotated coordinates and as

$$\nabla^2 f = \frac{\partial^2 f}{\partial x'^2} + \frac{\partial^2 f}{\partial y'^2}.$$

for rotated coordinates. It is given that

$$x = x' \cos\theta - y' \sin\theta \quad \text{and} \quad y = x' \sin\theta + y' \cos\theta$$

where θ is the angle of rotation. We want to show that the right sides of the first two equations are equal. We start with

$$\begin{aligned} \frac{\partial f}{\partial x'} &= \frac{\partial f}{\partial x}\frac{\partial x}{\partial x'} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial x'} \\ &= \frac{\partial f}{\partial x}\cos\theta + \frac{\partial f}{\partial y}\sin\theta \end{aligned}$$

Taking the partial derivative of this expression again with respect to x' yields

$$\frac{\partial^2 f}{\partial x'^2} = \frac{\partial^2 f}{\partial x^2}\cos^2\theta + \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial y}\right)\sin\theta\cos\theta + \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial x}\right)\cos\theta\sin\theta + \frac{\partial^2 f}{\partial y^2}\sin^2\theta$$

Next, we compute

$$\begin{aligned} \frac{\partial f}{\partial y'} &= \frac{\partial f}{\partial x}\frac{\partial x}{\partial y'} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial y'} \\ &= -\frac{\partial f}{\partial x}\sin\theta + \frac{\partial f}{\partial y}\cos\theta \end{aligned}$$

Taking the derivative of this expression again with respect to y' gives

$$\frac{\partial^2 f}{\partial y'^2} = \frac{\partial^2 f}{\partial x^2}\sin^2\theta - \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial y}\right)\cos\theta\sin\theta - \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial x}\right)\sin\theta\cos\theta + \frac{\partial^2 f}{\partial y^2}\cos^2\theta$$

Adding the two expressions for the second derivatives yields

$$\frac{\partial^2 f}{\partial x'^2} + \frac{\partial^2 f}{\partial y'^2} = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

which proves that the Laplacian operator is independent of rotation.

Problem 3.27

Consider the following equation:

$$\begin{aligned} f(x,y) - \nabla^2 f(x,y) &= f(x,y) - [f(x+1,y) + f(x-1,y) + f(x,y+1) \\ &\quad + f(x,y-1) - 4f(x,y)] \\ &= 6f(x,y) - [f(x+1,y) + f(x-1,y) + f(x,y+1) \\ &\quad + f(x,y-1) + f(x,y)] \\ &= 5\{1.2f(x,y) - \\ &\quad \frac{1}{5}[f(x+1,y) + f(x-1,y) + f(x,y+1) \\ &\quad + f(x,y-1) + f(x,y)]\} \\ &= 5\left[1.2f(x,y) - \overline{f}(x,y)\right] \end{aligned}$$

where $\overline{f}(x,y)$ denotes the average of $f(x,y)$ in a predefined neighborhood that is centered at (x,y) and includes the center pixel and its four immediate neighbors. Treating the constants in the last line of the above equation as proportionality factors, we may write

$$f(x,y) - \nabla^2 f(x,y) \sim f(x,y) - \overline{f}(x,y).$$

The right side of this equation is recognized as the definition of unsharp masking given in Eq. (3.7-7). Thus, it has been demonstrated that subtracting the Laplacian from an image is proportional to unsharp masking.

4 Solutions (Students)

Problem 4.1

By direct substitution of $f(x)$ [Eq. (4.2-6)] into $F(u)$ [Eq. (4.2-5)]:

$$\begin{aligned} F(u) &= \frac{1}{M}\sum_{x=0}^{M-1}\left[\sum_{r=0}^{M-1}F(r)e^{j2\pi rx/M}\right]e^{-j2\pi ux/M} \\ &= \frac{1}{M}\sum_{r=0}^{M-1}F(r)\sum_{x=0}^{M-1}e^{j2\pi rx/M}e^{-j2\pi ux/M} \\ &= \frac{1}{M}F(u)\,[M] \\ &= F(u) \end{aligned}$$

where the third step follows from the orthogonality condition given in the problem statement. Substitution of $F(u)$ into $f(x)$ is handled in a similar manner.

Problem 4.4

See errata sheet in book web site

An important aspect of this problem is to recognize that the quantity $(u^2 + v^2)$ can be replaced by the distance squared, $D^2(u, v)$. This reduces the problem to one variable, which is notationally easier to manage. Rather than carry an award capital letter throughout the development, we define $w^2 \triangleq D^2(u, v) = (u^2 + v^2)$. Then we proceed as follows:

$$H(w) = e^{-w^2/2\sigma^2}.$$

The inverse Fourier transform is

$$\begin{aligned} h(z) &= \int_{-\infty}^{\infty}H(w)e^{j2\pi wz}dw \\ &= \int_{-\infty}^{\infty}e^{-w^2/2\sigma^2}e^{j2\pi wz}dw \\ &= \int_{-\infty}^{\infty}e^{-\frac{1}{2\sigma^2}\left[w^2-j4\pi\sigma^2wz\right]}dw. \end{aligned}$$

We now make use of the identity

$$e^{-\frac{(2\pi)^2 z^2 \sigma^2}{2}} e^{\frac{(2\pi)^2 z^2 \sigma^2}{2}} = 1.$$

Inserting this identity in the preceding integral yields

$$\begin{aligned} h(z) &= e^{-\frac{(2\pi)^2 z^2 \sigma^2}{2}} \int_{-\infty}^{\infty} e^{-\frac{1}{2\sigma^2}\left[w^2 - j4\pi\sigma^2 wz - (2\pi)^2\sigma^4 z^2\right]} dw \\ &= e^{-\frac{(2\pi)^2 z^2 \sigma^2}{2}} \int_{-\infty}^{\infty} e^{-\frac{1}{2\sigma^2}\left[w - j2\pi\sigma^2 z\right]^2} dw. \end{aligned}$$

Next we make the change of variable $r = w - j2\pi\sigma^2 z$. Then, $dr = dw$ and the above integral becomes

$$h(z) == e^{-\frac{(2\pi)^2 z^2 \sigma^2}{2}} \int_{-\infty}^{\infty} e^{-\frac{r^2}{2\sigma^2}}\, dr.$$

Finally, we multiply and divide the right side of this equation by $\sqrt{2\pi}\sigma$:

$$h(z) = \sqrt{2\pi}\sigma e^{-\frac{(2\pi)^2 z^2 \sigma^2}{2}} \left[\frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{\infty} e^{-\frac{r^2}{2\sigma^2}}\, dr\right].$$

The expression inside the brackets is recognized as a Gaussian probability density function, whose integral from $-\infty$ to ∞ is 1. Then,

$$h(z) = \sqrt{2\pi}\sigma e^{-\frac{(2\pi)^2 z^2 \sigma^2}{2}}.$$

Going back to two spatial variables gives the final result:$h(x, y) = \sqrt{2\pi}\sigma\, e^{-2\pi^2\sigma^2(x^2+y^2)}$.

Problem 4.6

(a) We note first that $(-1)^{x+y} = e^{j\pi(x+y)}$. Then,

$$\begin{aligned} \Im\left[f(x,y)e^{j\pi(x+y)}\right] &= \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[f(x,y)e^{j\pi(x+y)}\right] e^{-j2\pi(ux/M\, +\, vy/N)} \\ &= \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[f(x,y)e^{-j2\pi(-\frac{xM}{2M}-\frac{yN}{2N})}\right] \\ &\qquad e^{-j2\pi(ux/M\, +\, vy/N)} \\ &= \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi\left(x[u-\frac{M}{2}]/M + y[v-\frac{N}{2}]/N\right)} \\ &= F(u - M/2, v - N/2). \end{aligned}$$

Problem 4.8

With reference to Eq. (4.4-1), all the highpass filters in discussed in Section 4.4 can be expressed a 1 minus the transfer function of lowpass filter (which we know do not have

an impulse at the origin). The inverse Fourier transform of 1 gives an impulse at the origin in the highpass spatial filters.

Problem 4.11

Starting from Eq. (4.2-30), we easily find the expression for the definition of continuous convolution in one dimension:

$$f(x) * g(x) = \int_{-\infty}^{\infty} f(\alpha)g(x-\alpha)d\alpha.$$

The Fourier transform of this expression is

$$\begin{aligned}\Im\left[f(x) * g(x)\right] &= \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} f(\alpha)g(x-\alpha)d\alpha\right]e^{-j2\pi ux}dx \\ &= \int_{-\infty}^{\infty} f(\alpha)\left[\int_{-\infty}^{\infty} g(x-\alpha)e^{-j2\pi ux}dx\right]d\alpha.\end{aligned}$$

The term inside the inner brackets is the Fourier transform of $g(x-\alpha)$. But,

$$\Im\left[g(x-\alpha)\right] = G(u)e^{-j2\pi u\alpha}$$

so

$$\begin{aligned}\Im\left[f(x) * g(x)\right] &= \int_{-\infty}^{\infty} f(\alpha)\left[G(u)e^{-j2\pi u\alpha}\right]d\alpha \\ &= G(u)\int_{-\infty}^{\infty} f(\alpha)e^{-j2\pi u\alpha}d\alpha \\ &= G(u)F(u).\end{aligned}$$

This proves that multiplication in the frequency domain is equal to convolution in the spatial domain. The proof that multiplication in the spatial domain is equal to convolution in the spatial domain is done in similar way.

Problem 4.13

(a) One application of the filter gives:

$$\begin{aligned}G(u,v) &= H(u,v)F(u,v) \\ &= e^{-D^2(u,v)/2D_0^2}F(u,v).\end{aligned}$$

Similarly, K applications of the filter would give

$$G_K(u,v) = e^{-KD^2(u,v)/2D_0^2}F(u,v).$$

The inverse DFT of $G_K(u,v)$ would give the image resulting from K passes of the Gaussian filter. If K is "large enough," the Gaussian LPF will become a notch pass filter, passing only $F(0,0)$. We know that this term is equal to the average value of the image. So, there is a value of K after which the result of repeated lowpass filtering

will simply produce a constant image. The value of all pixels on this image will be equal to the average value of the original image. Note that the answer applies even as K approaches infinity. In this case the filter will approach an impulse at the origin, and this would still give us $F(0,0)$ as the result of filtering.

Problem 4.15

The problem statement gives the form of the difference in the x-direction. A similar expression gives the difference in the y-direction. The filtered function in the spatial domain then is:

$$g(x,y) = f(x,y) - f(x+1,y) + f(x,y) - f(x,y+1).$$

From Eq. (4.6-2),

$$\begin{aligned} G(u,v) &= F(u,v) - F(u,v)e^{j2\pi u/M} + F(u,v) - F(u,v)e^{j2\pi v/N} \\ &= [1 - e^{j2\pi u/M}]F(u,v) + [1 - e^{j2\pi v/N}]F(u,v) \\ &= H(u,v)F(u,v), \end{aligned}$$

where $H(u,v)$ is the filter function:

$$H(u,v) = -2j\left[\sin(\pi u/M)e^{j\pi u/M} + \sin(\pi v/N)e^{j\pi v/N}\right].$$

(b) To see that this is a highpass filter, it helps to express the filter function in the form of our familiar centered functions:

$$H(u,v) = -2j\left[\sin(\pi[u - M/2]/M)e^{j\pi u/M} + \sin(\pi[v - N/2]/N)e^{j\pi v/N}\right].$$

Consider one variable for convenience. As u ranges from 0 to M, $H(u,v)$ starts at its maximum (complex) value of $2j$ for $u = 0$ and decreases from there. When $u = M/2$ (the center of the shifted function), A similar argument is easily carried out when considering both variables simultaneously.. The value of $H(u,v)$ starts increasing again and achieves the maximum value of $2j$ again when $u = M$. Thus, this filter has a value of 0 a the origin and increases with increasing distance from the origin. This is the characteristic of a highpass filter. A similar argument is easily carried out when considering both variables simultaneously.

Problem 4.18

The answer is no. The Fourier transform is a linear process, while the square and square roots involved in computing the gradient are nonlinear operations. The Fourier transform could be used to compute the derivatives (as differences—see Prob.4.15), but the

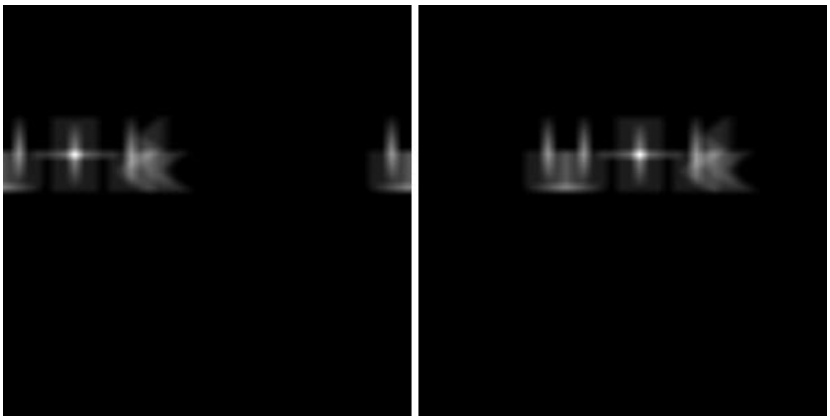
squares, square root, or absolute values must be computed directly in the spatial domain.

Problem 4.21

Recall that the reason for padding is to establish a "buffer" between the periods that are implicit in the DFT. Imagine the image on the left being duplicated infinitely many times to cover the xy-plane. The result would be a checkerboard, with each square being in the checkerboard being the image (and the black extensions). Now imagine doing the same thing to the image on the right. The results would be indistinguishable. Thus, either form of padding accomplishes the same separation between images, as desired.

Problem 4.24

(a) and (b) See Figs. P4.24(a) and (b).



Figures P4.24(a) and (b)

Problem 4.25

Because $M = 2^n$, we can write Eqs. (4.6-47) and (4.6-48) respectively as

$$m(n) = \frac{1}{2}Mn$$

and

$$a(n) = Mn.$$

Proof by induction begins by showing that both equations hold for $n = 1$:

$$m(1) = \frac{1}{2}(2)(1) = 1 \qquad \text{and} \qquad a(1) = (2)(1) = 2.$$

We know these results to be correct from the discussion in Section 4.6.6. Next, we assume that the equations hold for n. Then, we are required to prove that they also are true for $n+1$. From Eq. (4.6-45),

$$m(n+1) = 2m(n) + 2^n.$$

Substituting $m(n)$ from above,

$$\begin{aligned} m(n+1) &= 2\left(\frac{1}{2}Mn\right) + 2^n \\ &= 2\left(\frac{1}{2}2^n n\right) + 2^n \\ &= 2^n(n+1) \\ &= \frac{1}{2}\left(2^{n+1}\right)(n+1). \end{aligned}$$

Therefore, Eq. (4.6-47) is valid for all n.

From Eq. (4.6-46),

$$a(n+1) = 2a(n) + 2^{n+1}.$$

Substituting the above expression for $a(n)$ yields

$$\begin{aligned} a(n+1) &= 2Mn + 2^{n+1} \\ &= 2(2^n n) + 2^{n+1} \\ &= 2^{n+1}(n+1) \end{aligned}$$

which completes the proof.

