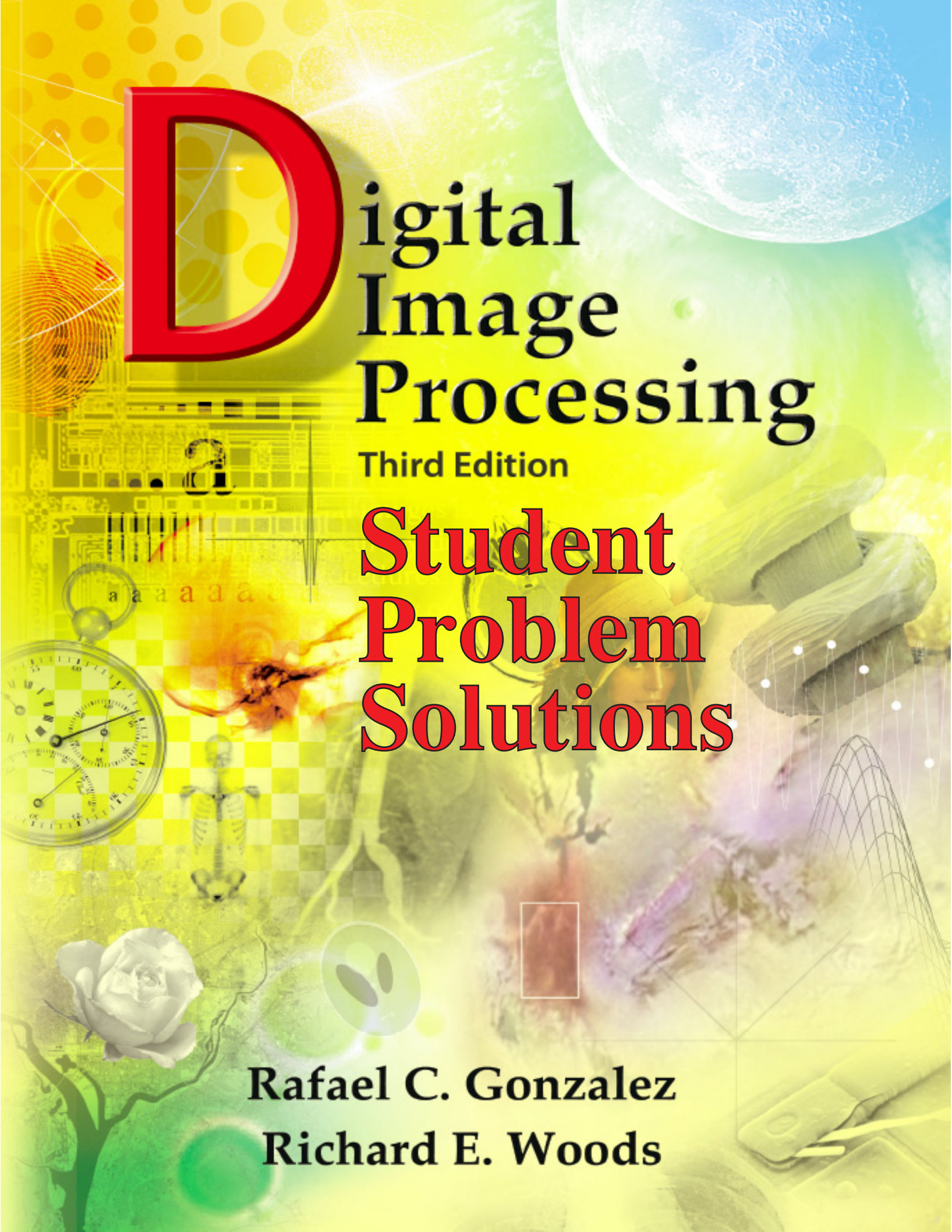

Digital Image Processing
Third Edition
Student Problem Solutions
Rafael C. Gonzalez
Richard E. Woods

Digital Image Processing

Third Edition

Student Problem Solutions

Version 3.0

Rafael C. Gonzalez
Richard E. Woods

Prentice Hall
Upper Saddle River, NJ 07458

www.imageprocessingplace.com

Chapter 1

Introduction

1.1 About This Manual

This abbreviated manual contains detailed solutions to all problems marked with a star in *Digital Image Processing*, 3rd Edition.

1.2 Projects

You may be asked by your instructor to prepare comptuter projects in the following format:

Page 1: Cover page.

- Project title
- Project number
- Course number
- Student's name
- Date due
- Date handed in
- Abstract (not to exceed 1/2 page)

Page 2: One to two pages (max) of technical discussion.

Page 3 (or 4): Discussion of results. One to two pages (max).

Results: Image results (printed typically on a laser or inkjet printer). All images must contain a number and title referred to in the discussion of results.

Appendix: Program listings, focused on any original code prepared by the student. For brevity, functions and routines provided to the student are referred to by name, but the code is not included.

Layout: The entire report must be on a standard sheet size (e.g., letter size in the U.S. or A4 in Europe), stapled with three or more staples on the left margin to form a booklet, or bound using clear plastic standard binding products.

1.3 About the Book Web Site

The companion web site

www.prenhall.com/gonzalezwoods

(or its mirror site)

www.imageprocessingplace.com

is a valuable teaching aid, in the sense that it includes material that previously was covered in class. In particular, the review material on probability, matrices, vectors, and linear systems, was prepared using the same notation as in the book, and is focused on areas that are directly relevant to discussions in the text. This allows the instructor to assign the material as independent reading, and spend no more than one total lecture period reviewing those subjects. Another major feature is the set of solutions to problems marked with a star in the book. These solutions are quite detailed, and were prepared with the idea of using them as teaching support. The on-line availability of projects and digital images frees the instructor from having to prepare experiments, data, and handouts for students. The fact that most of the images in the book are available for downloading further enhances the value of the web site as a teaching resource.

Chapter 2

Problem Solutions

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,

$$\frac{(d/2)}{0.2} = \frac{(x/2)}{0.017}$$

which gives $x = 0.085d$. 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 $0.085(d) < 1.3 \times 10^{-6}$ m, or $d < 15.3 \times 10^{-6}$ m.

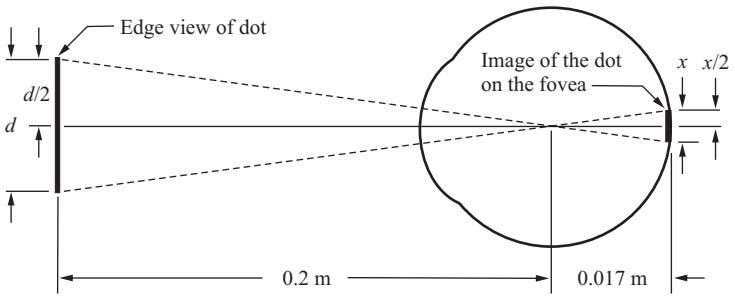


Figure P2.1

Problem 2.3

The solution is

$$\begin{aligned}\lambda &= c/v \\ &= 2.998 \times 10^8 (\text{m/s})/60(1/\text{s}) \\ &= 4.997 \times 10^6 \text{ m} = 4997 \text{ Km}.\end{aligned}$$

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 then would be 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 56K baud link is $(1024)^2 \times [8+2]/56000 = 187.25$ sec or about 3.1 min.

(b) At 3000K this time goes down to about 3.5 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.