5 Solutions (Students)

Problem 5.1

The solutions to (a), (b), and (c) are shown in Fig. P5.1, from left to right:

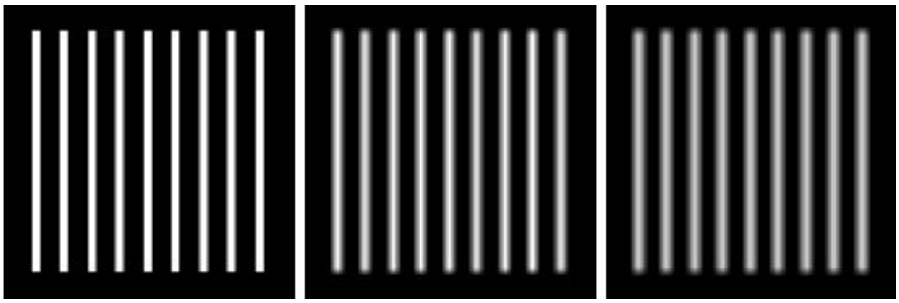


Figure P5.1

Problem 5.3

The solutions to (a), (b), and (c) are shown in Fig. P5.3, from left to right:

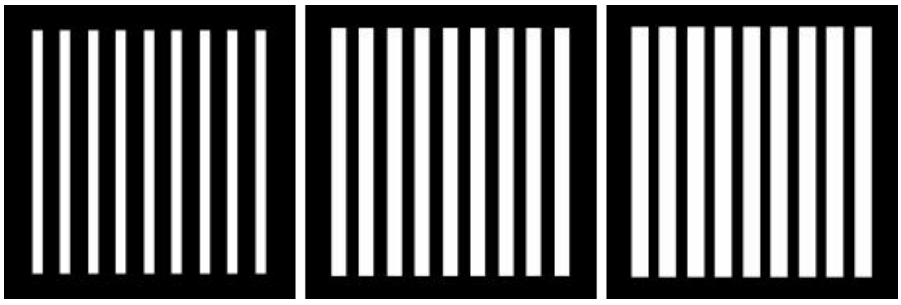


Figure P5.3

Problem 5.5

The solutions to (a), (b), and (c) are shown in Fig. P5.5, from left to right:

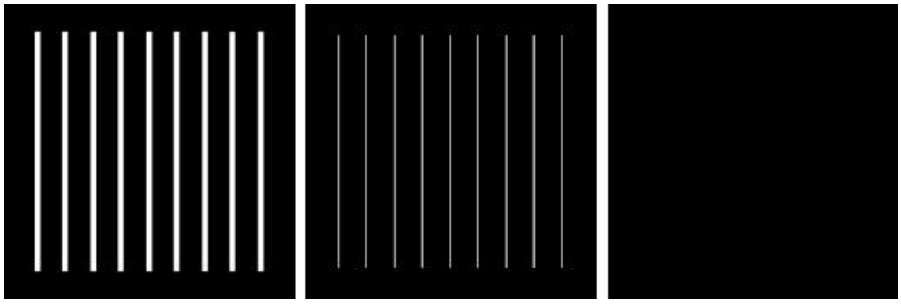


Figure P5.5

Problem 5.7

The solutions to (a), (b), and (c) are shown in Fig. P5.7, from left to right:

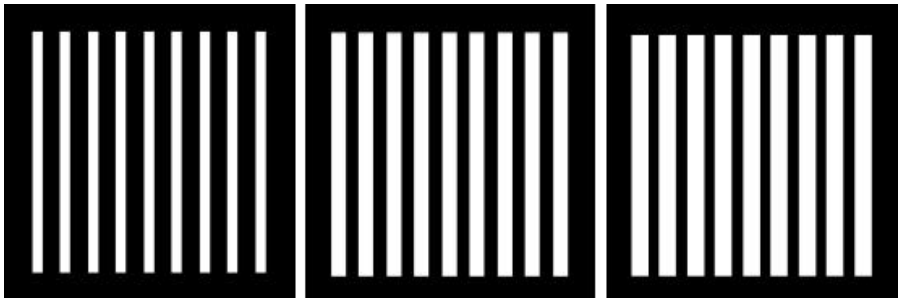


Figure P5.7

Problem 5.9

The solutions to (a), (b), and (c) are shown in Fig. P5.9, from left to right:

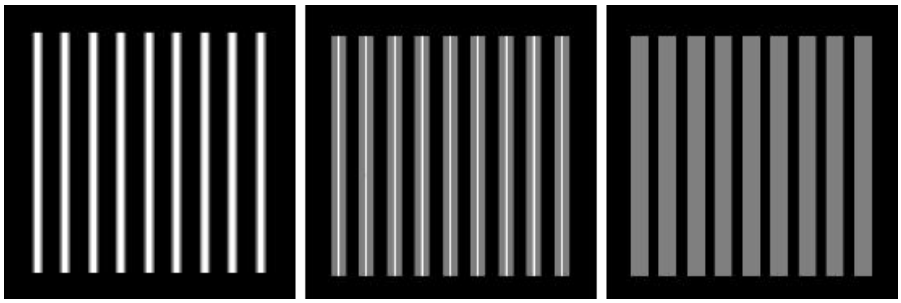


Figure P5.9

Problem 5.10

(a) The key to this problem is that the geometric mean is zero whenever any pixel is zero. Draw a profile of an ideal edge with a few points valued 0 and a few points valued 1. The geometric mean will give only values of 0 and 1, whereas the arithmetic mean will give intermediate values (blur).

Problem 5.12

A bandpass filter is obtained by subtracting the corresponding bandreject filter from 1:

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

Then:

(a) Ideal bandpass filter:

$$H_{\text{Ibp}}(u,v) = \begin{cases} 0 & \text{if } D(u,v) < D_0 - \frac{W}{2} \\ 1 & \text{if } D_0 - \frac{W}{2} \le D(u,v) \le D_0 + \frac{W}{2} \\ 0 & D(u,v) > D_0 + \frac{W}{2} \end{cases}.$$

(b) Butterworth bandpass filter:

$$\begin{aligned} H_{\text{Bbp}}(u,v) &= 1 - \frac{1}{1 + \left[\frac{D(u,v)W}{D^2(u,v) - D_0^2}\right]^{2n}} \\ &= \frac{\left[\frac{D(u,v)W}{D^2(u,v) - D_0^2}\right]^{2n}}{1 + \left[\frac{D(u,v)W}{D^2(u,v) - D_0^2}\right]^{2n}}. \end{aligned}$$

(c) Gaussian bandpass filter:

$$\begin{aligned} H_{\text{Gbp}}(u,v) &= 1 - \left[1 - e^{-\frac{1}{2}\left[\frac{D^2(u,v)-D_0^2}{D(u,v)W}\right]^2}\right] \\ &= e^{-\frac{1}{2}\left[\frac{D^2(u,v)-D_0^2}{D(u,v)W}\right]^2}. \end{aligned}$$

Problem 5.14

We proceed as follows:

$$\begin{aligned} F(u,v) &= \iint_{-\infty}^{\infty} f(x,y)e^{-j2\pi(ux+vy)}dx\,dy \\ &= \iint_{-\infty}^{\infty} A\sin(u_0x+v_0y)e^{-j2\pi(ux+vy)}dx\,dy. \end{aligned}$$

Using the exponential definition of the sine function:

$$\sin\theta = \frac{1}{2j}\left(e^{j\theta} - e^{-j\theta}\right)$$

gives us

$$\begin{aligned} F(u,v) &= \frac{-jA}{2}\iint_{-\infty}^{\infty}\left[e^{j(u_0x+v_0y)} - e^{-j(u_0x+v_0y)}\right]e^{-j2\pi(ux+vy)}dx\,dy \\ &= \frac{-jA}{2}\left[\iint_{-\infty}^{\infty} e^{j2\pi(u_0x/2\pi+v_0y/2\pi)}e^{-j2\pi(ux+vy)}dx\,dy\right] - \\ &\quad \frac{jA}{2}\left[\iint_{-\infty}^{\infty} e^{-j2\pi(u_0x/2\pi+v_0y/2\pi)}e^{-j2\pi(ux+vy)}dx\,dy\right]. \end{aligned}$$

These are the Fourier transforms of the functions

$$1 \times e^{j2\pi(u_0x/2\pi+v_0y/2\pi)}$$

and

$$1 \times e^{-j2\pi(u_0x/2\pi+v_0y/2\pi)}$$

respectively. The Fourier transform of the 1 gives an impulse at the origin, and the exponentials shift the origin of the impulse, as discussed in Section 4.6.1. Thus,

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

Problem 5.16

From Eq. (5.5-13),

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

It is given that $f(x,y) = \delta(x-a)$, so $f(\alpha,\beta) = \delta(\alpha-a)$. Then, using the impulse response given in the problem statement,

$$\begin{aligned} g(x,y) &= \iint_{-\infty}^{\infty} \delta(\alpha-a) e^{-\left[(x-\alpha)^2+(y-\beta)^2\right]}\, d\alpha\, d\beta \\ &= \iint_{-\infty}^{\infty} \delta(\alpha-a) e^{-\left[(x-\alpha)^2\right]} e^{-\left[(y-\beta)^2\right]}\, d\alpha\, d\beta \\ &= \int_{-\infty}^{\infty} \delta(\alpha-a) e^{-\left[(x-\alpha)^2\right]}\, d\alpha \int_{-\infty}^{\infty} e^{-\left[(y-\beta)^2\right]}\, d\beta \\ &= e^{-\left[(x-a)^2\right]} \int_{-\infty}^{\infty} e^{-\left[(y-\beta)^2\right]}\, d\beta \end{aligned}$$

where we used the fact that the integral of the impulse is nonzero only when $\alpha = a$. Next, we note that

$$\int_{-\infty}^{\infty} e^{-\left[(y-\beta)^2\right]}\, d\beta = \int_{-\infty}^{\infty} e^{-\left[(\beta-y)^2\right]}\, d\beta$$

which is in the form of a constant times a Gaussian density with variance $\sigma^2 = 1/2$ or standard deviation $\sigma = 1/\sqrt{2}$. In other words,

$$e^{-\left[(\beta-y)^2\right]} = \sqrt{2\pi(1/2)} \left[\frac{1}{\sqrt{2\pi(1/2)}} e^{-(1/2)\left[\frac{(\beta-y)^2}{(1/2)}\right]} \right].$$

The integral from minus to plus infinity of the quantity inside the brackets is 1, so

$$g(x,y) = \sqrt{\pi} e^{-\left[(x-a)^2\right]}$$

which is a blurred version of the original image.

Problem 5.18

Following the procedure in Section 5.6.3,

$$\begin{aligned} H(u,v) &= \int_0^T e^{-j2\pi u x_0(t)} dt \\ &= \int_0^T e^{-j2\pi u\left[(1/2)at^2\right]} dt \\ &= \int_0^T e^{-j\pi u a t^2} dt \\ &= \int_0^T \left[\cos(\pi u a t^2) - j\sin(\pi u a t^2)\right] dt \\ &= \sqrt{\frac{T^2}{2\pi u a T^2}} \left[C(\sqrt{\pi u a}T) - jS(\sqrt{\pi u a}T)\right] \end{aligned}$$

where

$$C(x) = \sqrt{\frac{2\pi}{T}} \int_0^x \cos t^2 dt$$

and

$$S(x) = \sqrt{\frac{2}{\pi}} \int_0^x \sin t^2 dt.$$

These are Fresnel cosine and sine integrals. They can be found, for example, the *Handbook of Mathematical Functions*, by Abramowitz, or other similar reference.

Problem 5.20

Measure the average value of the background. Set all pixels in the image, except the cross hairs, to that gray level. Denote the Fourier transform of this image by $G(u, v)$. Since the characteristics of the cross hairs are given with a high degree of accuracy, we can construct an image of the background (of the same size) using the background gray levels determined previously. We then construct a model of the cross hairs in the correct location (determined from he given image) using the provided dimensions and gray level of the crosshairs. Denote by $F(u, v)$ the Fourier transform of this new image . The ratio $G(u, v)/F(u, v)$ is an estimate of the blurring function $H(u, v)$. In the likely event of vanishing values in $F(u, v)$, we can construct a radially-limited filter using the method discussed in connection with Fig. 5.27. Because we know $F(u, v)$ and $G(u, v)$, and an estimate of $H(u, v)$, we can also refine our estimate of the blurring function by substituting G and H in Eq. (5.8-3) and adjusting K to get as close as possible to a good result for $F(u, v)$ [the result can be evaluated visually by taking the inverse Fourier transform]. The resulting filter in either case can then be used to deblur the image of the heart, if desired.

Problem 5.22

This is a simple plugin problem. Its purpose is to gain familiarity with the various terms of the Wiener filter. From Eq. (5.8-3),

$$H_W(u, v) = \left[\frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + K} \right]$$

where

$$\begin{aligned} |H(u, v)|^2 &= H^*(u, v)H(u, v) \\ &= 2\pi\sigma^2(u^2 + v^2)^2 e^{-4\pi^2\sigma^2(x^2+y^2)}. \end{aligned}$$

Then,

$$H_W(u, v) = -\left[\frac{\sqrt{2\pi}\sigma(u^2 + v^2)e^{-2\pi^2\sigma^2(x^2+y^2)}}{\left[2\pi\sigma^2(u^2 + v^2)^2 e^{-4\pi^2\sigma^2(x^2+y^2)}\right] + K} \right].$$

Problem 5.25

(a) It is given that

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

From Problem 5.24,

$$\left|\hat{F}(u,v)\right|^2 = |R(u,v)|^2 \left[|H(u,v)|^2 |F(u,v)|^2 + |N(u,v)|^2\right] .$$

Forcing $\left|\hat{F}(u,v)\right|^2$ to equal $|F(u,v)|^2$ gives

$$R(u,v) = \left[\frac{|F(u,v)|^2}{|H(u,v)|^2 |F(u,v)|^2 + |N(u,v)|^2}\right]^{1/2} .$$

Problem 5.27

The basic idea behind this problem is to use the camera and representative coins to model the degradation process and then utilize the results in an inverse filter operation. The principal steps are as follows:

1. Select coins as close as possible in size and content as the lost coins. Select a background that approximates the texture and brightness of the photos of the lost coins.

2. Set up the museum photographic camera in a geometry as close as possible to give images that resemble the images of the lost coins (this includes paying attention to illumination). Obtain a few test photos. To simplify experimentation, obtain a TV camera capable of giving images that resemble the test photos. This can be done by connecting the camera to an image processing system and generating digital images, which will be used in the experiment.

3. Obtain sets of images of each coin with different lens settings. The resulting images should approximate the aspect angle, size (in relation to the area occupied by the background), and blur of the photos of the lost coins.

4. The lens setting for each image in (3) is a model of the blurring process for the corresponding image of a lost coin. For each such setting, remove the coin and background and replace them with a small, bright dot on a uniform background, or other mechanism to approximate an impulse of light. Digitize the impulse. Its Fourier transform is the transfer function of the blurring process.

5. Digitize each (blurred) photo of a lost coin, and obtain its Fourier transform. At this point, we have $H(u,v)$ and $G(u,v)$ for each coin.

6. Obtain an approximation to $F(u,v)$ by using a Wiener filter. Equation (5.8-3) is particularly attractive because it gives an additional degree of freedom (K) for experimenting.

7. The inverse Fourier transform of each approximate $F(u, v)$ gives the restored image. In general, several experimental passes of these basic steps with various different settings and parameters are required to obtain acceptable results in a problem such as this.

6 Solutions (Students)

Problem 6.2

Denote by c the given color, and let its coordinates be denoted by (x_0, y_0). The distance between c and c_1 is

$$d(c, c_1) = \left[(x_0 - x_1)^2 + (y_0 - y_1)^2\right]^{1/2}.$$

Similarly the distance between c_1 and c_2

$$d(c_1, c_2) = \left[(x_1 - x_2)^2 + (y_1 - y_2)^2\right]^{1/2}.$$

The percentage p_1 of c_1 in c is

$$p_1 = \frac{d(c_1, c_2) - d(c, c_1)}{d(c_1, c_2)} \times 100.$$

The percentage p_2 of c_2 is simply $p_2 = 100 - p_1$. In the preceding equation we see, for example, that when $c = c_1$, then $d(c, c_1) = 0$ and it follows that $p_1 = 100\%$ and $p_2 = 0\%$. Similarly, when $d(c, c_1) = d(c_1, c_2)$, it follows that $p_1 = 0\%$ and $p_2 = 100\%$. Values in between are easily seen to follow from these simple relations.

Problem 6.4

Use color filters sharply tuned to the wavelengths of the colors of the three objects. Thus, with a specific filter in place, only the objects whose color corresponds to that wavelength will produce a predominant response on the monochrome camera. A motorized filter wheel can be used to control filter position from a computer. If one of the colors is white, then the response of the three filters will be approximately equal and high. If one of the colors is black, the response of the three filters will be approximately equal and low.

Problem 6.6

For the image given, the maximum intensity and saturation requirement means that the

RGB component values are 0 or 1. We can create the following table with 0 and 255 representing black and white, respectively:

Table P6.6

Color	R	G	B	Mono R	Mono G	Mono B
Black	0	0	0	0	0	0
Red	1	0	0	255	0	0
Yellow	1	1	0	255	255	0
Green	0	1	0	0	255	0
Cyan	0	1	1	0	255	255
Blue	0	0	1	0	0	255
Magenta	1	0	1	255	0	255
White	1	1	1	255	255	255
Gray	0.5	0.5	0.5	128	128	128

Thus, we get the monochrome displays shown in Fig. P6.6.

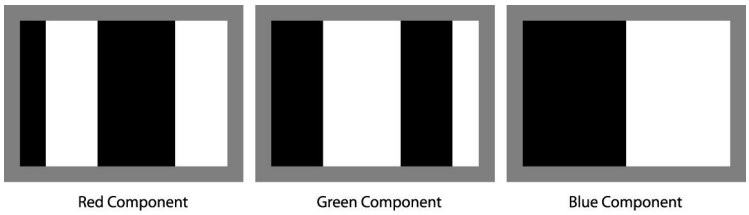


Figure P6.6

Problem 6.8

(a) All pixel values in the Red image are 255. In the Green image, the first column is all 0's; the second column all 1's; and so on until the last column, which is composed of all 255's. In the Blue image, the first row is all 255's; the second row all 254's, and so on until the last row which is composed of all 0's.

Problem 6.10

Equation (6.2-1) reveals that each component of the CMY image is a function of a single component of the corresponding RGB image—C is a function of R, M of G, and Y of B. For clarity, we will use a prime to denote the CMY components. From Eq. (6.5-6),

we know that

$$s_i = kr_i$$

for $i = 1, 2, 3$ (for the R, G, and B components). And from Eq. (6.2-1), we know that the CMY components corresponding to the r_i and s_i (which we are denoting with primes) are

$$r_i' = 1 - r_i$$

and

$$s_i' = 1 - s_i.$$

Thus,

$$r_i = 1 - r_i'$$

and

$$s_i' = 1 - s_i = 1 - kr_i = 1 - k\,(1 - r_i')$$

so that

$$s_i' = kr_i' + (1 - k)\,.$$

Problem 6.12

Using Eqs. (6.2-2) through (6.2-4), we get the results shown in Table P6.12.

Table P6.12

Color	R	G	B	H	S	I	Mono H	Mono S	Mono I
Black	0	0	0	–	0	0	–	–	0
Red	1	0	0	0	1	0.33	0	255	85
Yellow	1	1	0	0.17	1	0.67	43	255	170
Green	0	1	0	0.33	1	0.33	85	255	85
Cyan	0	1	1	0.5	1	0.67	128	255	170
Blue	0	0	1	0.67	1	0.33	170	255	85
Magenta	1	0	1	0.83	1	0.67	213	255	170
White	1	1	1	–	0	1	–	0	255
Gray	0.5	0.5	0.5	–	0	0.5	–	0	128

Note that, in accordance with Eq. (6.2-2), hue is undefined when $R = G = B$ since $\theta = \cos^{-1}\left(\frac{0}{0}\right)$. In addition, saturation is undefined when $R = G = B = 0$ since Eq. (6.2-3) yields $S = 1 - \frac{3\min(0)}{3 \cdot 0} = 1 - \frac{0}{0}$. Thus, we get the monochrome display shown in Fig. P6.12.

Figure P6.12

Problem 6.14

There are two important aspects to this problem. One is to approach it in HSI space and the other is to use polar coordinates to create a hue image whose values grow as a function of angle. The center of the image is the middle of whatever image area is used. Then, for example, the values of the hue image along a radius when the angle is 0° would be all 0's. The angle then is incremented by, say, one degree, and all the values along that radius would be 1's, and so on. Values of the saturation image decrease linearly in all radial directions from the origin. The intensity image is just a specified constant. With these basics in mind it is not difficult to write a program that generates the desired result.

Problem 6.16

(a) It is given that the colors in Fig. 6.16(a) are primary spectrum colors. It also is given that the gray-level images in the problem statement are 8-bit images. The latter condition means that hue (angle) can only be divided into a maximum number of 256 values. Since hue values are represented in the interval from 0° to 360° this means that for an 8-bit image the increments between contiguous hue values are now $360/255$. Another way of looking at this is that the entire [0, 360] hue scale is compressed to the range [0, 255]. Thus, for example, yellow (the first primary color we encounter), which is 60° now becomes 43 (the closest integer) in the integer scale of the 8-bit image shown in the problem statement. Similarly, green, which is 120° becomes 85 in this image. From this we easily compute the values of the other two regions as being 170 and 213. The region in the middle is pure white [equal proportions of red green and blue in Fig. 6.61(a)] so its hue by definition is 0. This also is true of the black background.

Problem 6.18

Using Eq. (6.2-3), we see that the basic problem is that many different colors have the same saturation value. This was demonstrated in Problem 6.12, where pure red, yellow, green, cyan, blue, and magenta all had a saturation of 1. That is, as long as any one of the RGB components is 0, Eq. (6.2-3) yields a saturation of 1.

Consider RGB colors (1, 0, 0) and (0, 0.59, 0), which represent a red and a green. The HSI triplets for these colors [per Eq. (6.4-2) through (6.4-4)] are (0, 1, 0.33) and (0.33, 1, 0.2), respectively. Now, the complements of the beginning RGB values (see Section 6.5.2) are (0, 1, 1) and (1, 0.41, 1), respectively; the corresponding colors are cyan and magenta. Their HSI values [per Eqs. (6.4-2) through (6.4-4)] are (0.5, 1, 0.66) and (0.83, 0.48, 0.8), respectively. Thus, for the red, a starting saturation of 1 yielded the cyan "complemented" saturation of 1, while for the green, a starting saturation of 1 yielded the magenta "complemented" saturation of 0.48. That is, the same starting saturation resulted in two different "complemented" saturations. Saturation alone is not enough information to compute the saturation of the complemented color.

Problem 6.20

The RGB transformations for a complement [from Fig. 6.33(b)] are:

$$s_i = 1 - r_i$$

where $i = 1, 2, 3$ (for the R, G, and B components). But from the definition of the CMY space in Eq. (6.2-1), we know that the CMY components corresponding to r_i and s_i, which we will denote using primes, are

$$r_i' = 1 - r_i$$
$$s_i' = 1 - s_i$$

Thus,

$$r_i = 1 - r_i'$$

and

$$s_i' = 1 - s_i = 1 - (1 - r_i) = 1 - (1 - (1 - r_i'))$$

so that

$$s' = 1 - r_i'$$

Problem 6.22

Based on the discussion is Section 6.5.4 and with reference to the color wheel in Fig. 6.32, we can decrease the proportion of yellow by (1) decreasing yellow, (2) increasing blue, (3) increasing cyan and magenta, or (4) decreasing red and green.

Problem 6.24

The conceptually simplest approach is to transform every input image to the HSI color space, perform histogram specification per the discussion in Section 3.3.2 on the intensity (I) component only (leaving H and S alone), and convert the resulting intensity component with the original hue and saturation components back to the starting color space.

Problem 6.27

(a) The cube is composed of 6 intersecting planes in RGB space. The general equation for such planes is

$$a\,z_R + b\,z_G + c\,z_B + d = 0$$

where a, b, c, and d are parameters and the z's are the components of any point (vector) $\mathbf{z}$ in RGB space lying on the plane. If an RGB point $\mathbf{z}$ does not lie on the plane, and its coordinates are substituted in the preceding equation, then equation will give either a positive or a negative value; it will not yield zero. We say that $\mathbf{z}$ lies on the positive or negative side of the plane, depending on whether the result is positive or negative. We can change the positive side of a plane by multiplying its coefficients (except d) by -1. Suppose that we test the point $\mathbf{a}$ given in the problem statement to see whether it is on the positive or negative side each of the six planes composing the box, and change the coefficients of any plane for which the result is negative. Then, $\mathbf{a}$ will lie on the positive side of all planes composing the bounding box. In fact all points inside the bounding box will yield positive values when their coordinates are substituted in the equations of the planes. Points outside the box will give at least one negative or zero value. Thus, the method consists of substituting an unknown color point in the equations of all six planes. If all the results are positive, the point is inside the box; otherwise it is outside the box. A flow diagram is asked for in the problem statement to make it simpler to evaluate the student's line of reasoning.

7 Solutions (Students)

Problem 7.2

A mean approximation pyramid is formed by forming 2×2 block averages. Since the starting image is of size 4×4, $J = 2$, and $f(x, y)$ is placed in level 2 of the mean approximation pyramid. The level 1 approximation is (by taking 2×2 block averages over $f(x, y)$ and subsampling):

$$\begin{bmatrix} 3.5 & 5.5 \\ 11.5 & 13.5 \end{bmatrix}$$

and the level 0 approximation is similarly [8.5]. The completed mean approximation pyramid is

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} \begin{bmatrix} 3.5 & 5.5 \\ 11.5 & 13.5 \end{bmatrix} \begin{bmatrix} 8.5 \end{bmatrix}.$$

Since no interpolation filtering is specified, pixel replication is used in the generation of the mean prediction residual pyramid levels. Level 0 of the prediction residual pyramid is the lowest resolution approximation, [8.5]. The level 2 prediction residual is obtained by upsampling the level 1 approximation and subtracting it from the level 2 (original image). Thus, we get

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix} - \begin{bmatrix} 3.5 & 3.5 & 5.5 & 5.5 \\ 3.5 & 3.5 & 5.5 & 5.5 \\ 11.5 & 11.5 & 13.5 & 13.5 \\ 11.5 & 11.5 & 13.5 & 13.5 \end{bmatrix}$$

$$= \begin{bmatrix} -2.5 & -1.5 & -2.5 & -1.5 \\ 1.5 & 2.5 & 1.5 & 2.5 \\ -2.5 & -1.5 & -2.5 & -1.5 \\ 1.5 & 2.5 & 1.5 & 2.5 \end{bmatrix}.$$

Similarly, the level 1 prediction residual is obtained by upsampling the level 0 approximation and subtracting it from the level 1 approximation to yield

$$\begin{bmatrix} 3.5 & 5.5 \\ 11.5 & 13.5 \end{bmatrix} - \begin{bmatrix} 8.5 & 8.5 \\ 8.5 & 8.5 \end{bmatrix} = \begin{bmatrix} -5 & -3 \\ 3 & 5 \end{bmatrix}.$$

The mean prediction residual pyramid is therefore

$$\begin{bmatrix} -2.5 & -1.5 & -2.5 & -1.5 \\ 1.5 & 2.5 & 1.5 & 2.5 \\ -2.5 & -1.5 & -2.5 & -1.5 \\ 1.5 & 2.5 & 1.5 & 2.5 \end{bmatrix} \begin{bmatrix} -5 & -3 \\ 3 & 5 \end{bmatrix} [8.5].$$

Problem 7.3

The number of elements in a $J+1$ level pyramid is bounded by 4/3 (see Section 7.1.1):

$$2^{2J}\left[1 + \frac{1}{(4)^1} + \frac{1}{(4)^2} + \cdots + \frac{1}{(4)^J}\right] \leq \frac{4}{3}2^{2J}$$

for $J > 0$. We can generate the following table:

Table P7.3

J	Pyramid Elements	Compression Ratio
0	1	1
1	5	$5/4 = 1.25$
2	21	$21/16 = 1.3125$
3	85	$85/86 = 1.328$
$\vdots$		
∞		$4/3 = 1.33$

All but the trivial case $(J = 0)$ are expansions. The expansion factor is a function of and bounded by 4/3 or 1.33.

Problem 7.4

(a) The QMF filters must satisfy Eqs. (7.1-9) and (7.1-10). From Table 7.1, $G_0(z) = H_0(z)$ and $H_1(z) = H_0(-z)$, so $H_1(-z) = H_0(z)$. Thus, beginning with Eq. (7.1-9),

$$\begin{aligned} H_0(-z)G_0(z) + H_1(-z)G_1(z) &= 0 \\ H_0(-z)H_0(z) - H_0(z)H_0(-z) &= 0 \\ 0 &= 0. \end{aligned}$$

Similarly, beginning with Eq. (7.1-10) and substituting for $H_1(z)$, $G_0(z)$, and $G_1(z)$ from rows 2, 3, and 4 of Table 7.1, we get

$$\begin{aligned} H_0(z)G_0(z) + H_1(z)G_1(z) &= 2 \\ H_0(z)H_0(z) + H_0(-z)[-H_0(-z)] &= 2 \\ H_0^2(z) - H_0^2(-z) &= 2 \end{aligned}$$

which is the design equation for the $H_0(z)$ prototype filter in row 1 of the table.

PROBLEM 7.7 Reconstruction is performed by reversing the decomposition process; that is, by replacing the downsamplers with upsamplers and the analysis filters by their synthesis filter counterparts, as shown in Fig. P7.7.

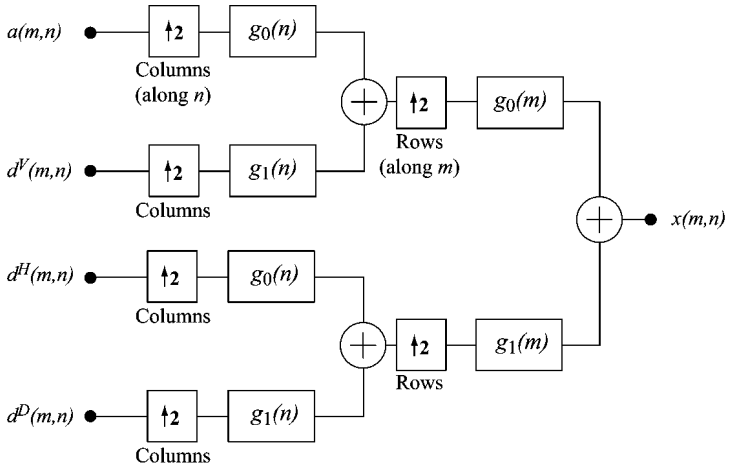


Figure P7.7

Problem 7.10

(a) The basis is orthonormal and the coefficients are computed by the vector equivalent

of Eq. (7.2-5):

$$\begin{aligned}\alpha_0 &= \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} \\ &= \frac{5\sqrt{2}}{2} \\ \alpha_1 &= \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} \\ &= \frac{\sqrt{2}}{2}\end{aligned}$$

so,

$$\begin{aligned}\frac{5\sqrt{2}}{2}\varphi_0 + \frac{\sqrt{2}}{2}\varphi_1 &= \frac{5\sqrt{2}}{2} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} + \frac{\sqrt{2}}{2} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix} \\ &= \begin{bmatrix} 3 \\ 2 \end{bmatrix}.\end{aligned}$$

Problem 7.13

From Eq. (7.2-19) we find that

$$\begin{aligned}\psi_{3,3}(x) &= 2^{3/2}\psi(2^3x - 3) \\ &= 2\sqrt{2}\psi(8x - 3)\end{aligned}$$

and using the Haar wavelet function definition from Eq. (7.2-30), obtain the plot shown in Fig. P7.13.

To express $\psi_{3,3}(x)$ as a function of scaling functions, we employ Eq. (7.2-28) and the Haar wavelet vector defined in Example 7.6—that is, $h_\psi(0) = 1/\sqrt{2}$ and $h_\psi(1) = -1/\sqrt{2}$. Thus we get

$$\psi(x) = \sum_n h_\psi(n)\sqrt{2}\varphi(2x - n)$$

so that

$$\begin{aligned}\psi(8x - 3) &= \sum_n h_\psi(n)\sqrt{2}\varphi(2[8x - 3] - n) \\ &= \frac{1}{\sqrt{2}}\sqrt{2}\varphi(16x - 6) + \left(\frac{-1}{\sqrt{2}}\right)\sqrt{2}\varphi(16x - 7) \\ &= \varphi(16x - 6) - \varphi(16x - 7).\end{aligned}$$

Then, since $\psi_{3,3} = 2\sqrt{2}\psi(8x - 3)$,

$$\begin{aligned}\psi_{3,3} &= 2\sqrt{2}\psi(8x-3) \\ &= 2\sqrt{2}\varphi(16x-6) - 2\sqrt{2}\varphi(16x-7).\end{aligned}$$

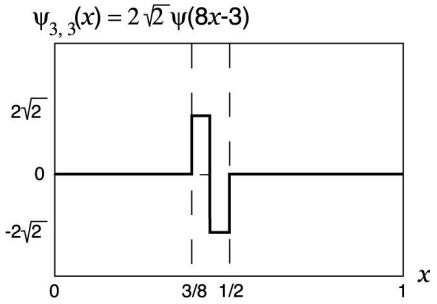


Figure P7.13

Problem 7.17

Intuitively, the continuous wavelet transform (CWT) calculates a "resemblance index" between the signal and the wavelet at various scales and translations. When the index is large, the resemblance is strong; else it is weak. Thus, if a function is similar to itself at different scales, the resemblance index will be similar at different scales. The CWT coefficient values (the index) will have a characteristic pattern. As a result, we can say that the function whose CWT is shown is self-similar—like a fractal signal.

Problem 7.18

(b) The DWT is a better choice when we need a space saving representation that is sufficient for reconstruction of the original function or image. The CWT is often easier to interpret because the built-in redundancy tends to reinforce traits of the function or image. For example, see the self-similarity of Problem 7.18.

Problem 7.19

The filter bank is the first bank in Fig. (7.17), as shown in Fig. P7.19:

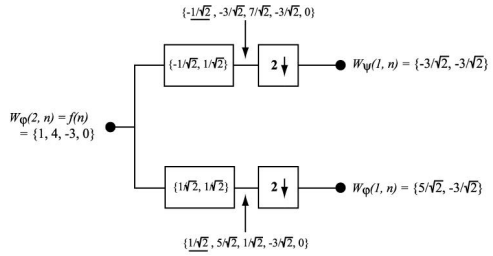


Figure P7.19

PROBLEM 7.21 (a) Input $\varphi(n) = \{1, 1, 1, 1, 1, 1, 1, 1\} = \varphi_{0,0}(n)$ for a three-scale wavelet transform with Haar scaling and wavelet functions. Since wavelet transform coefficients measure the similarity of the input to the basis functions, the resulting transform is

$$\{W_\varphi(0,0), W_\psi(0,0), W_\psi(1,0), W_\psi(1,1), W_\psi(2,0), W_\psi(2,1), W_\psi(2,2)$$
$$W_\psi(2,3)\} = \{2\sqrt{2}, 0, 0, 0, 0, 0, 0, 0\}$$

The $W_\varphi(0,0)$ term can be computed using Eq. (7.3-5) with $j_0 = k = 0$.