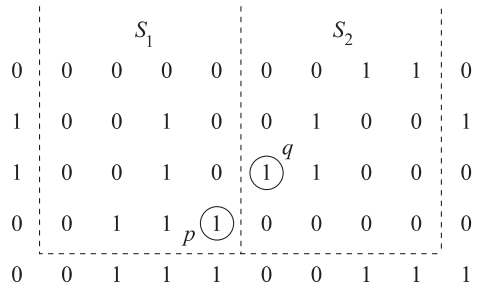


Figure P2.11

Problem 2.12

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

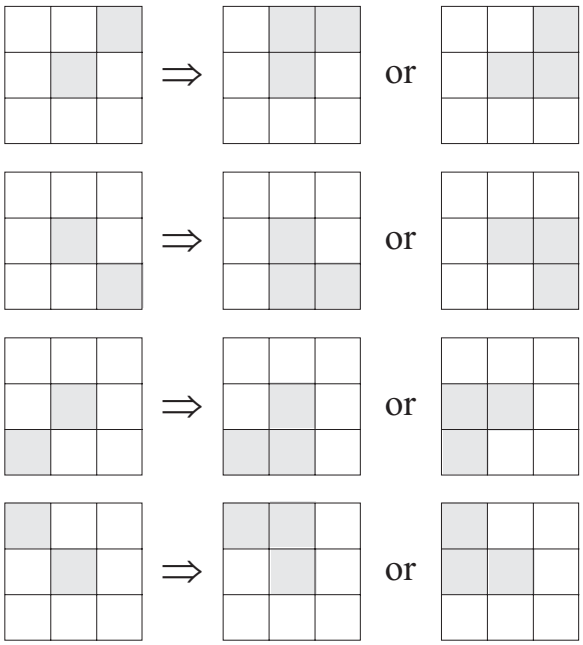


Figure P2.12

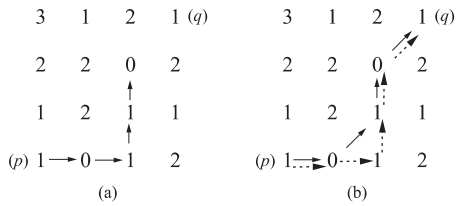


Figure P.2.15

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 the shortest m- path (shown dashed) is 5. Both of these shortest paths are unique in this case.

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. (2.5-2). (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).

Problem 2.18

With reference to Eq. (2.6-1), let H denote the 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 2.6.1. 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 in a given neighborhood. Then, $H(aS_1 + bS_2)$ means: (1) multiply the pixels in each of the subimage areas by the constants shown, (2) add

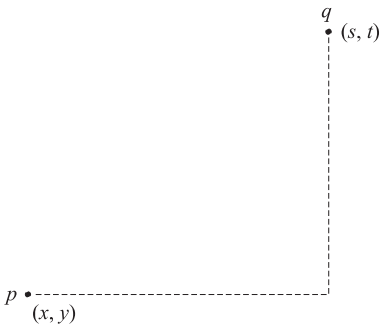


Figure P2.16

the pixel-by-pixel values from aS_1 and bS_2 (which produces a single subimage area), and (3) compute 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.

Problem 2.20

From Eq. (2.6-5), at any point (x, y),

$$\overline{g} = \frac{1}{K} \sum_{i=1}^{K} g_i = \frac{1}{K} \sum_{i=1}^{K} f_i + \frac{1}{K} \sum_{i=1}^{K} \eta_i.$$

Then

$$E\{\overline{g}\} = \frac{1}{K} \sum_{i=1}^{K} E\{f_i\} + \frac{1}{K} \sum_{i=1}^{K} E\{\eta_i\}.$$

But all the f_i are the same image, so $E\{f_i\} = f$. Also, it is given that the noise has zero mean, so $E\{\eta_i\} = 0$. Thus, it follows that $E\{\overline{g}\} = f$, which proves the validity of Eq. (2.6-6).

To prove the validity of Eq. (2.6-7) consider the preceding equation again:

$$\overline{g} = \frac{1}{K}\sum_{i=1}^{K} g_i = \frac{1}{K}\sum_{i=1}^{K} f_i + \frac{1}{K}\sum_{i=1}^{K} \eta_i.$$

It is known from random-variable theory that the variance of the sum of uncorrelated random variables is the sum of the variances of those variables (Papoulis [1991]). Because it is given that the elements of f are constant and the η_i are uncorrelated, then

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

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

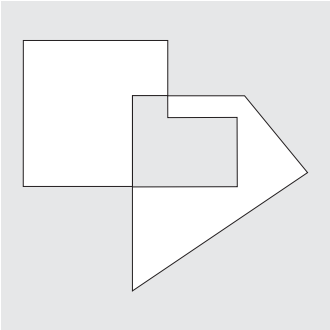
$$\sigma^2_{\overline{g}} = \frac{K}{K^2}\sigma^2_\eta = \frac{1}{K}\sigma^2_\eta$$

which proves the validity of Eq. (2.6-7).

Problem 2.22

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 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, so that 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 $\left|d(x,y)\right|$ and compare the sum against a threshold Q. Note that the absolute value needs to be used to avoid errors canceling 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,



$(A \cap B) \cup (A \cup B)^c$

Figure P2.23

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 used often 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 used often 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 2.23

(a) The answer is shown in Fig. P2.23.

Problem 2.26

From Eq. (2.6-27) and the definition of separable kernels,

$$\begin{aligned} T(u,v) &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)r(x,y,u,v) \\ &= \sum_{x=0}^{M-1} r_1(x,u)\sum_{y=0}^{N-1} f(x,y)r_2(y,v) \\ &= \sum_{x=0}^{M-1} T(x,v)r_1(x,u) \end{aligned}$$

where

$$T(x,v)=\sum_{y=0}^{N-1} f(x,y)r_2(y,v).$$

For a fixed value of x, this equation is recognized as the 1-D transform along one row of $f(x,y)$. By letting x vary from 0 to $M-1$ we compute the entire array $T(x,v)$. Then, by substituting this array into the last line of the previous equation we have the 1-D transform along the columns of $T(x,v)$. In other words, when a kernel is separable, we can compute the 1-D transform along the rows of the image. Then we compute the 1-D transform along the columns of this intermediate result to obtain the final 2-D transform, $T(u,v)$. We obtain the same result by computing the 1-D transform along the columns of $f(x,y)$ followed by the 1-D transform along the rows of the intermediate result.

This result plays an important role in Chapter 4 when we discuss the 2-D Fourier transform. From Eq. (2.6-33), the 2-D Fourier transform is given by

$$T(u,v)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi\left(ux/M+vy/N\right)}.$$

It is easily verified that the Fourier transform kernel is separable (Problem 2.25), so we can write this equation as

$$\begin{aligned} T(u,v) &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi\left(ux/M+vy/N\right)} \\ &= \sum_{x=0}^{M-1} e^{-j2\pi(ux/M)}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi\left(vy/N\right)} \\ &= \sum_{x=0}^{M-1} T(x,v)e^{-j2\pi(ux/M)} \end{aligned}$$

where

$$T(x,v)=\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(vy/N)}$$

is the 1-D Fourier transform along the rows of $f(x,y)$, as we let $x=0,1,\ldots,M-1$.

Chapter 3

Problem Solutions

Problem 3.1

Let f denote the original image. First subtract the minimum value of f denoted f_{min} from f to yield a function whose minimum value is 0:

$$g_1 = f - f_{\text{min}}$$

Next divide g_1 by its maximum value to yield a function in the range $[0,1]$ and multiply the result by $L-1$ to yield a function with values in the range $[0, L-1]$

$$\begin{aligned} g &= \frac{L-1}{\max(g1)} g1 \\ &= \frac{L-1}{\max(f - f_{\text{min}})} (f - f_{\text{min}}) \end{aligned}$$

Keep in mind that f_{min} is a scalar and f is an image.

Problem 3.3

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

Problem 3.5

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

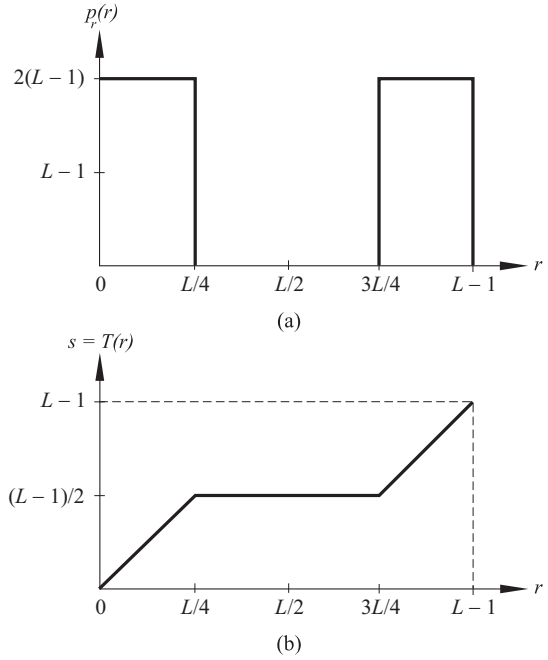


Figure P3.9.