PROBLEM 7.22 They are both multi-resolution representations that employ a single reduced-resolution approximation image and a series of "difference" images. For the FWT, these "difference" images are the transform detail coefficients; for the pyramid, they are the prediction residuals.

To construct the approximation pyramid that corresponds to the transform in Fig. 7.8(a), we will use the FWT^{-1} 2-d synthesis bank of Fig. 7.22(c). First, place the 64×64 approximation "coefficients" from Fig. 7.8(a) at the top of the pyramid being constructed. Then use it, along with 64×64 horizontal, vertical, and diagonal detail coefficients from the upper-left of Fig. 7.8(a), to drive the filter bank inputs in Fig. 7.22(c). The output will be a 128×128 approximation of the original image and should be used as the next level of the approximation pyramid. The 128×128 approximation is then used with the three 128×128 detail coefficient images in the upper 1/4 of the transform in Fig. 7.8(a) to drive the synthesis filter bank in Fig. 7.22(c) a second time—producing a 256×256 approximation that is placed as the next level of the approximation pyramid. This process is then repeated a third time to recover the 512×512 original image, which is placed at the bottom of the approximation pyramid. Thus, the approximation pyramid would have 4 levels.

PROBLEM 7.24 As can be seen in the sequence of images that are shown, the DWT

is not shift invariant. If the input is shifted, the transform changes. Since all original images in the problem are 128×128, they become the $W_\varphi(7, m, n)$ inputs for the FWT computation process. The filter bank of Fig. 7.22(a) can be used with $j + 1 = 7$. For a single scale transform, transform coefficients $W_\varphi(6, m, n)$ and $W_\psi^i(6, m, n)$ for $i = H, V, D$ are generated. With Haar wavelets, the transformation process subdivides the image into non-overlapping 2×2 blocks and computes 2-point averages and differences (per the scaling and wavelet vectors). Thus, there are no horizontal, vertical, or diagonal detail coefficients in the first two transforms shown; the input images are constant in all 2×2 blocks (so all differences are 0). If the original image is shifted by 1 pixel, detail coefficients are generated since there are then 2×2 areas that are not constant. This is the case in the third transform shown.

8 Solutions (Students)

Problem 8.3

Table P8.3 shows the data, its 6-bit code, the IGS sum for each step, the actual IGS 3-bit code and its equivalent decoded value, the error between the decoded IGS value and the input values, and the squared error.

Table P8.3

Data	6-bit Code	Sum	IGS Code	Decoded IGS	Error	Sq. Error
		000000				
12	001100	001100	001	8	4	16
12	001100	010000	010	16	-4	16
13	001101	001101	001	8	5	25
13	001101	010010	010	16	-3	9
10	001010	001100	001	8	2	4
13	001101	010001	010	16	-3	9
57	111001	111001	111	56	1	1
54	110110	110111	110	48	6	36

Problem 8.5

(b) For 1100111, construct the following three bit odd parity word:

$$c_1 = h_1 \oplus h_3 \oplus h_5 \oplus h_7 = 1 \oplus 0 \oplus 1 \oplus 1 = 1$$
$$c_2 = h_2 \oplus h_3 \oplus h_6 \oplus h_7 = 1 \oplus 0 \oplus 1 \oplus 1 = 1$$
$$c_4 = h_4 \oplus h_5 \oplus h_6 \oplus h_7 = 0 \oplus 1 \oplus 1 \oplus 1 = 1$$

A parity word of 111_2 indicates that bit 7 is in error. The correctly decoded binary value is 0110_2. In a similar manner, the parity words for 1100110 and 1100010 are 000 and 101, respectively. The decoded values are identical and are 0110.

Problem 8.6

The conversion factors are computed using the logarithmic relationship

$$\log_a x = \frac{1}{\log_b a} \log_b x$$

Thus, 1 Hartley = 3.3219 bits and 1 nat = 1.4427 bits.

Problem 8.7

Let the set of source symbols be $\{a_1, a_2, ..., a_q\}$ and have probabilities

$$\mathbf{z} = [P(a_1), P(a_2), ..., P(a_q)]^T .$$

Then, using Eq. (8.3-3) and the fact that the sum of all $P(a_i)$ is 1, we get

$$\begin{aligned} \log q - H(\mathbf{z}) &= \sum_{i=1}^{q} P(a_i) \log q + \sum_{i=1}^{q} P(a_i) \log P(a_i) \\ &= \sum_{i=1}^{q} P(a_i) \log qP(a_i) \end{aligned}$$

Using the log relationship from Problem 8.6, this becomes

$$= \log e \sum_{i=1}^{q} P(a_i) \ln qP(a_i)$$

Then, multiplying the inequality $\ln x \leq x - 1$ by -1 to get $\ln 1/x \geq 1 - x$ and applying it to this last result,

$$\begin{aligned} \log q - H(\mathbf{z}) &\geq \log e \sum_{i=1}^{q} P(a_i) \left[1 - \frac{1}{qP(a_i)}\right] \\ &\geq \log e \left[\sum_{i=1}^{q} P(a_i) - \frac{1}{q} \sum_{i=1}^{q} \frac{P(a_i)}{P(a_i)}\right] \\ &\geq \log e [1 - 1] \\ &\geq 0 \end{aligned}$$

so that

$$\log q \geq H(\mathbf{z})$$

Thus, $H(\mathbf{z})$ is always less than, or equal to, $\log q$. Furthermore, in view of the equality condition $(x = 1)$ for $\ln 1/x \geq 1 - x$, which was introduced at only one point in the above derivation, we will have strict equality if and only if $P(a_i) = 1/q$ for all i.

Problem 8.9

(a) Substituting the given values of p_{bs} and p_e into the binary entropy function derived in the example, the average information or entropy of the source is 0.811 bits/symbol.

(b) The equivocation or average entropy of the source given that the output has been observed (using Eq. 8.3-9) is 0.75 bits/symbol. Thus, the decrease in uncertainty is 0.061 bits/symbol.

(c) It is the mutual information $I(\mathbf{z}, \mathbf{v})$ of the system and is less than the capacity of the channel, which is, in accordance with the equation derived in the example, 0.0817 bits/symbol.

Problem 8.10

(b) Substituting 0.5 into the above equation, the capacity of the erasure channel is 0.5. Substituting 0.125 into the equation for the capacity of a BSC given in Section 8.3.2, we find that its capacity is 0.456. Thus, the binary erasure channel with a higher probability of error has a larger capacity to transfer information.

Problem 8.11

(a) The plot is shown in Fig. P8.11.

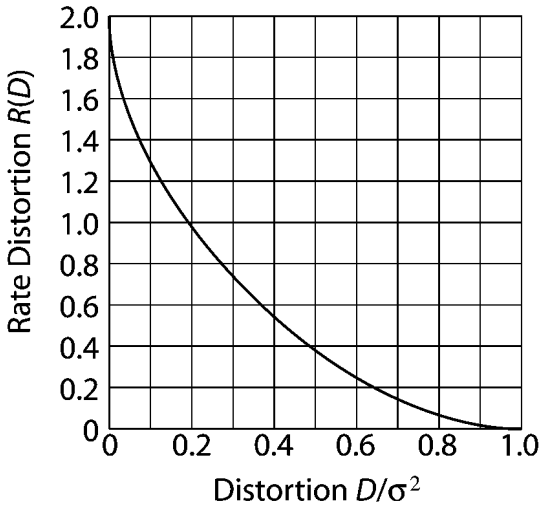


Figure P8.11

Problem 8.14

The arithmetic decoding process is the reverse of the encoding procedure. Start by dividing the [0, 1) interval according to the symbol probabilities. This is shown in Table P8.14.

Table P8.14

Symbol	Probability	Range
a	0.2	[0.0, 0.2)
e	0.3	[0.2, 0.5)
i	0.1	[0.5, 0.6)
o	0.2	[0.6, 0.8)
u	0.1	[0.8, 0.9)
!	0.1	[0.9, 1.0)

The decoder immediately knows the message 0.23355 begins with an "e", since the coded message lies in the interval [0.2, 0.5). This makes it clear that the second symbol is an "a", which narrows the interval to [0.2, 0.26). To further see this, divide the interval [0.2, 0.5) according to the symbol probabilities. Proceeding like this, which is the same procedure used to code the message, we get "eaii!".

Problem 8.16

The input to the LZW decoding algorithm for the example in Example 8.12 is

39 39 126 126 256 258 260 259 257 126

The starting dictionary, to be consistent with the coding itself, contains 512 locations– with the first 256 corresponding to gray level values 0 through 255. The decoding algorithm begins by getting the first encoded value, outputting the corresponding value from the dictionary, and setting the "recognized sequence" to the first value. For each additional encoded value, we (1) output the dictionary entry for the pixel value(s), (2) add a new dictionary entry whose content is the "recognized sequence" plus the first element of the encoded value being processed, and (3) set the "recognized sequence" to the encoded value being processed. For the encoded output in Example 8.12, the sequence of operations is as shown in Table P8.16.

Note, for example, in row 5 of the table that the new dictionary entry for location 259

is 126-39, the concatenation of the currently recognized sequence, 126, and the first element of the encoded value being processed–the 39 from the 39-39 entry in dictionary location 256. The output is then read from the third column of the table to yield

$$\begin{matrix} 39 & 39 & 126 & 126 \\ 39 & 39 & 126 & 126 \\ 39 & 39 & 126 & 126 \\ 39 & 39 & 126 & 126 \end{matrix}$$

where it is assumed that the decoder knows or is given the size of the image that was recieved. Note that the dictionary is generated as the decoding is carried out.

Table P8.16

Recognized	Encoded Value	Pixels	Dict. Address	Dict. Entry
	39	39		
39	39	39	256	39-39
39	126	126	257	39-126
126	126	126	258	126-126
126	256	39-39	259	126-39
256	258	126-126	260	39-39-126
258	260	39-39-126	261	126-126-39
260	259	126-39	262	39-39-126-126
259	257	39-126	263	126-39-39
257	126	126	264	39-126-126

Problem 8.19

(a) The motivation is clear from Fig. 8.17. The transition at c must somehow be tied to a particular transition on the previous line. Note that there is a closer white to black transition on the previous line to the right of c, but how would the decoder know to use it instead of the one to the left. Both are less than ec. The first similar transition past e establishes the convention to make this decision.

Problem 8.21

The derivation proceeds by substituting the uniform probability function into Eqs. (8.5-20) - (8.5-22) and solving the resulting simultaneous equations with $L = 4$. Eq. (8.5-21)

yields

$$s_0 = 0$$
$$s_1 = \tfrac{1}{2}(t_1 + t_2)$$
$$s_2 = \infty$$

Substituting these values into the integrals defined by Eq. (8.5-20), we get two equations. The first is (assuming $s_1 \leq A$)

$$\int_{s_0}^{s_1} (s - t_1)\, p(s)\, ds = 0$$
$$\frac{1}{2A} \int_0^{\frac{1}{2}(t_1+t_2)} (s - t_1)\, ds = \frac{s^2}{2} - t_1 s \Bigg|_0^{\frac{1}{2}(t_1+t_2)} = 0$$
$$(t_1 + t_2)^2 - 4t_1 (t_1 + t_2) = 0$$
$$(t_1 + t_2)(t_2 - 3t_1) = 0$$
$$t_1 = -t_2 \text{ and } t_2 = 3t_1$$

The first of these relations does not make sense since both t_1 and t_2 must be positive. The second relationship is a valid one. The second integral yields (noting that s_1 is less than A so the integral from A to ∞ is 0 by the definition of $p(s)$)

$$\frac{1}{2A} \int_{\frac{1}{2}(t_1+t_2)}^{A} (s - t_2)\, ds = \frac{s^2}{2} - t_2 s \Bigg|_{\frac{1}{2}(t_1+t_2)}^{A} = 0$$
$$4A^2 - 8At_2 - (t_1 + t_2)^2 - 4t_2 (t_1 + t_2) = 0$$

Substituting $t_2 = 3t_1$ from the first integral simplification into this result, we get

$$8t_1^2 - 6At_1 + A^2 = 0$$
$$\left[t_1 - \tfrac{A}{2}\right](8t_1 - 2A) = 0$$
$$t_1 = \tfrac{A}{2} \text{ and } t_1 = \tfrac{A}{4}$$

Back substituting these values of t_1, we find the corresponding t_2 and s_1 values:

$$t_2 = \tfrac{3A}{2} \text{ and } s_1 = A \text{ for } t_1 = \tfrac{A}{2}$$
$$t_2 = \tfrac{3A}{4} \text{ and } s_1 = \tfrac{A}{2} \text{ for } t_1 = \tfrac{A}{4}$$

Since $s_1 = A$ is not a real solution (the second integral equation would then be evaluated from A to A, yielding 0 or no equation), the solution is given by the second. That is,

$$s_0 = 0 \quad s_1 = \tfrac{A}{2} \quad s_2 = \infty$$
$$t_1 = \tfrac{A}{4} \quad t_2 = \tfrac{3A}{4}$$

Problem 8.23

(a) - (b) Following the procedure outlined in Section 8.6.2, we obtain the results shown

in Table P8.23.

Table P8.23

DC Coefficient Difference	Two's Complement Value	Code
-7	1...1001	00000
-6	1...1010	00001
-5	1...1011	00010
-4	1...1100	00011
4	0...0100	00100
5	0...0101	00101
6	0...0110	00110
7	0...0111	00111

Problem 8.27

The appropriate MPEG decoder is shown in Fig. P8.27

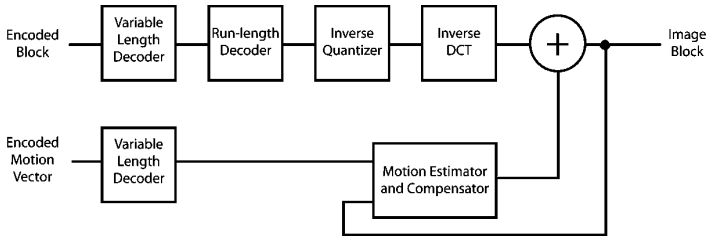


Figure P8.27

9 Solutions (Students)

Problem 9.2

(a) The answer is shown shaded in Fig. P9.2.

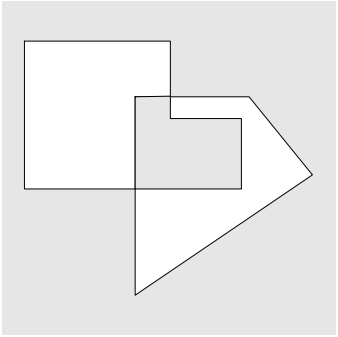


Figure P9.2

Problem 9.3

(a) With reference to the discussion in Section 2.5.2, *m*-connectivity is used to avoid multiple paths that are inherent in 8-connectivity. In one-pixel-thick, fully connected boundaries, these multiple paths manifest themselves in the four basic patterns shown in Fig. P9.3. The solution to the problem is to use the hit-or-miss transform to detect the patterns and then to change the center pixel to 0, thus eliminating the multiple paths. A

basic sequence of morphological steps to accomplish this is as follows:

$$\begin{aligned} X_1 &= A \circledast B^1 \\ Y_1 &= A \cap X_1^c \\ X_2 &= Y_1 \circledast B^2 \\ Y_2 &= Y_1 \cap X_2^c \\ X_3 &= Y_2 \circledast B^3 \\ Y_3 &= Y_2 \cap X_3^c \\ X_4 &= Y_3 \circledast B^4 \\ Y_4 &= Y_3 \cap X_4^c \end{aligned}$$

where A is the input image.

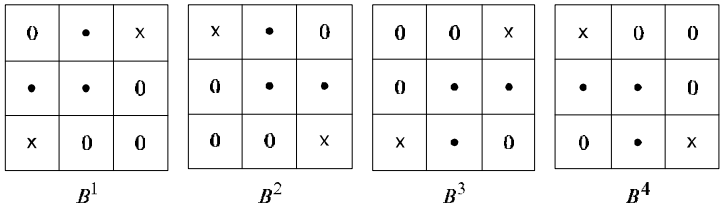


Figure P9.3

Problem 9.5

(a) Erosion is set intersection. The intersection of two convex sets is convex also. See Fig. P9.5 the for solution to part (b). Keep in mind that the digital sets in question are the larger black dots. The lines are shown for convenience in visualizing what the continuous sets would be. In (b) the result of dilation is not convex because the center point is not in the set.

Problem 9.6

Refer to Fig. P9.6. The center of each structuring element is shown as a black dot. Solution (a) was obtained by eroding the original set (shown dashed) with the structuring element shown (note that the origin is at the bottom, right). Solution (b) was obtained by eroding the original set with the tall rectangular structuring element shown. Solution (c) was obtained by first eroding the image shown down to two vertical lines using the rectangular structuring element; this result was then dilated with the circular structuring

element. Solution (d) was obtained by first dilating the original set with the large disk shown. Then dilated image was then eroded with a disk of half the diameter of the disk used for dilation.

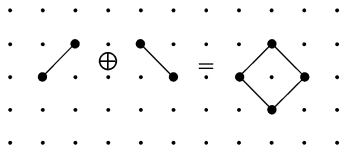


Figure P9.5

Problem 9.8

(a) The dilated image will grow without bound. (b) A one-element set (i.e., a one-pixel image).

Problem 9.10

The approach is to prove that

$$\left\{x \in Z^2 \,\middle|\, (\hat{B})_x \cap A \neq \emptyset\right\} \equiv \left\{x \in Z^2 \,|\, x = a + b \quad \text{for } a \in A \text{ and } b \in B.\right\}$$

The elements of $(\hat{B})_x$ are of the form $x - b$ for $b \in B$. The condition $(\hat{B})_x \cap A \neq \emptyset$ implies that for some $b \in B$, $x - b \in A$, or $x - b = a$ for some $a \in A$ (note in the preceding equation that $x = a + b$). Conversely, if $x = a + b$ for some $a \in A$ and $b \in B$, then $x - b = a$ or $x - b \in A$, which implies that $(\hat{B})_x \cap A \neq \emptyset$.

Problem 9.12

The proof, which consists of proving that

$$\left\{x \in Z^2 \,|\, x + b \in A \text{ , for every } b \in B\right\} \equiv \left\{x \in Z^2 \,|\, (B)_x \subseteq A \;\right\},$$

follows directly from the definition of translation because the set $(B)_x$ has elements of the form $x + b$ for $b \in B$. That is, $x + b \in A$ for *every* $b \in B$ implies that $(B)_x \subseteq A$. Conversely, $(B)_x \subseteq A$ implies that *all* elements of $(B)_x$ are contained in A, or $x + b \in A$ for every $b \in B$.

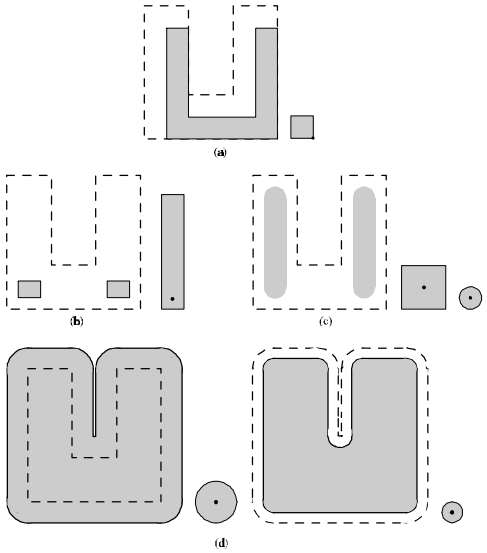


Figure P9.6

Problem 9.14

Starting with the definition of closing,

$$\begin{aligned}(A \bullet B)^c &= [(A \oplus B) \ominus B]^c \\ &= (A \oplus B)^c \oplus \hat{B} \\ &= (A^c \ominus \hat{B}) \oplus \hat{B} \\ &= A^c \circ \hat{B}.\end{aligned}$$

Problem 9.15

(a) Erosion of a set A by B is defined as the set of all values of translates, z, of B such that $(B)_z$ is contained in A. If the origin of B is contained in B, then the set of points describing the erosion is simply all the possible locations of the origin of B such that $(B)_z$ is contained in A. Then it follows from this interpretation (and the definition of erosion) that erosion of A by B is a subset of A. Similarly, dilation of a set C by B is the set of all locations of the origin of $\hat{B}$ such that the intersection of C and $(\hat{B})_z$ is not empty. If the origin of B is contained in B, this implies that C is a subset of the dilation of C by B. Now, from Eq. (9.3-1), we know that $A \circ B = (A \ominus B) \oplus B$. Let C denote the erosion of A by B. It was already established that C is a subset of A. From the

preceding discussion, we know also that C is a subset of the dilation of C by B. But C is a subset of A, so the opening of A by B (the erosion of A by B followed by a dilation of the result) is a subset of A.

Problem 9.18

It was possible to reconstruct the three large squares to their original size because they were not completely eroded and the geometry of the objects and structuring element was the same (i.e., they were squares). This also would have been true if the objects and structuring elements were rectangular. However, a complete reconstruction, for instance, by dilating a rectangle that was partially eroded by a circle, would not be possible.

Problem 9.20

The key difference between the Lake and the other two features is that the former forms a closed contour. Assuming that the shapes are processed one at a time, basic two-step approach for differentiating between the three shapes is as follows:

Step 1. Apply an end-point detector to the object until convergence is achieved. If the result is not the empty set, the object is a Lake. Otherwise it is a Bay or a Line.

Step 2. There are numerous ways to differentiate between a lake and a line. One of the simplest is to determine a line joining the two end points of the object. If the AND of the object and this line contains only two points, the figure is a Bay. Otherwise it is a line segment. There are pathological cases in which this test will fail, and additional ”intelligence” needs to be built into the process, but these pathological cases become less probable with increasing resolution of the thinned figures.

Problem 9.22

(a) With reference to the example shown in Fig. P9.22(a), the boundary that results from using the structuring element in Fig. 9.15(c) generally forms an 8-connected path (leftmost figure), whereas the boundary resulting from the structuring element in Fig. 9.13(b) forms a 4-connected path (rightmost figure).

(b) Using a 3×3 structuring element of all 1's would introduce corner pixels into segments characterized by diagonally-connected pixels. For example, square (2,2) in Fig. 9.15(e) would be a 1 instead of a 0. That value of 1 would carry all the way to the final result in Fig. 9.15(i). There would be other 1's introduced that would turn Fig. 9.15(i) into a much more distorted object.

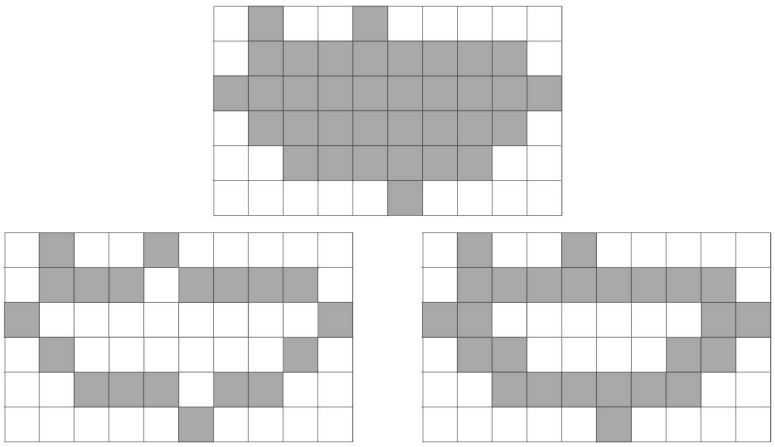


Figure P9.22(a)

PROBLEM 9.24

Denote the original image by A. Create an image of the same size as the original, but consisting of all 0's, call it B. Choose an arbitrary point labeled 1 in A, call it p_1, and apply the algorithm. When the algorithm converges, a connected component has been detected. Label and copy into B the set of all points in A belonging to the connected components just found, set those points to 0 in A and call the modified image A_1. Choose an arbitrary point labeled 1 in A_1, call it p_2, and repeat the procedure just given. If there are K connected components in the original image, this procedure will result in an image consisting of all 0's after K applications of the procedure just given. Image B will contain K labeled connected components.

Problem 9.25

(a) Equation (9.6-1) requires that the (x, y) used in the computation of dilation must satisfy the condition $(x, y) \in D_b$. In terms of the intervals given in the problem statement, this means that x and y must be in the closed interval $x \in [B_{x1}, B_{x2}]$ and

$y \in [B_{y1}, B_{y2}]$. It is required also that $(s-x), (t-y) \in D_f$, which means that $(s-x) \in [F_{x1}, F_{x2}]$ and $(t-y) \in [F_{y1}, F_{y2}]$. Since the valid range of x is the interval $[B_{x1}, B_{x2}]$, the valid range of $(s-x)$ is $[s-B_{x1}, s-B_{x2}]$. But, since x must also satisfy the condition $(s-x) \in [F_{x1}, F_{x2}]$, it follows that $F_{x1} \leq s - B_{x1}$ and $F_{x2} \geq s - B_{x2}$, which finally yields $F_{x1} + B_{x1} \leq s \leq F_{x2} + B_{x2}$. Following the same analysis for t yields $F_{y1} + B_{y1} \leq t \leq F_{y2} + B_{y2}$. Since dilation is a function of (s,t), these two inequalities establish the domain of $(f \oplus b)(s,t)$ in the st-plane.

Problem 9.26

(a) The noise spikes are of the general form shown in Fig. P9.26(a), with other possibilities in between. The amplitude is irrelevant in this case; only the shape of the noise spikes is of interest. To remove these spikes we perform an opening with a cylindrical structuring element of radius greater than $R_{\max}$, as shown in Fig. P9.26(b) (see Fig. 9.30 for an explanation of the process). Note that the shape of the structuring element is matched to the known shape of the noise spikes.

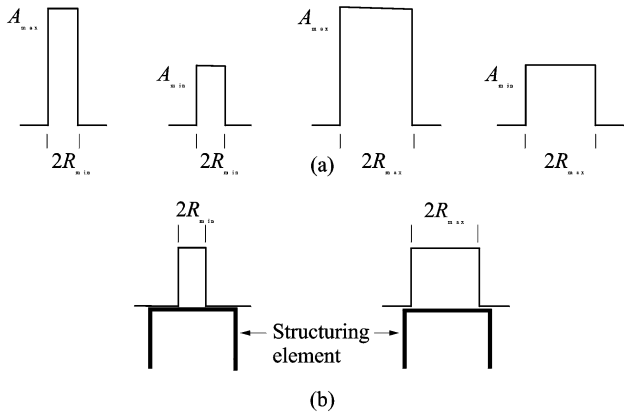


Figure P9.26

Problem 9.27

(a) Color the image border pixels the same color as the particles (white). Call the resulting set of border pixels B. Apply the connected component algorithm. All connected components that contain elements from B are particles that have merged with the border of the image.

10 Solutions (Students)

Problem 10.1

The masks would have the coefficients shown in Fig. P10.1. Each mask would yield a value of 0 when centered on a pixel of an unbroken 3-pixel segment oriented in the direction favored by that mask. Conversely, the response would be a +2 when a mask is centered on a one-pixel gap in a 3-pixel segment oriented in the direction favored by that mask.

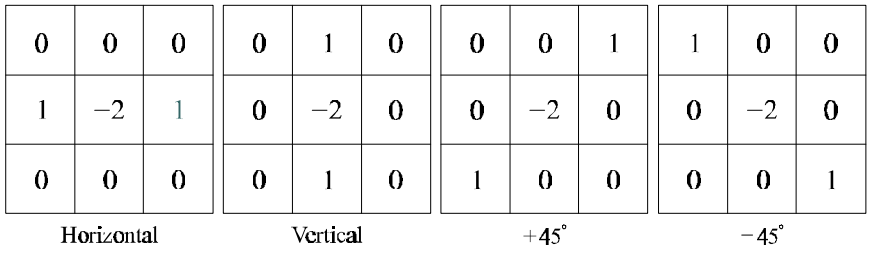


Figure P10.1

Problem 10.3

(a) The lines were thicker than the width of the line detector masks. Thus, when, for example, a mask was centered on the line it ”saw” a constant area and gave a response of 0.

Problem 10.5

The gradient and Laplacian (first and second derivatives) are shown in Fig. P10.5.

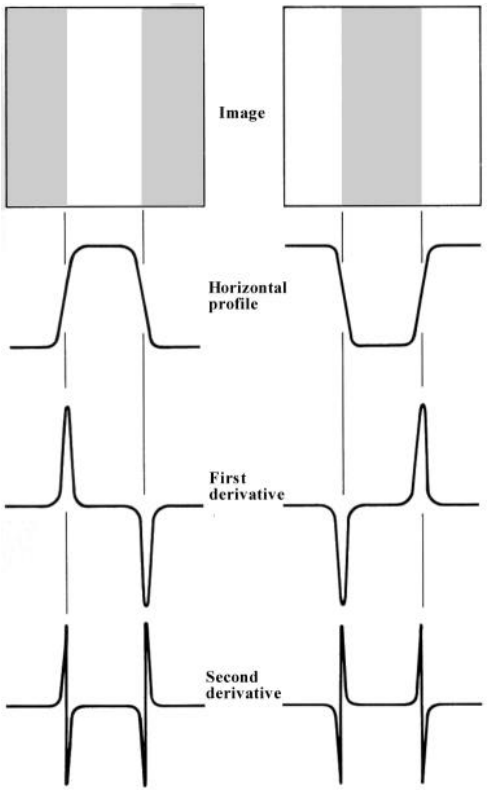


Figure P10.5

Problem 10.7

Consider first the Sobel masks of Figs. 10.8 and 10.9. The easiest way to prove that these masks give isotropic results for edge segments oriented at multiples of 45° is to obtain the mask responses for the four general edge segments shown in Fig. P10.7, which are oriented at increments of 45°. The objective is to show that the responses of the Sobel masks are indistinguishable for these four edges. That this is the case is evident from Table P10.1, which shows the response of each Sobel mask to the four general edge segments. We see that in each case the response of the mask that matches the edge direction is $(4a - 4b)$, and the response of the corresponding orthogonal mask is 0. The response of the remaining two masks is either $(3a - 3b)$ or $(3b - 3a)$. The sign difference is not significant because the gradient is computed by either squaring or taking the absolute value of the mask responses. The same line of reasoning applies to the Prewitt masks.

Table P10.7

Edge direction	Horizontal Sobel (G_x)	Vertical Sobel (G_y)	$+45°$ Sobel (G_{45})	$-45°$ Sobel (G_{-45})
Horizontal	$4a - 4b$	0	$3a - 3b$	$3b - 3a$
Vertical	0	$4a - 4b$	$3a - 3b$	$3a - 3b$
$+45°$	$3a - 3b$	$3a - 3b$	$4a - 4b$	0
$-45°$	$3b - 3a$	$3a - 3b$	0	$4a - 4b$

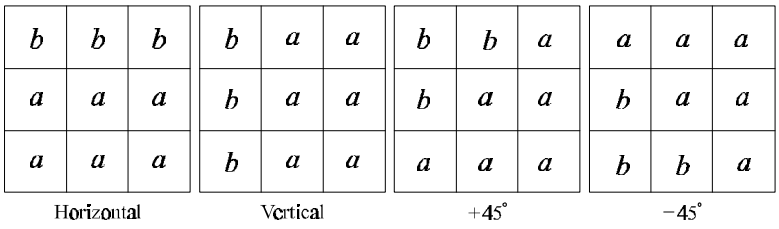


Figure P10.7

Problem 10.9

The solution is as follows (negative numbers are shown underlined):

Edge direction

E NE N NW W SW S SE

Gradient direction

N NW W SW S SE E NE

Compass gradient operators

1 1 1	1 1 0	1 0 $\underline{1}$	0 1 $\underline{1}$	$\underline{1}$ $\underline{1}$ $\underline{1}$	$\underline{1}$ $\underline{1}$ 0	$\underline{1}$ 0 1	0 1 1
0 0 0	1 0 $\underline{1}$	1 0 $\underline{1}$	1 0 $\underline{1}$	0 0 0	$\underline{1}$ 0 1	$\underline{1}$ 0 1	$\underline{1}$ 0 1
$\underline{1}$ $\underline{1}$ $\underline{1}$	0 1 $\underline{1}$	1 0 $\underline{1}$	1 1 0	1 1 1	0 1 1	$\underline{1}$ 0 1	$\underline{1}$ $\underline{1}$ 0

Problem 10.11

(a) With reference to Eq. (10.1-17), we need to prove that

$$\int_{-\infty}^{\infty} \left[\frac{r^2 - \sigma^2}{\sigma^4}\right] e^{-\frac{r^2}{2\sigma^2}} dr = 0.$$

Expanding this equation results in the expression

$$\int_{-\infty}^{\infty}\left[\frac{r^2-\sigma^2}{\sigma^4}\right]e^{-\frac{r^2}{2\sigma^2}}dr = \frac{1}{\sigma^4}\int_{-\infty}^{\infty}r^2e^{-\frac{r^2}{2\sigma^2}}dr - \frac{1}{\sigma^2}\int_{-\infty}^{\infty}e^{-\frac{r^2}{2\sigma^2}}dr.$$

Recall from the definition of the Gaussian density that

$$\frac{1}{\sqrt{2\pi\sigma^2}}\int_{-\infty}^{\infty}e^{-\frac{r^2}{2\sigma^2}}dr = 1$$

and, from the definition of the variance of a Gaussian random variable that

$$\mathrm{Var}(r) = \sigma^2 = \int_{-\infty}^{\infty}r^2e^{-\frac{r^2}{2\sigma^2}}dr.$$

Thus, it follows from the preceding equations that

$$\int_{-\infty}^{\infty}\left[\frac{r^2-\sigma^2}{\sigma^4}\right]e^{-\frac{r^2}{2\sigma^2}}dr = \frac{\sqrt{2\pi\sigma^2}}{\sigma^4}\sigma^2 - \frac{\sqrt{2\pi\sigma^2}}{\sigma^2} = 0.$$

Problem 10.13

(a) Point 1 has coordinates $x = 0$ and $y = 0$. Substituting into Eq. (10.2-3) yields $\rho = 0$, which, in a plot of ρ vs. θ,is a straight line.