Problem 3.6

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 actually be 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 $n = MN$ is the total number of pixels in the input image. The histogram equalization method has no provisions for this type of (artificial) intensity redistribution process.

Problem 3.9

We are interested in just one example in order to satisfy the statement of the problem. Consider the probability density function in Fig. P3.9(a). A plot of the transformation $T(r)$ in Eq. (3.3-4) using this particular density function is shown in Fig. P3.9(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 con-

ditions (a) and (b) stated in that section. However, we see from Fig. P3.9(b) that the inverse transformation from s back to r is not single valued, as there are an infinite number of possible mappings from $s = (L-1)/2$ 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 $[L/4, 3L/4]$.

Problem 3.10

(b) If none of the intensity levels r_k, $k = 1, 2, \ldots, L-1$, are 0, then $T(r_k)$ will be strictly monotonic. This implies a one-to-one mapping both ways, meaning that both forward and inverse transformations will be single-valued.

Problem 3.12

The value of the histogram component corresponding to the kth 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 intensity level r_k, n is the total number of pixels in the neighborhood, and K is the total number of possible intensity levels. Suppose that the neighborhood is moved one pixel to the right (we are assuming rectangular neighborhoods). This deletes the left-most 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.

Problem 3.13

The purpose of this simple problem is to make the student think of the meaning of histograms and arrive at the conclusion that histograms carry no information about spatial properties of images. Thus, the only time that the histogram of the images formed by the operations shown in the problem statement can be determined in terms of the original histograms is when one (both) of the images is (are) constant. In (d) we have the additional requirement that none of the pixels of $g(x, y)$ can be 0. Assume for convenience that the histograms are not normalized, so that, for example, $h_f(r_k)$ is the number of pixels in $f(x, y)$ having intensity level r_k. Assume also that all the pixels in $g(x, y)$ have constant value c. The pixels of both images are assumed to be positive. Finally, let u_k denote the intensity levels of the pixels of the images formed by any of the arithmetic operations given in the problem statement. Under the preceding set of conditions, the histograms are determined as follows:

(a) We obtain the histogram $h_{\text{sum}}(u_k)$ of the sum by letting $u_k = r_k + c$, and also $h_{\text{sum}}(u_k) = h_f(r_k)$ for all k. In other words, the values (height) of the components of h_{sum} are the same as the components of h_f, but their locations on the intensity axis are shifted right by an amount c.

Problem 3.15

(a) Consider a 3×3 mask first. Because 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 intensity values 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 3.16

(a) The key to solving this problem is to recognize (1) that the convolution result at any location (x,y) consists of centering the mask at that point and then forming the sum of the products of the mask coefficients with the corresponding pixels in the image; and (2) that convolution of the mask with the entire image results in every pixel in the image being visited only once by every element of the mask (i.e., every pixel is multiplied once by every coefficient of the mask). Because the coefficients of the mask sum to zero, this means that the sum of the products of the coefficients with the same pixel also sum to zero. Carrying out this argument for every pixel in the image leads to the conclusion that the sum of the elements of the convolution array also sum to zero.

Problem 3.18

(a) There are n^2 points in an $n \times n$ median filter mask. Because 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, because 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 3.19

(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

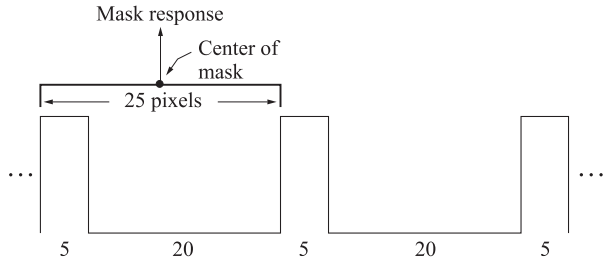


Figure P3.21

in the appropriate locations in the sorted array.

Problem 3.21

From Fig. 3.33 we know that 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. P3.21. 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 one 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 why no white gaps are 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.

Problem 3.24

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 3.25

The Laplacian mask with a -4 in the center performs an operation proportional to differentiation in the horizontal and vertical directions. Consider for a moment a 3×3 "Laplacian" mask with a -2 in the center and 1s above and below the center. All other elements are 0. This mask will perform differentiation in only one direction, and will ignore intensity transitions in the orthogonal direction. An image processed with such a mask will exhibit sharpening in only one direction. A Laplacian mask with a -4 in the center and 1s in the vertical and horizontal directions will obviously produce an image with sharpening in both directions and in general will appear sharper than with the previous mask. Similarly, and mask with a -8 in the center and 1s in the horizontal, vertical, and diagonal directions will detect the same intensity changes as the mask with the -4 in the center but, in addition, it will also be able to detect changes along the diagonals, thus generally producing sharper-looking results.

Problem 3.28

Consider the following equation:

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

where $\overline{f}(x,y)$ denotes the average of $f(x,y)$ in a predefined neighborhood centered at (x,y) and including 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 within the just-mentioned proportionality factors to be of the same form as the definition of unsharp masking given in Eqs. (3.6-8) and (3.6-9). Thus, it has been demonstrated that subtracting the Laplacian from an image is proportional to unsharp masking.

Problem 3.33

The thickness of the boundaries increases as a the size of the filtering neighborhood increases. We support this conclusion with an example. Consider a one-pixel-thick straight black line running vertically through a white image. If a 3×3 neighborhood is used, any neighborhoods whose centers are more than two pixels away from the line will produce differences with values of zero and the center pixel will be designated a region pixel. Leaving the center pixel at same location, if we increase the size of the neighborhood to, say, 5×5, the line will be encompassed and not all differences be zero, so the center pixel will now be designated a boundary point, thus increasing the thickness of the boundary. As the size of the neighborhood increases, we would have to be further and further from the line before the center point ceases to be called a boundary point. That is, the thickness of the boundary detected increases as the size of the neighborhood increases.

Problem 3.34

(a) If the intensity of the center pixel of a 3×3 region is larger than the intensity of all its neighbors, then decrease it. If the intensity is smaller than the intensity of all its neighbors, then increase it. Else, do not nothing.

(b) Rules

IF d_2 is PO AND d_4 is PO AND d_6 is PO AND d_8 is PO THEN v is PO
IF d_2 is NE AND d_4 is NE AND d_6 is NE AND d_8 is NE THEN v is NE
ELSE v is ZR.

Note: In rule 1, all positive differences mean that the intensity of the noise pulse (z_5) is less than that of all its 4-neighbors. Then we'll want to make the output z_5' more positive so that when it is added to z_5 it will bring the value of the center pixel closer to the values of its neighbors. The converse is true when all the differences are negative. A mixture of positive and negative differences calls for no action because the center pixel is not a clear spike. In this case the correction should be zero (keep in mind that zero is a fuzzy set too).

Chapter 4

Problem Solutions

Problem 4.2

(a) To prove infinite periodicity in both directions with period $1/\Delta T$, we have to show that $\tilde{F}\left(\mu + k[1/\Delta T]\right) = \tilde{F}(\mu)$ for $k = 0, \pm 1, \pm 2, \ldots$. From Eq. (4.3-5),

$$\begin{aligned}
\tilde{F}\left(\mu + k[1/\Delta T]\right) &= \frac{1}{\triangle T}\sum_{n=-\infty}^{\infty} F\left(\mu + \frac{k}{\triangle T} - \frac{n}{\triangle T}\right) \\
&= \frac{1}{\triangle T}\sum_{n=-\infty}^{\infty} F\left(\mu + \frac{k-n}{\triangle T}\right) \\
&= \frac{1}{\triangle T}\sum_{m=-\infty}^{\infty} F\left(\mu - \frac{m}{\triangle T}\right) \\
&= \tilde{F}\left(\mu\right)
\end{aligned}$$

where the third line follows from the fact that k and n are integers and the limits of summation are symmetric about the origin. The last step follows from Eq. (4.3-5).

(b) Again, we need to show that $\tilde{F}\left(\mu + k/\Delta T\right) = \tilde{F}(\mu)$ for $k = 0, \pm 1, \pm 2, \ldots$. From Eq. (4.4-2),

$$\begin{aligned}\tilde{F}(\mu+k/\Delta T) &= \sum_{n=-\infty}^{\infty} f_n e^{-j2\pi(\mu+k/\Delta T)n\Delta T} \\ &= \sum_{n=-\infty}^{\infty} f_n e^{-j2\pi\mu n\Delta T} e^{-j2\pi kn} \\ &= \sum_{n=-\infty}^{\infty} f_n e^{-j2\pi\mu n\Delta T} \\ &= \tilde{F}(\mu)\end{aligned}$$

where the third line follows from the fact that $e^{-j2\pi kn} = 1$ because both k and n are integers (see Euler's formula), and the last line follows from Eq. (4.4-2).

Problem 4.3

From the definition of the 1-D Fourier transform in Eq. (4.2-16),

$$\begin{aligned}F(\mu) &= \int_{-\infty}^{\infty} f(t)\, e^{-j2\pi\mu t}\, dt \\ &= \int_{-\infty}^{\infty} \sin(2\pi nt)\, e^{-j2\pi\mu t}\, dt \\ &= \frac{-j}{2}\int_{-\infty}^{\infty} \left[e^{j2\pi nt} - e^{-j2\pi nt}\right] e^{-j2\pi\mu t}\, dt \\ &= \frac{-j}{2}\int_{-\infty}^{\infty} \left[e^{j2\pi nt}\right] e^{-j2\pi\mu t}\, dt - \frac{-j}{2}\int_{-\infty}^{\infty} \left[e^{-j2\pi nt}\right] e^{-j2\pi\mu t}\, dt.\end{aligned}$$

From the translation property in Table 4.3 we know that

$$f(t)e^{j2\pi\mu_0 t} \Leftrightarrow F(\mu-\mu_0)$$

and we know from the statement of the problem that the Fourier transform of a constant $[f(t)=1]$ is an impulse. Thus,

$$(1)\, e^{j2\pi\mu_0 t} \Leftrightarrow \delta(\mu-\mu_0).$$

Thus, we see that the leftmost integral in the the last line above is the Fourier transform of $(1)\, e^{j2\pi nt}$, which is $\delta(\mu-n)$, and similarly, the second integral is the transform of $(1)\, e^{-j2\pi nt}$, or $\delta(\mu+n)$. Combining all results yields

$$F(\mu) = \frac{j}{2}\left[\delta(\mu+n) - \delta(\mu-n)\right]$$

as desired.

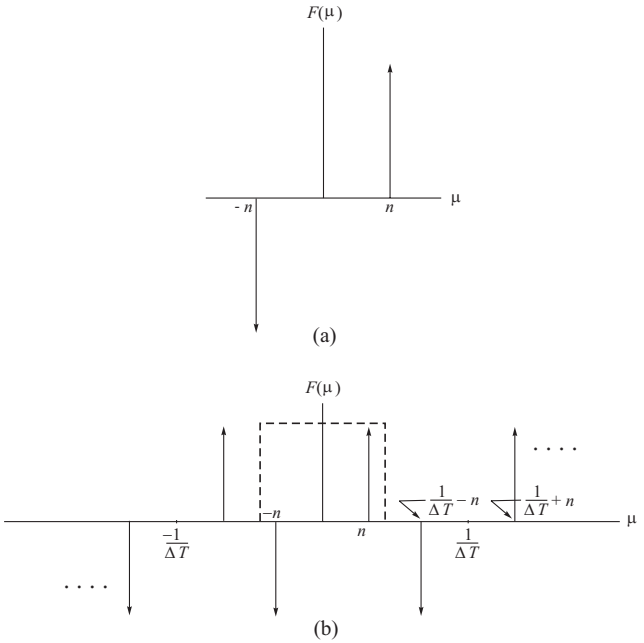


Figure P4.4

Problem 4.4

(a) The period is such that $2\pi n t = 2\pi$, or $t = 1/n$.

(b) The frequency is 1 divided by the period, or n. The continuous Fourier transform of the given sine wave looks as in Fig. P4.4(a) (see Problem 4.3), and the transform of the sampled data (showing a few periods) has the general form illustrated in Fig. P4.4(b) (the dashed box is an ideal filter that would allow reconstruction if the sine function were sampled, with the sampling theorem being satisfied).

(c) The Nyquist sampling rate is exactly twice the highest frequency, or $2n$. That is, $(1/\Delta T) = 2n$, or $\Delta T = 1/2n$. Taking samples at $t = \pm\Delta T, \pm 2\Delta T, \ldots$ would yield the sampled function $\sin(2\pi n \Delta T)$ whose values are all 0s because $\Delta T = 1/2n$ and n is an integer. In terms of Fig. P4.4(b), we see that when $\Delta T = 1/2n$ all the positive and negative impulses would coincide, thus canceling each other and giving a result of 0 for the sampled data.

Problem 4.5

Starting from Eq. (4.2-20),

$$f(t)\star g(t)=\int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau.$$

The Fourier transform of this expression is

$$\begin{aligned}\Im\left[f(t)\star g(t)\right] &= \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau\right]e^{-j2\pi\mu t}dt \\ &= \int_{-\infty}^{\infty} f(\tau)\left[\int_{-\infty}^{\infty} g(t-\tau)e^{-j2\pi\mu t}dt\right]d\tau.\end{aligned}$$

The term inside the inner brackets is the Fourier transform of $g(t-\tau)$. But, we know from the translation property (Table 4.3) that

$$\Im\left[g(t-\tau)\right]=G(\mu)e^{-j2\pi\mu\tau}$$

so

$$\begin{aligned}\Im\left[f(t)\star g(t)\right] &= \int_{-\infty}^{\infty} f(\tau)\left[G(\mu)e^{-j2\pi\mu\tau}\right]d\tau \\ &= G(\mu)\int_{-\infty}^{\infty} f(\tau)e^{-j2\pi\mu\tau}d\tau \\ &= G(\mu)F(\mu).\end{aligned}$$

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 a similar way.

Problem 4.8

(b) We solve this problem as above, by direct substitution and using orthogonality. Substituting Eq. (4.4-7) into (4.4-6) yields

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

where the last step follows from the orthogonality condition given in the problem statement. Substituting Eq. (4.4-6) into (4.6-7) and using the same basic procedure yields a similar identity for $f(x)$.

Problem 4.10

With reference to the statement of the convolution theorem given in Eqs. (4.2-21) and (4.2-22), we need to show that

$$f(x)\star h(x) \Leftrightarrow F(u)H(u)$$

and that

$$f(x)h(x) \Leftrightarrow F(u)\star H(u).$$

From Eq. (4.4-10) and the definition of the DFT in Eq. (4.4-6),

$$\begin{aligned}
\Im\left[f(x)\star h(x)\right] &= \sum_{x=0}^{M-1}\left[\sum_{m=0}^{M-1} f(m)h(x-m)\right]e^{-j2\pi ux/M} \\
&= \sum_{m=0}^{M-1} f(m)\left[\sum_{x=0}^{M-1} h(x-m)e^{-j2\pi ux/M}\right] \\
&= \sum_{m=0}^{M-1} f(m)H(u)e^{-j2\pi um/M} \\
&= H(u)\sum_{m=0}^{M-1} f(m)e^{-j2\pi um/M} \\
&= H(u)F(u).
\end{aligned}$$

The other half of the discrete convolution theorem is proved in a similar manner.

Problem 4.11

With reference to Eq. (4.2-20),

$$f(t,z)\star h(t,z) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)h(t-\alpha,z-\beta)d\alpha d\beta.$$

Problem 4.14

From Eq. (4.5-7),

$$F(\mu,v) = \Im\left[f(t,z)\right] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t,z)e^{-j2\pi(\mu t+vz)}dt\,dz.$$

Recall that in this chapter we use (t,z) and (μ,v) for continuous variables, and (x,y) and (u,v) for discrete variables.