(b) Only the origin $(0, 0)$ would yield this result.

(c) At $\theta = +90°$, it follows from Eq. (10.2-3) that $x \cdot (0) + y \cdot (1) = \rho$, or $y = \rho$. At $\theta = -90°$, $x \cdot (0) + y \cdot (-1) = \rho$, or $-y = \rho$. Thus the reflective adjacency.

Problem 10.16

(a) The paths are shown in Fig. P10.16. These paths are as follows:

$$1 : (1,1)(1,2) \rightarrow (2,1)(2,2) \rightarrow (3,1)(3,2)$$
$$2 : (1,1)(1,2) \rightarrow (2,1)(2,2) \rightarrow (3,2)(2,2) \rightarrow (3,2)(3,3)$$

$3 : (1,1)(1,2) \rightarrow (2,2)(1,2) \rightarrow (2,2)(2,3) \rightarrow (3,2)(3,3)$
$4 : (1,1)(1,2) \rightarrow (2,2)(1,2) \rightarrow (2,2)(2,3) \rightarrow (2,2)(3,2) \rightarrow (3,1)(3,2)$
$5 : (1,2)(1,3) \rightarrow (2,2)(2,3) \rightarrow (3,2)(3,3)$
$6 : (1,2)(1,3) \rightarrow (2,2)(2,3) \rightarrow (2,2)(3,2) \rightarrow (3,1)(3,2)$
$7 : (1,2)(1,3) \rightarrow (1,2)(2,2) \rightarrow (2,1)(2,2) \rightarrow (3,1)(3,2)$
$8 : (1,2)(1,3) \rightarrow (1,2)(2,2) \rightarrow (2,1)(2,2) \rightarrow (3,2)(2,2) \rightarrow (3,2)(3,3)$

(b) From Fig. 10.24 and (a), we see that the optimum path is path 6. Its cost is $c = 2 + 0 + 1 + 1 = 4$.

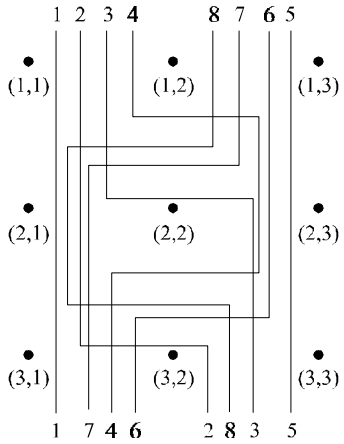


Figure P10.16

Problem 10.18

(a) The number of boundary points between black and white regions is much larger in the image on the right. When the images are blurred, the boundary points will give rise to a larger number of different values for the image on the right, so the histograms of the two blurred images will be different.

(b) To handle border effects, we surround the image with a border of 0's. We assume that the image is of size $N \times N$ (the fact that the image is square is evident from the right image in the problem statement). Blurring is implemented by a 3×3 mask whose coefficients are $1/9$. Figure P10.18 shows the different types of values that the blurred left image (see problem statement) will have. These values are summarized in Table P10.18-1. It is easily verified that the sum of the numbers on the left column of the table is N^2.

Table P10.18-1

No. of Points	Value
$N\left(\frac{N}{2}-1\right)$	0
2	2/9
$N-2$	3/9
4	4/9
$3N-8$	6/9
$(N-2)\left(\frac{N}{2}-2\right)$	1

A histogram is easily constructed from the entries in this table. A similar (tedious, but not difficult) procedure yields the results shown in Table P10.18-2 for the checkerboard image.

Table P10.18-2

No. of Points	Value
$\frac{N^2}{2}-14N+98$	0
28	2/9
$14N-224$	3/9
128	4/9
98	5/9
$16N-256$	6/9
$\frac{N^2}{2}-16N+128$	1

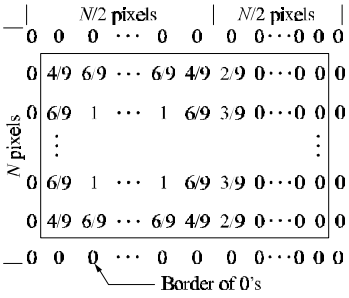


Figure P10.18

Problem 10.20

(a) $A_1 = A_2$ and $\sigma_1 = \sigma_2 = \sigma$, which makes the two modes identical. If the number of samples is not large, convergence to a value at or near the mid point between the two

means also requires that a clear valley exist between the two modes. We can guarantee this by assuming that $\sigma << (m_1 + m_2)/2$.

(b) That this condition cannot happen if $A2 \neq 0$. This is easily established by starting the algorithm with an initial value less than m_1. Even if the right mode associated with m_2 is much smaller in size (e.g., $A1 >> A2$ and $\sigma_1 >> \sigma_2$) the average value of the region to the left of the starting threshold will be smaller than the average of the region to the right because the modes are symmetrical about their mean, and the mode associated with m_2 will bias the data to the right. Thus, the next iterative step will bring the value of the threshold closer to m_1, and eventually to the right of it. This analysis assumes that enough points are available in order to avoid pathological cases in which the algorithm can get "stuck" due to insufficient data that truly represents the shapes assumed in the problem statement.

Problem 10.22

From the figure in the problem statement,

$$p_1(z) = \begin{cases} 0 & z < 1 \\ \frac{1}{2}z - \frac{1}{2} & 1 \leq z \leq 3 \\ 0 & z > 3 \end{cases}$$

and

$$p_2(z) = \begin{cases} 0 & z < 0 \\ -\frac{1}{2}z + 1 & 0 \leq z \leq 2 \\ 0 & z > 2 \end{cases} .$$

The optimum threshold is the value $z = T$ for which $P_1 p_1(T) = P_2 p_2(T)$. In this case $P_1 = P_2$, so

$$\frac{1}{2}T - \frac{1}{2} = -\frac{1}{2}T + 1$$

from which we get $T = 1.5$.

Problem 10.24

From Eq. (10.3-10),

$$P_1 p_1(T) = P_2 p_2(T).$$

Taking the ln of both sides yields

$$\ln P_1 + \ln p_1(T) = \ln P_2 + \ln p_2(T).$$

But

$$p_1(T) = \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(T-\mu_1)^2}{2\sigma_1^2}}$$

and

$$p_2(T) = \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(T-\mu_2)^2}{2\sigma_2^2}}$$

so it follows that

$$\ln P_1 + \ln \frac{1}{\sqrt{2\pi}\sigma_1} - \frac{(T-\mu_1)^2}{2\sigma_1^2} = \ln P_2 + \ln \frac{1}{\sqrt{2\pi}\sigma_2} - \frac{(T-\mu_2)^2}{2\sigma_2^2}$$

$$\ln P_1 - \ln \sigma_1 - \frac{(T-\mu_1)^2}{2\sigma_1^2} - \ln P2 + \ln \sigma_2 + \frac{(T-\mu_2)^2}{2\sigma_2^2} = 0$$

$$\ln \frac{P_1}{P_2} + \ln \frac{\sigma_1}{\sigma_2} - \frac{1}{2\sigma_1^2}(T^2 - 2\mu_1 T + \mu_1^2) + \frac{1}{2\sigma_2^2}(T^2 - 2\mu_2 T + \mu_2^2) = 0$$

$$\ln \frac{\sigma_2 P_1}{\sigma_1 P_2} + T^2\left(\frac{1}{2\sigma_2^2} - \frac{1}{2\sigma_1^2}\right) + T\left(\frac{\mu_1}{\sigma_1^2} - \frac{\mu_2}{\sigma_2^2}\right) + \left(\frac{\mu_2^2}{2\sigma_2^2} - \frac{\mu_1^2}{2\sigma_1^2}\right) = 0.$$

From this expression we get

$$AT^2 + BT + C = 0$$

with

$$\begin{aligned} A &= (\sigma_1^2 - \sigma_2^2) \\ B &= 2(\sigma_2^2\mu_1 - \sigma_1^2\mu_2) \end{aligned}$$

and

$$C = \sigma_1^2\mu_2^2 - \sigma_2^2\mu_1^2 + 2\sigma_1^2\sigma_2^2 \ln \frac{\sigma_2 P_1}{\sigma_1 P_2}.$$

Problem 10.26

The simplest solution is to use the given means and standard deviations to form two Gaussian probability density functions, and then to use the optimum thresholding approach discussed in Section 10.3.5 (in particular, see Eqs. (10.3-11) through (10.3-13). The probabilities P_1 and P_2 can be estimated by visual analysis of the images (i.e., by determining the relative areas of the image occupied by objects and background). It is clear by looking at the image that the probability of occurrence of object points is less than that of background points. Alternatively, an automatic estimate can be obtained by thresholding the image into points with values greater than 200 and less than 110 (see problem statement). Using the given parameters, the results would be good estimates of the relative probability of occurrence of object and background points due to the separation between means, and the relatively tight standard deviations. A more sophisticated approach is to use the Chow-Kaneko procedure discussed in Section 10.3.5.

Problem 10.29

(a) The elements of $T[n]$ are the coordinates of points in the image below the plane $g(x, y) = n$, where n is an integer that represents a given step in the execution of the algorithm. Since n never decreases, the set of elements in $T[n-1]$ is a subset of the elements in $T[n]$. In addition, we note that all the points below the plane $g(x, y) = n - 1$ are also below the plane $g(x, y) = n$, so the elements of $T[n]$ are never replaced. Similarly, $C_n(M_i)$ is formed by the intersection of $C(M_i)$ and $T[n]$, where $C(M_i)$ (whose elements never change) is the set of coordinates of *all* points in the catchment basin associated with regional minimum M_i. Since the elements of $C(M_i)$ never change, and the elements of $T[n]$ are never replaced, it follows that the elements in $C_n(M_i)$ are never replaced either. In addition, we see that $C_{n-1}(M_i) \subseteq C_n(M_i)$.

Problem 10.31

The first step in the application of the watershed segmentation algorithm is to build a dam of height $max + 1$ to prevent the rising water from running off the ends of the function, as shown in Fig. P10.31(b). For an image function we would build a box of height $max + 1$ around its border. The algorithm is initialized by setting $C[1] = T[1]$. In this case, $T[1] = \{g(2)\}$, as shown in Fig. P10.31(c) (note the water level). There is only one connected component in this case: $Q[1] = \{q_1\} = \{g(2)\}$.

Next, we let $n = 2$ and, as shown in Fig. P10.31(d), $T[2] = \{g(2), g(14)\}$ and $Q[2] = \{q_1; q_2\}$, where, for clarity, different connected components are separated by semicolons. We start construction of $C[2]$ by considering each connected component in $Q[2]$. When $q = q_1$, the term $q \cap C[1]$ is equal to $\{g(2)\}$, so condition 2 is satisfied and, therefore, $C[2] = \{g(2)\}$. When $q = q_2$, $q \cap C[1] = \emptyset$ (the empty set) so condition 1 is satisfied and we incorporate q in $C[2]$, which then becomes $C[2] = \{g(2); g(14)\}$ where, as above, different connected components are separated by semicolons.

When $n = 3$ [Fig. P10.31(e)], $T[3] = \{2, 3, 10, 11, 13, 14\}$ and $Q[3] = \{q_1; q_2; q_3\} = \{2, 3; 10, 11; 13, 14\}$ where, in order to simplify the notation we let k denote $g(k)$. Proceeding as above, $q_1 \cap C[2] = \{2\}$ satisfies condition 2, so q_1 is incorporated into the new set to yield $C[3] = \{2, 3; 14\}$. Similarly, $q_2 \cap C[2] = \emptyset$ satisfies condition 1 and $C[3] = \{2, 3; 10, 11; 14\}$. Finally, $q_3 \cap C[2] = \{14\}$ satisfies condition 2 and $C[3] = \{2, 3; 10, 11; 13, 14\}$. It is easily verified that $C[4] = C[3] = \{2, 3; 10, 11; 13, 14\}$.

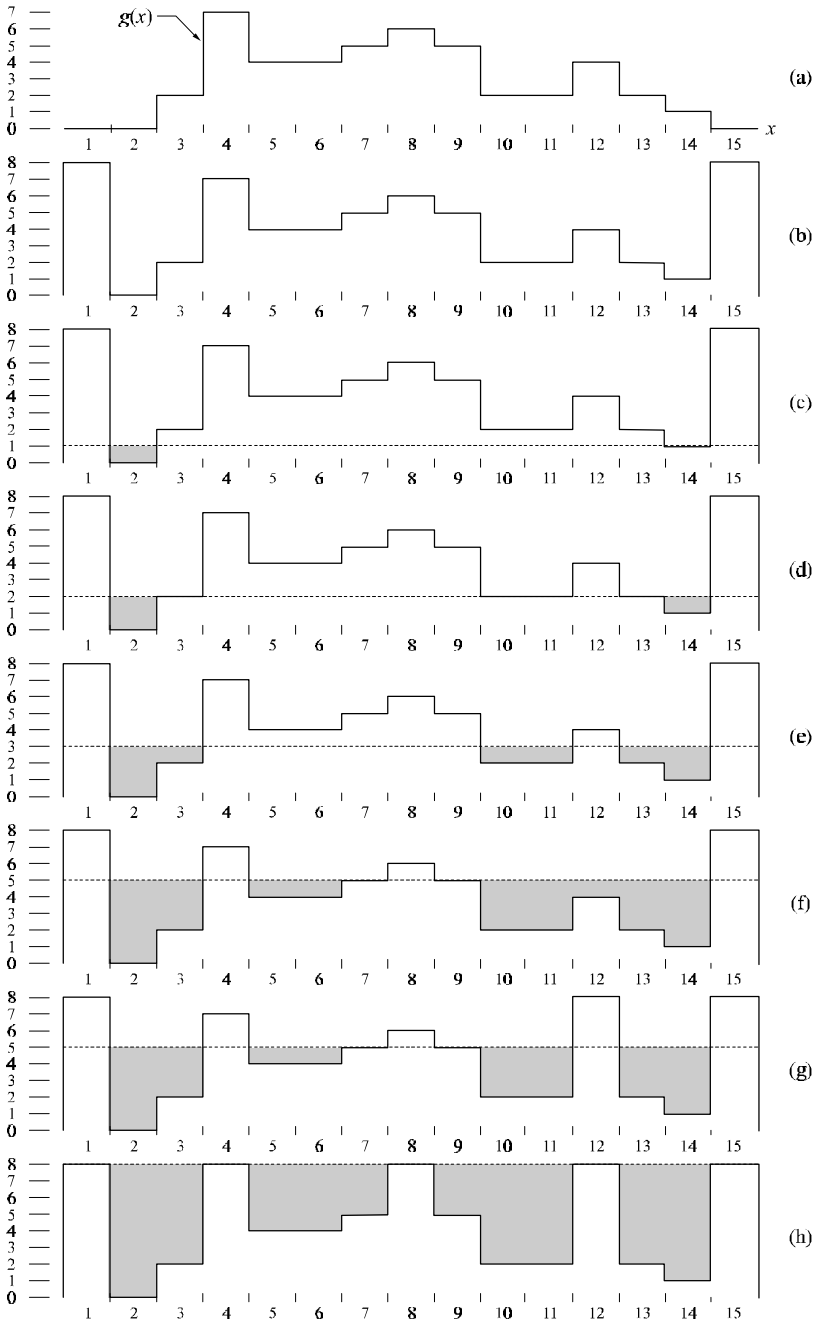


Figure P10.31

When $n = 5$ [Fig. P10.31(f)], we have, $T[5] = \{2, 3, 5, 6, 10, 11, 12, 13, 14\}$ and $Q[5] = \{q_1; q_2; q_3\} = \{2, 3; 5, 6; 10, 11, 12, 13, 14\}$ (note the merging of two previously

distinct connected components). Is is easily verified that $q_1 \cap C[4]$ satisfies condition 2 and that $q_2 \cap C[4]$ satisfied condition 1. Proceeding with these two connected components exactly as above yields $C[5] = \{2, 3; 5, 6; 10, 11; 13, 14\}$ up to this point. Things get more interesting when we consider q_3. Now, $q_3 \cap C[4] = \{10, 11; 13, 14\}$ which, since it contains two connected components of $C[4]$ satisfies condition 3. As mentioned previously, this is an indication that water from two different basins has merged and a dam must be built to prevent this. Dam building is nothing more than separating q_3 into the two original connected components. In this particular case, this is accomplished by the dam shown in Fig. P10.31(g), so that now $q_3 = \{q_{31}; q_{32}\} = \{10, 11; 13, 14\}$. Then, $q_{31} \cap C[4]$ and $q_{32} \cap C[4]$ each satisfy condition 2 and we have the final result for $n = 5$, $C[5] = \{2, 3; 5, 6; 10, 11; 13; 14\}$.