From Eq. (2.6-2), the Fourier transform operation is linear if

$$\Im\left[a_1 f_1(t,z)+a_2 f_2(t,z)\right] = a_1\Im\left[f_1(t,z)\right]+a_2\Im\left[f_2(t,z)\right].$$

Substituting into the definition of the Fourier transform yields

$$\begin{aligned}
\Im\left[a_1 f_1(t,z)+a_2 f_2(t,z)\right] &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[a_1 f_1(t,z)+a_2 f_2(t,z)\right] \\
&\qquad \times e^{-j2\pi(\mu t+\nu z)}\,dt\,dz \\
&= a_1\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(t,z)e^{-j2\pi(\mu t+\nu z)}\,dt\,dz \\
&\qquad + a_2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_2(t,z)e^{-j2\pi(\mu t+\nu z)}\,dt\,dz \\
&= a_1\Im\left[f_1(t,z)\right]+a_2\Im\left[f_2(t,z)\right].
\end{aligned}$$

where the second step follows from the distributive property of the integral. Similarly, for the discrete case,

$$\begin{aligned}
\Im\left[a_1 f_1(x,y)+a_2 f_2(x,y)\right] &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[a_1 f_1(x,y)+a_2 f_2(x,y)\right] e^{-j2\pi(ux/M+vy/N)} \\
&= a_1\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f_1(x,y)e^{-j2\pi(ux/M+vy/N)} \\
&\qquad + a_2\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f_2(x,y)e^{-j2\pi(ux/M+vy/N)} \\
&= a_1\Im\left[f_1(x,y)\right]+a_2\Im\left[f_2(x,y)\right].
\end{aligned}$$

The linearity of the inverse transforms is proved in exactly the same way.

Problem 4.16

(a) From Eq. (4.5-15),

$$\begin{aligned}
\Im\left[f(x,y)e^{j2\pi(u_0x+v_0y)}\right] &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[f(x,y)e^{j2\pi(u_0x+v_0y)}\right] e^{-j2\pi(ux/M+vy/N)} \\
&= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-j2\pi\left[(u-u_0)x/M+(v-v_0)y/N\right]} \\
&= F(u-u_0,v-v_0).
\end{aligned}$$

Problem 4.20

The following are proofs of some of the properties in Table 4.1. Proofs of the other properties are given in Chapter 4. Recall that when we refer to a function as imaginary, its real part is zero. We use the term complex to denote a function whose real and imaginary parts are not zero. We prove only the forward part the Fourier transform pairs. Similar techniques are used to prove the inverse part.

(a) *Property 2:* If $f(x,y)$ is imaginary, $f(x,y) \Leftrightarrow F^*(-u,-v) = -F(u,v)$. *Proof:* Because $f(x,y)$ is imaginary, we can express it as $j\,g(x,y)$, where $g(x,y)$ is a real function. Then the proof is as follows:

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

(b) *Property 4:* If $f(x,y)$ is imaginary, then $R(u,v)$ is odd and $I(u,v)$ is even. *Proof:* F is complex, so it can be expressed as

$$\begin{aligned}
F(u,v) &= \text{real}\,[F(u,v)] + j\,\text{imag}\,[F(u,v)] \\
&= R(u,v) + j\,I(u,v).
\end{aligned}$$

Then, $-F(u,v) = -R(u,v) - j\,I(u,v)$ and $F^*(-u,-v) = R(-u,-v) - j\,I(-u,-v)$. But, because $f(x,y)$ is imaginary, $F^*(-u,-v) = -F(u,v)$ (see Property 2). It then follows from the previous two equations that $R(u,v) = -R(-u,-v)$ (i.e., R is odd) and $I(u,v) = I(-u,-v)$ (I is even).

(d) *Property 7:* When $f(x,y)$ is complex, $f^*(x,y) \Leftrightarrow F^*(-u,-v)$. *Proof:*

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

(g) *Property 11:* If $f(x,y)$ is imaginary and odd, then $F(u,v)$ is real and odd, and conversely. *Proof:* If $f(x,y)$ is imaginary, we know that the real part of $F(u,v)$ is odd and its imaginary part is even. If we can show that the imaginary part is zero, then we will have the proof for this property. As above,

$$\begin{aligned}F(u,v) &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[j\text{odd}]\left[(\text{even})(\text{even})-2j(\text{even})(\text{odd})-(\text{odd})(\text{odd})\right] \\ &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[j\text{odd}][\text{even}-j\text{odd}][\text{even}-j\text{odd}] \\ &= j\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[(\text{odd})(\text{even})]+2\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[(\text{even})(\text{even})] \\ &\qquad -j\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[(\text{odd})(\text{even})] \\ &= \text{real}\end{aligned}$$

where the last step follows from Eq. (4.6-13).

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 identical. Thus, either form of padding accomplishes the same separation between images, as desired.

Problem 4.22

Unless all borders on of an image are black, padding the image with 0s introduces significant discontinuities (edges) at one or more borders of the image.

These can be strong horizontal and vertical edges. These sharp transitions in the spatial domain introduce high-frequency components along the vertical and horizontal axes of the spectrum.

Problem 4.23

(a) The averages of the two images are computed as follows:

$$\bar{f}(x,y) = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)$$

and

$$\begin{aligned}\bar{f}_p(x,y) &= \frac{1}{PQ}\sum_{x=0}^{P-1}\sum_{y=0}^{Q-1} f_p(x,y)\\ &= \frac{1}{PQ}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\\ &= \frac{MN}{PQ}\bar{f}(x,y)\end{aligned}$$

where the second step is result of the fact that the image is padded with 0s. Thus, the ratio of the average values is

$$r = \frac{PQ}{MN}$$

Thus, we see that the ratio increases as a function of PQ, indicating that the average value of the padded image decreases as a function of PQ. This is as expected; padding an image with zeros decreases its average value.

Problem 4.25

(a) From Eq. (4.4-10) and the definition of the 1-D DFT,

$$\begin{aligned}\Im\left[f(x)\bigstar h(x)\right] &= \sum_{x=0}^{M-1} f(x)\bigstar h(x)e^{-j2\pi ux/M}\\ &= \sum_{x=0}^{M-1}\sum_{m=0}^{M-1} f(m)h(x-m)e^{-j2\pi ux/M}\\ &= \sum_{m=0}^{M-1} f(m)\sum_{x=0}^{M-1} h(x-m)e^{-j2\pi ux/M}\end{aligned}$$

but

$$\sum_{x=0}^{M-1} h(x-m)e^{-j2\pi ux/M} = \Im[h(x-m)] = H(u)e^{-j2\pi mu/M}$$

where the last step follows from Eq. (4.6-4). Substituting this result into the previous equation yields

$$\begin{aligned}\Im\left[f(x)\bigstar h(x)\right] &= \sum_{m=0}^{M-1} f(m)e^{-j2\pi mu/M}H(u) \\ &= F(u)H(u).\end{aligned}$$

The other part of the convolution theorem is done in a similar manner.

(c) Correlation is done in the same way, but because of the difference in sign in the argument of h the result will be a conjugate:

$$\begin{aligned}\Im\left[f(x,y)\star h(x,y)\right] &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)\star h(x,y)e^{-j2\pi(ux/M+vy/N)} \\ &= \sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\left[\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)h(x+m,y+n)\right] \\ &\qquad \times e^{-j2\pi(ux/M+vy/N)} \\ &= \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} h(x+m,y+n) \\ &\qquad \times e^{-j2\pi(ux/M+vy/N)} \\ &= \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)e^{j2\pi(um/M+vn/N)}H(u,v) \\ &= F^*(u,v)H(u,v).\end{aligned}$$

(d) We begin with one variable:

$$\Im\left[\frac{df(z)}{dz}\right] = \int_{-\infty}^{\infty}\frac{df(z)}{dz}e^{-j2\pi vz}dz$$

Integration by parts has the following general form,

$$\int s\,dw = sw - \int w\,ds.$$

Let $s = e^{-j2\pi vz}$ and $w = f(z)$. Then, $dw/dz = df(z)/dz$ or

$$dw = \frac{df(z)}{dz}dz \qquad \text{and} \qquad ds = (-j2\pi v)e^{-j2\pi vz}dz$$

so it follows that

$$\begin{aligned}\Im\left[\frac{df(z)}{dz}\right] &= \int_{-\infty}^{\infty}\frac{df(z)}{dz}e^{-j2\pi vz}dz \\ &= f(z)\,e^{-j2\pi vz}\Big]_{\infty}^{\infty}-\int_{-\infty}^{\infty}f(z)(-j2\pi v)e^{-j2\pi vz}dz \\ &= (j2\pi v)\int_{-\infty}^{\infty}f(z)e^{-j2\pi vz}dz \\ &= (j2\pi v)F(v)\end{aligned}$$

because $f(\pm\infty)=0$ by assumption (see Table 4.3). Consider next the second derivative. Define $g(z)=df(z)/dz$. Then

$$\Im\left[\frac{dg(z)}{dz}\right]=(j2\pi v)G(v)$$

where $G(v)$ is the Fourier transform of $g(z)$. But $g(z)=df(z)/dz$, so $G(v)=(j2\pi v)F(v)$, and

$$\Im\left[\frac{d^2f(z)}{dz^2}\right]=(j2\pi v)^2F(v).$$

Continuing in this manner would result in the expression

$$\Im\left[\frac{d^nf(z)}{dz^n}\right]=(j2\pi v)^nF(v).$$

If we now go to 2-D and take the derivative of only one variable, we would get the same result as in the preceding expression, but we have to use partial derivatives to indicate the variable to which differentiation applies and, instead of $F(\mu)$, we would have $F(\mu,v)$. Thus,

$$\Im\left[\frac{\partial^nf(t,z)}{\partial z^n}\right]=(j2\pi v)^nF(\mu,v).$$

Define $g(t,z)=\partial^nf(t,z)/\partial t^n$, then

$$\Im\left[\frac{\partial^mg(t,z)}{\partial t^m}\right]=(j2\pi\mu)^mG(\mu,v).$$

But G(μ,v) is the transform of $g(t,z)=\partial^nf(t,z)/\partial t^n$, which we know is equal to $(j2\pi\mu)^nF(\mu,v)$. Therefore, we have established that

$$\Im\left[\left(\frac{\partial}{\partial t}\right)^m\left(\frac{\partial}{\partial z}\right)^nf(t,z)\right]=(j2\pi\mu)^m(j2\pi v)^nF(\mu,v).$$

Because the Fourier transform is unique, we know that the inverse transform of the right of this equation would give the left, so the equation constitutes a Fourier transform pair (keep in mind that we are dealing with continuous variables).

Problem 4.26

(b) As the preceding derivation shows, the Laplacian filter applies to *continuous* variables. We can generate a filter for using with the DFT simply by sampling this function:

$$H(u,v) = -4\pi^2(u^2+v^2)$$

for $u = 0,1,2,\ldots,M-1$ and $v = 0,1,2,\ldots,N-1$. When working with centered transforms, the Laplacian filter function in the frequency domain is expressed as

$$H(u,v) = -4\pi^2([u-M/2]^2+[v-N/2]^2).$$

In summary, we have the following Fourier transform pair relating the Laplacian in the spatial and frequency domains:

$$\nabla^2 f(x,y) \Leftrightarrow -4\pi^2([u-M/2]^2+[v-N/2]^2)F(u,v)$$

where it is understood that the filter is a sampled version of a continuous function.

(c) The Laplacian filter is isotropic, so its symmetry is approximated much closer by a Laplacian mask having the additional diagonal terms, which requires a -8 in the center so that its response is 0 in areas of constant intensity.

Problem 4.27

(a) The spatial average (excluding the center term) is

$$g(x,y) = \frac{1}{4}\left[f(x,y+1)+f(x+1,y)+f(x-1,y)+f(x,y-1)\right].$$

From property 3 in Table 4.3,

$$\begin{aligned} G(u,v) &= \frac{1}{4}\left[e^{j2\pi v/N}+e^{j2\pi u/M}+e^{-j2\pi u/M}+e^{-j2\pi v/N}\right]F(u,v) \\ &= H(u,v)F(u,v) \end{aligned}$$

where

$$H(u,v) = \frac{1}{2}\left[\cos(2\pi u/M)+\cos(2\pi v/N)\right]$$

is the filter transfer function in the frequency domain.

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

$$H(u,v) = \frac{1}{2}\left[\cos(2\pi[u - M/2])/M) + \cos(2\pi[v - N/2]/N)\right].$$

Consider one variable for convenience. As u ranges from 0 to $M-1$, the value of $\cos(2\pi[u - M/2]/M)$ starts at -1, peaks at 1 when $u = M/2$ (the center of the filter) and then decreases to -1 again when $u = M$. Thus, we see that the amplitude of the filter decreases as a function of distance from the origin of the centered filter, which is the characteristic of a lowpass filter. A similar argument is easily carried out when considering both variables simultaneously.

Problem 4.30

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 (as in Problem 4.28), but the squares, square root, or absolute values must be computed directly in the spatial domain.

Problem 4.31

We want to show that

$$\Im^{-1}\left[Ae^{-(\mu^2+\nu^2)/2\sigma^2}\right] = A2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2+z^2)}.$$

The explanation will be clearer if we start with one variable. We want to show that, if

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

then

$$\begin{aligned} h(t) &= \Im^{-1}\left[H(\mu)\right] \\ &= \int_{-\infty}^{\infty} e^{-\mu^2/2\sigma^2} e^{j2t\mu t}\, d\mu \\ &= \sqrt{2\pi}\sigma^{-2\pi^2\sigma^2 t^2}. \end{aligned}$$

We can express the integral in the preceding equations as

$$h(t) = \int_{-\infty}^{\infty} e^{-\frac{1}{2\sigma^2}[\mu^2 - j4\pi\sigma^2\mu t]} d\mu.$$

Making use of the identity

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

in the preceding integral yields

$$\begin{aligned} h(t) &= e^{-\frac{(2\pi)^2\sigma^2t^2}{2}}\int_{-\infty}^{\infty}e^{-\frac{1}{2\sigma^2}[\mu^2-j4\pi\sigma^2\mu t-(2\pi)^2\sigma^4t^2]}d\mu. \\ &= e^{-\frac{(2\pi)^2\sigma^2t^2}{2}}\int_{-\infty}^{\infty}e^{-\frac{1}{2\sigma^2}[\mu-j2\pi\sigma^2t]^2}d\mu. \end{aligned}$$

Next, we make the change of variables $r=\mu-j2\pi\sigma^2t$. Then, $dr=d\mu$ and the preceding integral becomes

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

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

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

The expression inside the brackets is recognized as the Gaussian probability density function whose value from -∞ to ∞ is 1. Therefore,

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

With the preceding results as background, we are now ready to show that

$$\begin{aligned} h(t,z) &= \Im^{-1}\left[Ae^{-(\mu^2+\nu^2)/2\sigma^2}\right] \\ &= A2\pi\sigma^2e^{-2\pi^2\sigma^2(t^2+z^2)}. \end{aligned}$$

By substituting directly into the definition of the inverse Fourier transform we have:

$$\begin{aligned} h(t,z) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}Ae^{-(\mu^2+\nu^2)/2\sigma^2}e^{j2\pi(\mu t+\nu z)}d\mu d\nu \\ &= \int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}Ae^{\left(-\frac{\mu^2}{2\sigma^2}+j2\pi\mu t\right)}d\mu\right]e^{\left(-\frac{\nu^2}{2\sigma^2}+j2\pi\nu z\right)}d\nu. \end{aligned}$$

The integral inside the brackets is recognized from the previous discussion to be equal to $A\sqrt{2\pi}\sigma e^{-2\pi^2\sigma^2 t^2}$. Then, the preceding integral becomes

$$h(t,z) = A\sqrt{2\pi}\sigma e^{-2\pi^2\sigma^2 t^2}\int_{-\infty}^{\infty} e^{\left(-\frac{v^2}{2\sigma^2}+j2\pi vz\right)}dv.$$

We now recognize the remaining integral to be equal to $\sqrt{2\pi}\sigma e^{-2\pi^2\sigma^2 z^2}$, from which we have the final result:

$$\begin{aligned} h(t,z) &= \left(A\sqrt{2\pi}\sigma e^{-2\pi^2\sigma^2 t^2}\right)\left(\sqrt{2\pi}\sigma e^{-2\pi^2\sigma^2 z^2}\right) \\ &= A2\pi\sigma^2 e^{-2\pi^2\sigma^2(t^2+z^2)}. \end{aligned}$$

Problem 4.35

With reference to Eq. (4.9-1), all the highpass filters in discussed in Section 4.9 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 4.37

(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 4.41

Because $M = 2^n$, we can write Eqs. (4.11-16) and (4.11-17) 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 4.11.3. 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. (4.11-14),

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

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. (4.11-16) is valid for all n.