Continuing in the manner just explained yields the final segmentation result shown in Fig. P10.31(h), where the ”edges” are visible (from the top) just above the water line. A final post-processing step would remove the outer dam walls to yield the inner edges of interest.

11 Solutions (Students)

Problem 11.1

(a) The key to this problem is to recognize that the value of every element in a chain code is relative to the value of its predecessor. The code for a boundary that is traced in a consistent manner (e.g., clockwise) is a unique circular set of numbers. Starting at different locations in this set does not change the structure of the circular sequence. Selecting the smallest integer as the starting point simply identifies the same point in the sequence. Even if the starting point is not unique, this method would still give a unique sequence. For example, the sequence 101010 has three possible starting points, but they all yield the same smallest integer 010101.

Problem 11.3

(a) The rubber-band approach forces the polygon to have vertices at every inflection of the cell wall. That is, the locations of the vertices are fixed by the structure of the inner and outer walls. Since the vertices are joined by straight lines, this produces the minimum-perimeter polygon for any given wall configuration.

Problem 11.4

(a) The resulting polygon would contain all the boundary pixels.

Problem 11.5

(a) The solution is shown in Fig. P11.5(b).

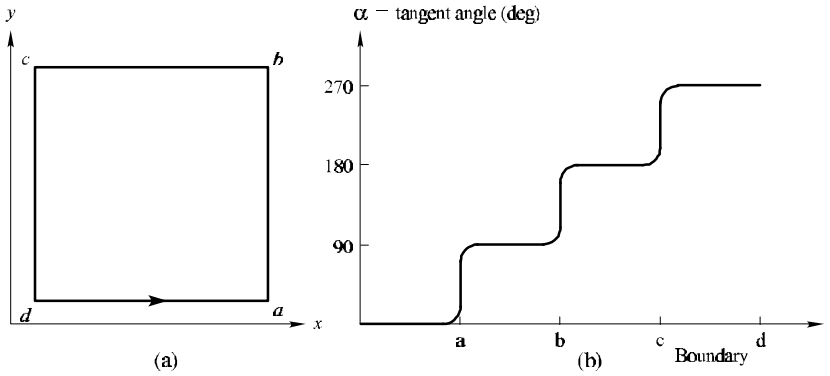


Figure P11.5

Problem 11.6

(a) From Fig. P11.6(a), we see that the distance from the origin to the triangle is given by

$$
\begin{aligned}
r(\theta) &= \frac{D_0}{\cos\theta} \qquad 0^\circ \le \theta < 60^\circ \\
&= \frac{D_0}{\cos(120^\circ - \theta)} \qquad 60^\circ \le \theta < 120^\circ \\
&= \frac{D_0}{\cos(180^\circ - \theta)} \qquad 120^\circ \le \theta < 180^\circ \\
&= \frac{D_0}{\cos(240^\circ - \theta)} \qquad 180^\circ \le \theta < 240^\circ \\
&= \frac{D_0}{\cos(300^\circ - \theta)} \qquad 240^\circ \le \theta < 300^\circ \\
&= \frac{D_0}{\cos(360^\circ - \theta)} \qquad 300^\circ \le \theta < 360^\circ
\end{aligned}
$$

where D_0 is the perpendicular distance from the origin to one of the sides of the triangle, and $D = D_0/\cos(60^\circ) = 2D_0$. Once the coordinates of the vertices of the triangle are given, determining the equation of each straight line is a simple problem, and D_0 (which is the same for the three straight lines) follows from elementary geometry. The signature is shown in Fig. P11.6(b).

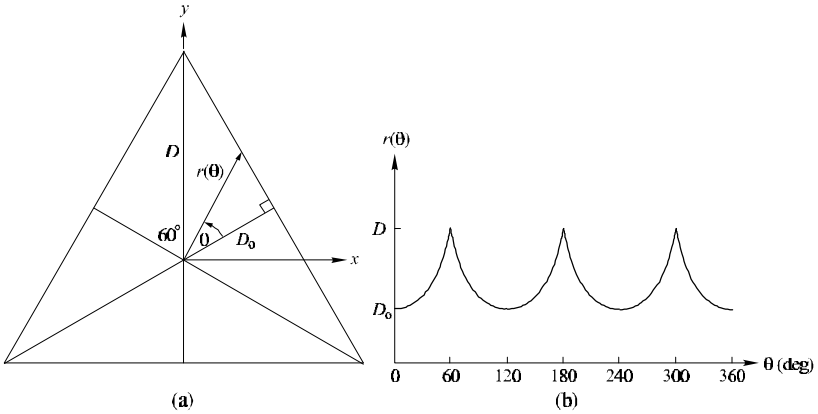


Figure P11.6

Problem 11.7

The solutions are shown in Fig. P11.7.

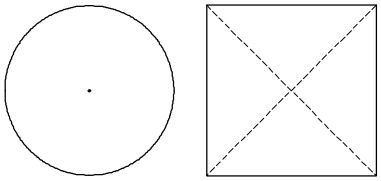


Figure P11.7

Problem 11.8

(a) In the first case, $N(p) = 5$, $S(p) = 1$, $p_2 \cdot p_4 \cdot p_6 = 0$, and $p4 \cdot p6 \cdot p8 = 0$, so Eq. (11.1-1) is satisfied and p is flagged for deletion. In the second case, $N(p) = 1$, so Eq. (11.1-1) is violated and p is left unchanged. In the third case $p2 \cdot p4 \cdot p6 = 1$ and $p4 \cdot p6 \cdot p8 = 1$, so conditions (c) and (d) of Eq. (11.1-1) are violated and p is left unchanged. In the forth case $S(p) = 2$, so condition (b) is violated and p is left unchanged.

Problem 11.9

(a) The result is shown in Fig. 11.9(b).

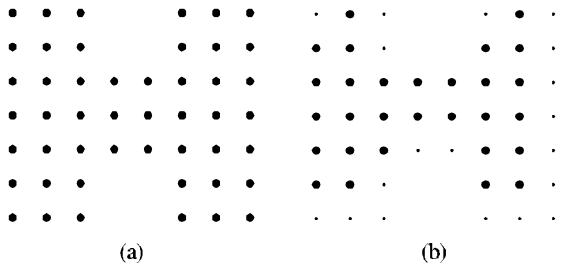


Figure P11.9

Problem 11.10

(a) The number of symbols in the first difference is equal to the number of segment primitives in the boundary, so the shape order is 12.

Problem 11.12

The mean is sufficient.

Problem 11.14

This problem can be solved by using two descriptors: holes and the convex deficiency (see Section 9.5.4 regarding the convex hull and convex deficiency of a set). The decision making process can be summarized in the form of a simple decision, as follows: If the character has two holes, it is an 8. If it has one hole it is a 0 or a 9. Otherwise, it is a 1 or an X. To differentiate between 0 and 9 we compute the convex deficiently. The presence of a ”significant” deficiency (say, having an area greater than 20% of the area of a rectangle that encloses the character) signifies a 9; otherwise we classify the character as a 0. We follow a similar procedure to separate a 1 from an X. The presence of a convex deficiency with four components whose centroids are located approximately in the North, East, West, and East quadrants of the character indicates that the character is an X. Otherwise we say that the character is a 1. This is the basic approach. Implementation of this technique in a real character recognition environment has to take into account other factors such as multiple ”small” components in the convex deficiency due to noise, differences in orientation, open loops, and the like. However, the material in Chapters 3, 9 and 11 provide a solid base from which to formulate solutions.

Problem 11.16

(a) The image is

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

Let $z_1 = 0$ and $z_2 = 1$. Since there are only two gray levels the matrix $\mathbf{A}$ is of order 2×2. Element a_{11} is the number of pixels valued 0 located one pixel to the right of a 0. By inspection, $a_{11} = 0$. Similarly, $a_{12} = 10$, $a_{21} = 10$, and $a_{22} = 0$. The total number of pixels satisfying the predicate P is 20, so

$$\mathbf{C} = \begin{bmatrix} 0 & 1/2 \\ 1/2 & 0 \end{bmatrix}.$$

Problem 11.18

The mean square error, given by Eq. (11.4-12), is the sum of the eigenvalues whose corresponding eigenvectors are not used in the transformation. In this particular case, the four smallest eigenvalues are applicable (see Table 11.5), so the mean square error is

$$e_{ms} = \sum_{j=3}^{6} \lambda_j = 280.$$

The maximum error occurs when $K = 0$ in Eq. (11.4-12) which then is the sum of all the eigenvalues, or 4421 in this case. Thus, the error incurred by using the two eigenvectors corresponding to the largest eigenvalues is only 6.3 % of the total possible error.

Problem 11.20

When the boundary is symmetric about the both the major and minor axes and both axes intersect at the centroid of the boundary.

Problem 11.22

We can compute a measure of texture using the expression

$$R(x, y) = 1 - \frac{1}{1 + \sigma^2(x, y)}$$

where $\sigma^2(x, y)$ is the gray-level variance computed in a neighborhood of (x, y). The size of the neighborhood must be sufficiently large so as to contain enough samples to have a stable estimate of the mean and variance. Neighborhoods of size 7×7 or 9×9 generally are appropriate for a low-noise case such as this.

Since the variance of normal wafers is known to be 400, we can obtain a normal value for $R(x, y)$ by using $\sigma^2 = 400$ in the above equation. An abnormal region will have a variance of about $(50)^2 = 2,500$ or higher, yielding a larger value of $R(x, y)$. The procedure then is to compute $R(x, y)$ at every point (x, y) and label that point as 0 if it is normal and 1 if it is not. At the end of this procedure we look for clusters of 1's using, for example, connected components (see Section 9.5.3 regarding computation of connected components) . If the area (number of pixels) of any connected component exceeds 400 pixels, then we classify the sample as defective.

12 Solutions (Students)

Problem 12.2

From the definition of the Euclidean distance,

$$D_j(\mathbf{x}) = \|\mathbf{x} - \mathbf{m}_j\| = \left[(\mathbf{x} - \mathbf{m}_j)^T(\mathbf{x} - \mathbf{m}_j)\right]^{1/2}$$

Since $D_j(\mathbf{x})$ is non-negative, choosing the smallest $D_j(\mathbf{x})$ is the same as choosing the smallest $D_j^2(\mathbf{x})$, where

$$\begin{aligned} D_j^2(\mathbf{x}) &= \|\mathbf{x} - \mathbf{m}_j\|^2 = (\mathbf{x} - \mathbf{m}_j)^T(\mathbf{x} - \mathbf{m}_j) \\ &= \mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{m}_j + \mathbf{m}_j^T\mathbf{m}_j \\ &= \mathbf{x}^T\mathbf{x} - \mathbf{2}\left(\mathbf{x}^T\mathbf{m}_j - \frac{1}{2}\mathbf{m}_j^T\mathbf{m}_j\right) \end{aligned}$$

We note that the term $\mathbf{x}^T\mathbf{x}$ is independent of j (that is, it is a constant with respect to j in $D_j^2(\mathbf{x})$, $j = 1, 2, ...$). Thus, choosing the minimum of $D_j^2(\mathbf{x})$ is equivalent to choosing the maximum of $\left(\mathbf{x}^T\mathbf{m}_j - \frac{1}{2}\mathbf{m}_j^T\mathbf{m}_j\right)$.

Problem 12.4

The solution is shown in Fig. P12.4, where the x's are treated as voltages and the Y's denote impedances. From basic circuit theory, the currents, I's, are the products of the voltages times the impedances.

Problem 12.6

The solution to the first part of this problem is based on being able to extract connected components (see Chapters 2 and 11) and then determining whether a connected component is convex or not (see Chapter 11). Once all connected components have been extracted we perform a convexity check on each and reject the ones that are not convex. All that is left after this is to determine if the remaining blobs are complete or incomplete. To do this, the region consisting of the extreme rows and columns of the image is

declared a region of 1's. Then if the pixel-by-pixel AND of this region with a particular blob yields at least one result that is a 1, it follows that the actual boundary touches that blob, and the blob is called incomplete. When only a single pixel in a blob yields an AND of 1 we have a marginal result in which only one pixel in a blob touches the boundary. We can arbitrarily declare the blob incomplete or not. From the point of view of implementation, it is much simpler to have a procedure that calls a blob incomplete whenever the AND operation yields one or more results valued 1.

After the blobs have been screened using the method just discussed, they need to be classified into one of the three classes given in the problem statement. We perform the classification problem based on vectors of the form $\mathbf{x} = (x_1, x_2)^T$, where x_1 and x_2 are, respectively, the lengths of the major and minor axis of an elliptical blob, the only type left after screening. Alternatively, we could use the eigen axes for the same purpose. (See Section 11.2.1 on obtaining the major axes or the end of Section 11.4 regarding the eigen axes.) The mean vector of each class needed to implement a minimum distance classifier is really given in the problem statement as the average length of each of the two axes for each class of blob. If' they were not given, they could be obtained by measuring the length of the axes for complete ellipses that have been classified a priori as belonging to each of the three classes. The given set of ellipses would thus constitute a training set, and learning would simply consist of computing the principal axes for all ellipses of one class and then obtaining the average. This would be repeated for each class. A block diagram outlining the solution to this problem is straightforward.

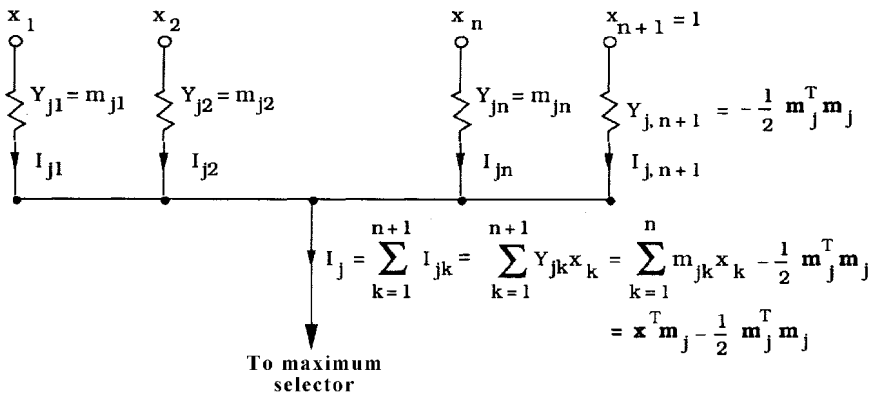


Figure P12.4

Problem 12.8

(a) As in Problem 12.7,

$$\mathbf{m}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathbf{m}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathbf{C}_1 = \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \quad \mathbf{C}_1^{-1} = 2\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \quad |\mathbf{C}_1| = 0.25$$

and

$$\mathbf{C}_2 = 2\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \quad \mathbf{C}_2^{-1} = \frac{1}{2}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; \quad |\mathbf{C}_2| = 4.00$$

Since the covariance matrices are not equal, it follows from Eq. (12.2-26) that

$$\begin{aligned} d_1(\mathbf{x}) &= -\frac{1}{2}\ln(0.25) - \frac{1}{2}\left\{\mathbf{x}^T\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}\mathbf{x}\right\} \\ &= -\frac{1}{2}\ln(0.25) - (x_1^2 + x_2^2) \end{aligned}$$

and

$$\begin{aligned} d_2(\mathbf{x}) &= -\frac{1}{2}\ln(4.00) - \frac{1}{2}\left\{\mathbf{x}^T\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}\mathbf{x}\right\} \\ &= -\frac{1}{2}\ln(4.00) - \frac{1}{4}(x_1^2 + x_2^2) \end{aligned}$$

where the term ln $P(\omega_j)$ was not included because it is the same for both decision functions in this case. The equation of the Bayes decision boundary is

$$d(\mathbf{x}) = d_1(\mathbf{x}) - d_2(\mathbf{x}) = 1.39 - \frac{3}{4}(x_1^2 + x_2^2) = 0.$$

(b) A plot of the boundary is shown in Fig. P12.8.