From Eq. (4.11-17),

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

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.

Chapter 5

Problem Solutions

Problem 5.1

The solutions are shown in Fig. P5.1, from left to right.

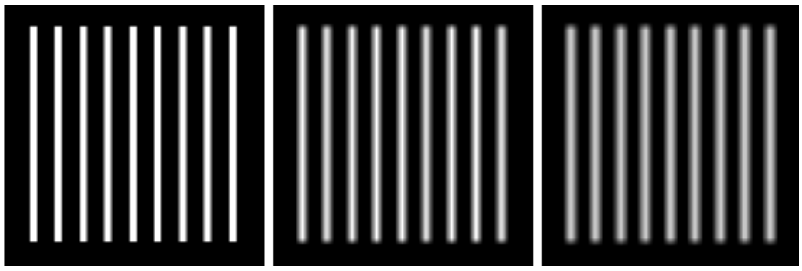


Figure P5.1

Problem 5.3

The solutions are shown in Fig. P5.3, from left to right.

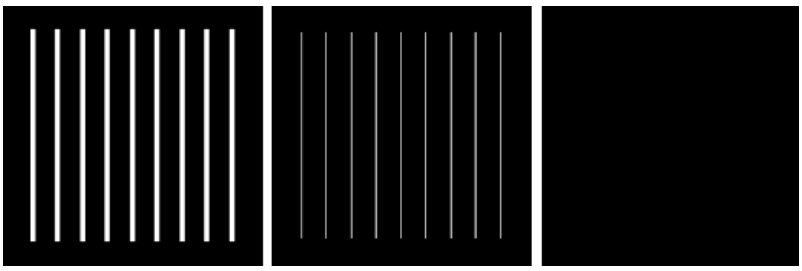


Figure P5.3

Problem 5.5

The solutions are shown in Fig. P5.5, from left to right.

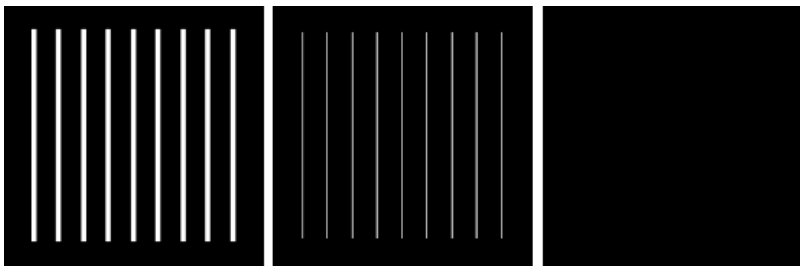


Figure P5.5

Problem 5.7

The solutions are shown in Fig. P5.7, from left to right.

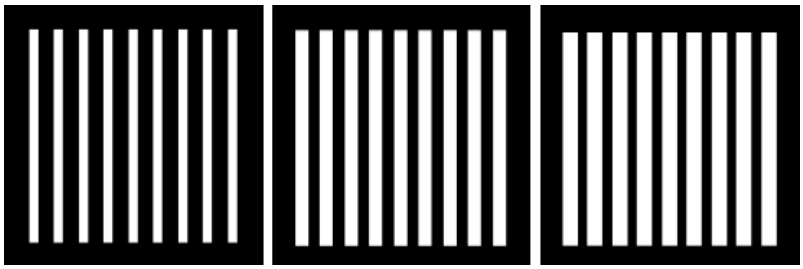


Figure P5.7

Problem 5.9

The solutions are shown in Fig. P5.9, from left to right.

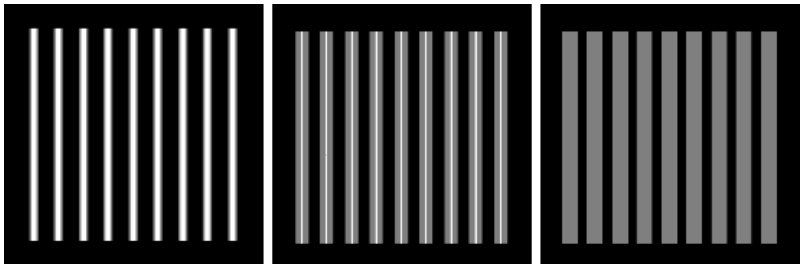


Figure P5.9

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:

$$H_{\mathrm{BP}}(u,v) = 1 - H_{\mathrm{BR}}(u,v).$$

Then:

(a) Ideal bandpass filter:

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

(b) Butterworth bandpass filter:

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

(c) Gaussian bandpass filter:

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

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 4.6.3 and Table 4.3. 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,

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

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

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

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

and

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

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 intensity value. Denote the Fourier transform of this image by $G(u,v)$. Because 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 intensity levels determined previously. We then construct a model of the cross hairs in the correct location (determined from the given image) using the dimensions provided and intensity level of the cross hairs. 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 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 plug in problem. Its purpose is to gain familiarity with the various terms of the Wiener filter. From Eq. (5.8-3),

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

where

$$\begin{aligned} |H(u,v)|^2 &= H^*(u,v)H(u,v) \\ &= H^2(u,v) \\ &= 64\pi^6\sigma^4(u^2+v^2)^2 e^{-4\pi^2\sigma^2(u^2+v^2)}. \end{aligned}$$

Then,

$$H_{\mathrm{W}}(u,v) = -\left[\frac{-8\pi^3\sigma^2(u^2+v^2)e^{-2\pi^2\sigma^2(u^2+v^2)}}{\left[64\pi^6\sigma^4(u^2+v^2)^2 e^{-4\pi^2\sigma^2(u^2+v^2)}\right] + K}\right].$$

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 (recall that the image and noise are assumed to be uncorrelated),

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

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 approximation $\hat{F}(u,v)$ gives the restored image for a coin. 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.

Problem 5.28

(b) The solution is shown in the following figure.The solutions are shown in Fig. P5.28. In each figure the horizontal axis is ρ and the vertical axis is θ, with $\theta = 0°$ at the bottom and going up to 180°. The fat lobes occur at 45° and the single point of intersection is at 135°. The intensity at that point is double the intensity of all other points.

Figure P5.28

Problem 5.30

(a) From Eq. (5.11-3),

$$\begin{aligned}
\Re\{f(x,y)\} = g(\rho,\theta) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(x\cos\theta + y\sin\theta - \rho)dxdy \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(x,y)\delta(x\cos\theta + y\sin\theta - \rho)dxdy \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} 1\times\delta(0-\rho)dxdy \\
&= \begin{cases} 1 & \text{if } \rho = 0 \\ 0 & \text{otherwise.} \end{cases}
\end{aligned}$$

where the third step follows from the fact that $\delta(x,y)$ is zero if x and/or y are not zero.

Problem 5.31

(a) From Section 2.6, we know that an operator, O, is linear if $O(af_1 + bf_2) = aO(f_1) + bO(f_2)$. From the definition of the Radon transform in Eq. (5.11-3),

$$\begin{aligned}
O(af_1 + bf_2) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (af_1 + bf_2)\delta(x\cos\theta + y\sin\theta - \rho)dxdy \\
&= a\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_1\delta(x\cos\theta + y\sin\theta - \rho)dxdy \\
&\qquad + b\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_2\delta(x\cos\theta + y\sin\theta - \rho)dxdy \\
&= aO(f_1) + bO(f_2)
\end{aligned}$$

thus showing that the Radon transform is a linear operation.