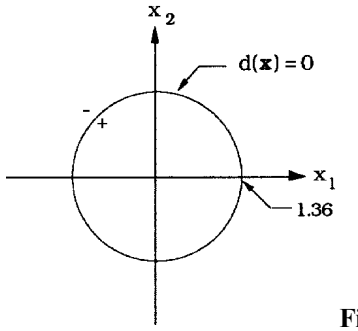


Figure P12.8

Problem 12.10

From basic probability theory,

$$p(c) = \sum_{\mathbf{x}} p(c/\mathbf{x})p(\mathbf{x}).$$

For any pattern belonging to class ω_j, $p(c/\mathbf{x}) = p(\omega_j/\mathbf{x})$. Therefore,

$$p(c) = \sum_{\mathbf{x}} p(\omega_j/\mathbf{x})p(\mathbf{x}).$$

Substituting into this equation the formula $p(\omega_j/\mathbf{x}) = p(\mathbf{x}/\omega_j)p(\omega_j)/p(\mathbf{x})$ gives

$$p(c) = \sum_{\mathbf{x}} p(\mathbf{x}/\omega_j)p(\omega_j).$$

Since the argument of the summation is positive, $p(c)$ is maximized by maximizing $p(\mathbf{x}/\omega_j)p(\omega_j)$ for each j. That is, if for each $\mathbf{x}$ we compute $p(\mathbf{x}/\omega_j)p(\omega_j)$ for $j = 1, 2, ..., W$, and use the largest value each time as the basis for selecting the class from which $\mathbf{x}$ came, then $p(c)$ will be maximized. Since $p(e) = 1 - p(c)$, the probability of error is minimized by this procedure.

Problem 12.12

We start by taking the partial derivative of J with respect to $\mathbf{w}$:

$$\frac{\partial J}{\partial \mathbf{w}} = \frac{1}{2}\left[\mathbf{y}\mathrm{sgn}(\mathbf{w}^T\mathbf{y}) - \mathbf{y}\right]$$

where, by definition, $\mathrm{sgn}(\mathbf{w}^T\mathbf{y}) = 1$ if $\mathbf{w}^T\mathbf{y} > 0$, and $\mathrm{sgn}(\mathbf{w}^T\mathbf{y}) = -1$ otherwise. Substituting the partial derivative into the general expression given in the problem statement gives

$$\mathbf{w}(k+1) = \mathbf{w}(k) + \frac{c}{2}\left\{\mathbf{y}(\mathbf{k}) - \mathbf{y}(\mathbf{k})\mathrm{sgn}\left[\mathbf{w}(\mathbf{k})^T\mathbf{y}(\mathbf{k})\right]\right\}$$

where $\mathbf{y}(k)$ is the training pattern being considered at the kth iterative step. Substituting the definition of the sgn function into this result yields

$$\mathbf{w}(k+1) = \mathbf{w}(k) + c\begin{cases} \mathbf{0} & \text{if } \mathbf{w}(\mathbf{k})^T\mathbf{y}(\mathbf{k}) \\ \mathbf{y}(k) & \text{otherwise} \end{cases}$$

where $c > 0$ and $\mathbf{w}(1)$ is arbitrary. This expression agrees with the formulation given in the problem statement.

Problem 12.14

The single decision function that implements a minimum distance classifier for two

classes is of the form

$$d_{ij}(\mathbf{x}) = \mathbf{x}^T(\mathbf{m}_i - \mathbf{m}_j) - \frac{1}{2}(\mathbf{m}_i^T\mathbf{m}_i - \mathbf{m}_j^T\mathbf{m}_j).$$

Thus, for a particular pattern vector $\mathbf{x}$, when $d_{ij}(\mathbf{x}) > 0$, $\mathbf{x}$ is assigned to class ω_1 and, when $d_{ij}(\mathbf{x}) < 0$, $\mathbf{x}$ is assigned to class ω_2. Values of $\mathbf{x}$ for which $d_{ij}(\mathbf{x}) = 0$ are on the boundary (hyperplane) separating the two classes. By letting $\mathbf{w} = (\mathbf{m}_i - \mathbf{m}_j)$ and $w_{n+1} = -\frac{1}{2}(\mathbf{m}_i^T\mathbf{m}_i - \mathbf{m}_j^T\mathbf{m}_j)$, we can express the above decision function in the form

$$d(\mathbf{x}) = \mathbf{w}^T\mathbf{x} - w_{n+1}.$$

This is recognized as a linear decision function in n dimensions, which is implemented by a single layer neural network with coefficients

$$w_k = (m_{ik} - m_{jk}) \qquad k = 1, 2, \ldots, n$$

and

$$\theta = w_{n+1} = -\frac{1}{2}(\mathbf{m}_i^T\mathbf{m}_i - \mathbf{m}_j^T\mathbf{m}_j).$$

Problem 12.16

(a) When $P(\omega_i) = P(\omega_j)$ and $\mathbf{C} = \mathbf{I}$.

(b) No. The minimum distance classifier implements a decision function that is the perpendicular bisector of the line joining the two means. If the probability densities are known, the Bayes classifier is guaranteed to implement an optimum decision function in the minimum average loss sense. The generalized delta rule for training a neural network says nothing about these two criteria, so it cannot be expected to yield the decision functions in Problems 12.14 or 12.15.

Problem 12.18

All that is needed is to generate for each class training vectors of the form $\mathbf{x} = (x_1, x_2)^T$, where x_1 is the length of the major axis and x_2 is the length of the minor axis of the blobs comprising the training set. These vectors would then be used to train a neural network using, for example, the generalized delta rule. (Since the patterns are in 2D, it is useful to point out to students that the neural network could be designed by inspection in the sense that the classes could be plotted, the decision boundary of minimum complexity obtained, and then its coefficients used to specify the neural network. In this case the classes are far apart with respect to their spread, so most likely a single layer network implementing a linear decision function could do the job.)

Problem 12.20

The first part of Eq. (12.3-3) is proved by noting that the degree of similarity, k, is non-negative, so $D(A,B) = 1/k \geq 0$. Similarly, the second part follows from the fact that k is infinite when (and only when) the shapes are identical.

To prove the third part we use the definition of D to write

$$D(A,C) \leq \max\left[D(A,B), D(B,C)\right]$$

as

$$\frac{1}{k_{ac}} \leq \max\left[\frac{1}{k_{ab}}, \frac{1}{k_{bc}}\right]$$

or, equivalently,

$$k_{ac} \geq \min\left[k_{ab}, k_{bc}\right]$$

where k_{ij} is the degree of similarity between shape i and shape j. Recall from the definition that k is the largest order for which the shape numbers of shape i and shape j still coincide. As Fig. 12.24(b) illustrates, this is the point at which the figures "separate" as we move further down the tree (note that k increases as we move further down the tree). We prove that $k_{ac} \geq \min[k_{ab}, k_{bc}]$ by contradiction. For $k_{ac} \leq \min[k_{ab}, k_{bc}]$ to hold, shape A has to separate from shape C *before* (1) shape A separates from shape B, *and* (2) *before* shape B separates from shape C, otherwise $k_{ab} \leq k_{ac}$ or $k_{bc} \leq k_{ac}$, which automatically violates the condition $k_{ac} < \min[k_{ab}, k_{bc}]$. But, if (1) has to hold, then Fig. P12.20 shows the only way that A can separate from C before separating from B. This, however, violates (2), which means that the condition $k_{ac} < \min[k_{ab}, k_{bc}]$ is violated (we can also see this in the figure by noting that $k_{ac} = k_{bc}$ which, since $k_{bc} < k_{ab}$, violates the condition). We use a similar argument to show that if (2) holds then (1) is violated. Thus, we conclude that it is impossible for the condition $k_{ac} < \min[k_{ab}, k_{bc}]$ to hold, thus proving that $k_{ac} \geq \min[k_{ab}, k_{bc}]$ or, equivalently, that $D(A,C) \leq \max[D(A,B), D(B,C)]$.

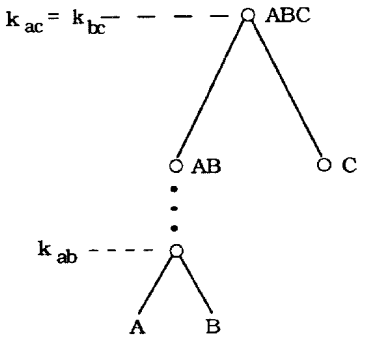


Figure P12.20

Problem 12.22

(a) An automaton capable of accepting *only* strings of the form $ab^n a \geq 1$, shown in Fig. P12.22, is given by

$$A_f = (Q, \Sigma, \delta, q_0, F),$$

with

$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3, q_\emptyset\}, \\ \Sigma &= \{a, b\}, \end{aligned}$$

mappings

$$\begin{aligned} \delta(q_0, a) &= \{q_1\}, \\ \delta(q_1, b) &= \{q_1, q_2\}, \\ \delta(q_2, a) &= \{q_3\} \end{aligned}$$

and

$$F = \{q3\}.$$

For completeness we write

$$\delta(q_0, b) = \delta(q_1, a) = \delta(q_2, b) = \delta(q_3, a) = \delta(q_3, b) = \delta(q_\emptyset, a) = \delta(q_\emptyset, b) = \{q_\emptyset\},$$

corresponding to the null state.

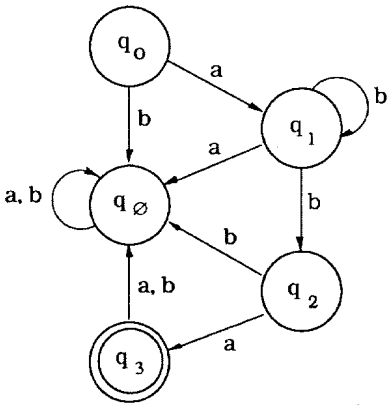


Figure P12.22

Problem 12.24

For the sample set $R^+ = \{aba, abba, abbba\}$ it is easily shown that, for $k = 1$ and 2, $h(\lambda, R^+, k) = \emptyset$, the null set. Since $q_0 = h(\lambda, R^+, k)$ is part of the inference procedure, we need to choose k large enough so that $h(\lambda, R^+, k)$ is not the null set. The shortest

string in R^+ has three symbols, so $k = 3$ is the smallest value that can accomplish this. For this value of k, a trial run will show that one more string needs to be added to R^+ in order for the inference procedure to discover iterative regularity in symbol b. The sample string set then becomes $R^+ = \{aba, abba, abbba, abbbba\}$. Recalling that $h(z, R^+, k) = \{w \,|zw \text{ in } R^+,\ |w| \leq\ k\}$ we proceed as follows:

$$
\begin{aligned}
z = \lambda, \qquad & h(\lambda, R^+, 3) && = \{w \,|\lambda w \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{aba\} \\
& && = q_0; \\
z = a, \qquad & h(a, R^+, 3) && = \{w \,|aw \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{ba, bba\} \\
& && = q_1; \\
z = ab, \qquad & h(ab, R^+, 3) && = \{w \,|abw \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{a, ba, bba\} \\
& && = q_2; \\
z = aba, \qquad & h(aba, R^+, 3) && = \{w \,|abaw \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{\lambda\} \\
& && = q_3; \\
z = abb, \qquad & h(abb, R^+, 3) && = \{w \,|abbw \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{a, ba, bba\} \\
& && = q_2; \\
z = abba, \qquad & h(abba, R^+, 3) && = \{w \,|abbaw \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{\lambda\} \\
& && = q_3; \\
z = abbb, \qquad & h(abbb, R^+, 3) && = \{w \,|abbbw \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{a, ba\} \\
& && == q_4; \\
z = abbba, \qquad & h(abbba, R^+, 3) && = \{w \,|abbbaw \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{\lambda\} \\
& && = q_3; \\
z = abbbb, \qquad & h(abbbb, R^+, 3) && = \{w \,|abbbbw \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{a\} \\
& && = q_5; \\
z = abbbba, \qquad & h(abbbba, R^+, 3) && = \{w \,|abbbbaw \text{ in } R^+, |w| \leq 3\,\} \\
& && = \{\lambda\} \\
& && = q_3;
\end{aligned}
$$

Other strings z in $\Sigma^* = (a, b)^*$ yield strings zw that do not belong to R^+, giving rise

to another state, denoted $q_\emptyset$, which corresponds to the condition that h is the null set. Therefore, the states are $q_0 = \{aba\}$, $q_1 = \{ba, bba\}$, $q_2 = \{a, ba, bba\}$, $q_3 = \{\lambda\}$, $q_4 = \{a, ba\}$, and $q_5 = \{a\}$, which gives the set $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_\emptyset\}$.

The next step is to obtain the mappings. We start by recalling that, in general, $q_0 = h(\lambda, R^+, k)$. Also in general,

$$\delta(q, c) = \{q' \text{ in } Q \,|\, q' = h(zc, R^+, k), \quad \text{with} \quad q = h(z, R+, k)\}.$$

In our case, $q_0 = h(\lambda, R^+, 3)$ and, therefore,

$$\delta(q_0, a) = h(\lambda a, R^+, 3) = h(a, R^+, 3) = \{q_1\} = q_1$$

and

$$\delta(q_0, b) = h(\lambda b, R^+, 3) = h(b, R^+, 3) = \{q_\emptyset\} = q_\emptyset,$$

where we have omitted the curly brackets for clarity in notation since the set contains only one element. Similarly, $q_1 = h(a, R^+, 3)$, and

$$\begin{aligned} \delta(q_1, a) &= h(aa, R^+, 3) = h(a, R^+, 3) = q_\emptyset, \\ \delta(q_1, b) &= h(ab, R^+, 3) = q_2. \end{aligned}$$

Continuing in this manner gives $q_2 = h(ab, R^+, 3) = h(abb, R^+, 3)$,

$$\begin{aligned} \delta(q2, a) &= h(aba, R^+, 3) = h(abba, R^+, 3) = q3, \\ \delta(q_2, b) &= h(abb, R^+, 3) = q_2, \end{aligned}$$

and, also,

$$\delta(q_2, b) = h(abbb, R^+, 3) = q_4.$$

Next, $q_3 = h(aba, R^+, 3) = h(abba, R^+, 3) = h(abbba, R^+, 3) = h(abbbba, R^+, 3)$, from which we obtain

$$\begin{aligned} \delta(q_3, a) &= h(abaa, R^+, 3) = h(abbaa, R^+, 3) \\ &= h(abbbaa, R^+, 3) = h(abbbbaa, R^+, 3) \\ &= q_\emptyset \\ \delta(q3, b) &= h(abab, R^+, 3) = h(abbab, R^+, 3) \\ &= h(abbbab, R^+, 3) = h(abbbbab, R^+, 3) \\ &= q_\emptyset; \end{aligned}$$

For the following state, $q_4 = h(abbb, R^+, 3)$,

$$\begin{aligned} \delta(q_4, a) &= h(abbba, R^+, 3) = q_3, \\ \delta(q_4, b) &= h(abbbb, R^+, 3) = q_5. \end{aligned}$$

Finally, for the last state, $q_5 = h(abbbb, R^+, 3)$, and

$$\begin{aligned}\delta(q_5, a) &= h(abbbba, R^+, 3) = q_3,\\ \delta(q_5, b) &= h(abbbbb, R^+, 3) = q_\emptyset.\end{aligned}$$

We complete the elements of the automaton by recalling that $F = \{q \,|q \text{ in } Q, \lambda \text{ in } q\} = q_3$. We also include two remaining mappings that yield the null set: $\delta(q_\emptyset, a) = \delta(q_\emptyset, b) = q_\emptyset$.

Summarizing, the state mappings are:

$$\begin{aligned}\delta(q_0, a) &= q_1, \delta(q_0, b) = q_\emptyset;\\ \delta(q_1, a) &= q_\emptyset, \delta(q_1, b) = q_2;\\ \delta(q_2, a) &= q_3, \delta(q_2, b) = \{q_2, q_4\};\\ \delta(q_3, a) &= q_\emptyset, \delta(q_3, b) = q_\emptyset;\\ \delta(q_4, a) &= q_3, \delta(q_4, b) = q_5;\\ \delta(q_5, a) &= q_3, \delta(q_5, b) = q_\emptyset;\\ \delta(q_\emptyset, a) &= q_\emptyset, \delta(q_\emptyset, b) = q_\emptyset.\end{aligned}$$

A diagram of the automaton is shown in Fig. P12.24. The iterative regularity on b is evident in state q_2. This automaton is not as elegant as its counterpart in Problem 12.22(a). This is not unexpected because nothing in the inference procedure deals with state minimization. Note, however, that the automaton accepts only strings of the form $ab^n a$, $b \geq 1$, as desired. The minimization aspects of a design generally follow inference and are based on one of several standard methods (see, for example, Gonzalez and Thomason [1978]). In this particular example, even visual inspection reveals that states q_4 and q_5 are redundant.

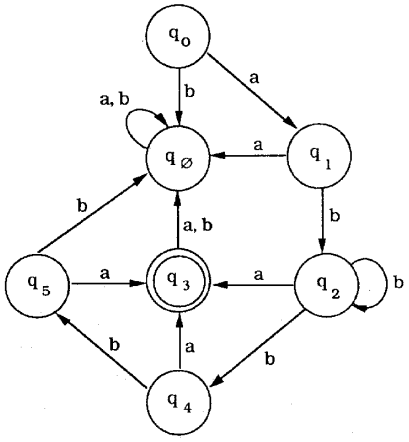


Figure P12.24