(c) From Chapter 4 (Problem 4.11), we know that the convolution of two function f and h is defined as

$$\begin{aligned} c(x,y) &= f(x,y)\bigstar h(x,y) \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)h(x-\alpha,y-\beta)d\alpha d\beta. \end{aligned}$$

We want to show that $\Re\{c\} = \Re\{f\}\bigstar\Re\{h\}$,where $\Re$ denotes the Radon transform. We do this by substituting the convolution expression into Eq. (5.11-3). That is,

$$\begin{aligned} \Re\{c\} &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\alpha,\beta)h(x-\alpha,y-\beta)d\alpha d\beta\right] \\ &\qquad \times\delta(x\cos\theta + y\sin\theta-\rho)dxdy \end{aligned}$$

$$\begin{aligned} = \int_{\alpha}\int_{\beta} f(\alpha,\beta) \\ \times\left[\int_x\int_y h(x-\alpha,y-\beta)\delta(x\cos\theta+y\sin\theta-\rho)dxdy\right]d\alpha d\beta \end{aligned}$$

where we used the subscripts in the integrals for clarity between the integrals and their variables. All integrals are understood to be between $-\infty$ and ∞. Working with the integrals inside the brackets with $x' = x-\alpha$ and $y' = y-\beta$ we have

$$\begin{aligned} &\textstyle\int_x\int_y h(x-\alpha,y-\beta)\delta(x\cos\theta+y\sin\theta-\rho)dxdy \\ &= \textstyle\int_{x'}\int_{y'} h(x',y')\delta(x'\cos\theta+y'\sin\theta-[\rho-\alpha\cos\theta-\beta\sin\theta])dx'dy' \\ &= \Re\{h\}(\rho-\alpha\cos\theta-\beta\sin\theta,\theta). \end{aligned}$$

We recognize the second integral as the Radon transform of h, but instead of being with respect to ρ and θ, it is a function of $\rho-\alpha\cos\theta-\beta\sin\theta$ and θ. The notation in the last line is used to indicate "the Radon transform of h as a function of $\rho-\alpha\cos\theta-\beta\sin\theta$ and θ." Then,

$$\begin{aligned} \Re\{c\} &= \int_{\alpha}\int_{\beta} f(\alpha,\beta) \\ &\qquad \times\left[\int_x\int_y h(x-\alpha,y-\beta)\delta(x\cos\theta+y\sin\theta-\rho)dxdy\right]d\alpha d\beta \\ &= \int_{\alpha}\int_{\beta} f(\alpha,\beta)\Re\{h\}(\rho-\rho',\theta)d\alpha d\beta \end{aligned}$$

where $\rho' = \alpha\cos\theta + \beta\sin\theta$. Then, based on the properties of the impulse, we can write

$$\Re\{h\}(\rho-\rho',\theta) = \int_{\rho'} \Re\{h\}(\rho-\rho',\theta)\delta(\alpha\cos\theta+\beta\sin\theta-\rho')d\rho'.$$

Then,

$$\Re\{c\} = \int_\alpha\int_\beta f(\alpha,\beta)\left[\Re\{h\}(\rho-\rho',\theta)\right]d\alpha d\beta$$

$$\begin{aligned}
&= \int_\alpha\int_\beta f(\alpha,\beta) \\
&\quad\times\left[\int_{\rho'}\Re\{h\}(\rho-\rho',\theta)\delta(\alpha\cos\theta+\beta\sin\theta-\rho')d\rho'\right]d\alpha d\beta \\
&= \int_{\rho'}\Re\{h\}(\rho-\rho',\theta)\left[\int_\alpha\int_\beta f(\alpha,\beta)\delta(\alpha\cos\theta+\beta\sin\theta-\rho')d\alpha d\beta\right]d\rho' \\
&= \int_{\rho'}\Re\{h\}(\rho-\rho',\theta)\Re\{f\}(\rho',\theta)d\rho' \\
&= \Re\{f\}\bigstar\Re\{h\}
\end{aligned}$$

where the fourth step follows from the definition of the Radon transform and the fifth step follows from the definition of convolution. This completes the proof.

Problem 5.33

The argument of function s in Eq.(5.11-24) may be written as:

$$r\cos(\beta+\alpha-\varphi)-D\sin\alpha = r\cos(\beta-\varphi)\cos\alpha-[r\sin(\beta-\varphi)+D]\sin\alpha.$$

From Fig. 5.47,

$$\begin{aligned}
R\cos\alpha' &= R+r\sin(\beta-\varphi) \\
R\sin\alpha' &= r\cos(\beta-\varphi).
\end{aligned}$$

Then, substituting in the earlier expression,

$$\begin{aligned}
r\cos(\beta+\alpha-\varphi)-R\sin\alpha &= R\sin\alpha'\cos\alpha-R\cos\alpha'\sin\alpha \\
&= R(\sin\alpha'\cos\alpha-\cos\alpha'\sin\alpha) \\
&= R\sin(\alpha'-\alpha)
\end{aligned}$$

which agrees with Eq. (5.11-25).

Chapter 6

Problem Solutions

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 that are sharply tuned to the wavelengths of the colors of the three objects. With a specific filter in place, only the objects whose color corresponds to that wavelength will produce a significant 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.

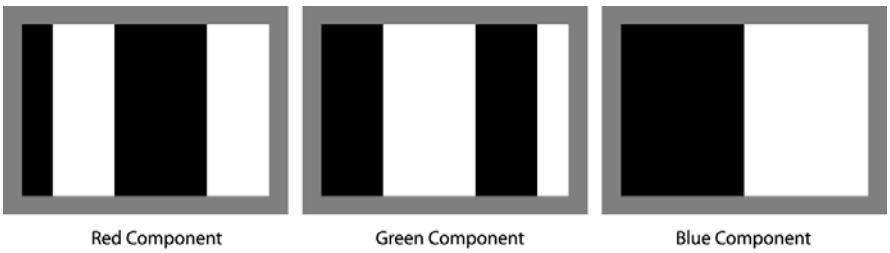


Figure P6.6

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 Table P6.6 with 0 and 255 representing black and white, respectively. Thus, we get the monochrome displays shown in Fig. 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

$$s_i = k\, r_i$$

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

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

so that

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

Problem 6.12

Using Eqs. (6.2-2) through (6.2-4), we get the results shown in Table P6.12. Note that, in accordance with Eq. (6.2-2), hue is undefined when $R = G = B$ since $\theta = \cos^{-1}(0/0)$. In addition, saturation is undefined when $R = G = B = 0$ since Eq. (6.2-3) yields $S = 1 - 3\min(0)/(3 \times 0) = 1 - (0/0)$. Thus, we get the monochrome display shown in Fig. 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

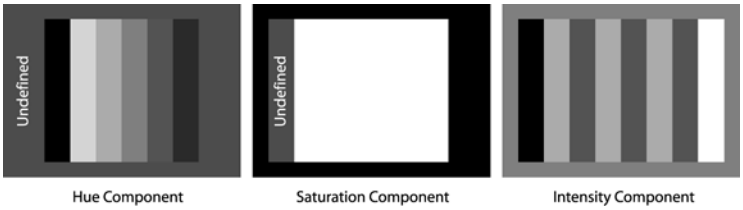


Figure P6.12

Problem 6.14

There are two important aspects to this problem. One is to approach it in the 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. Then the angle 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. Because 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 shades of red and 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

$$\begin{aligned} r_i' &= 1 - r_i \\ s_i' &= 1 - s_i. \end{aligned}$$

Thus,

$$r_i = 1 - r_i'$$

and

$$s_i' = 1 - s_i = 1 - (1 - r_i) = 1 - \left(1 - \left(1 - r_i'\right)\right)$$

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 simplest approach conceptually 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 six 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) **z** in RGB space lying on the plane. If an RGB point **z** does not lie on the plane, and its coordinates are substituted in the preceding equation, the equation will give either a positive or a negative value; it will not yield zero. We say that **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 **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, **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 if it is on a plane) 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.

Chapter 7

Problem Solutions

Problem 7.2

A mean approximation pyramid is created 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

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

Pixel replication is used in the generation of the complementary prediction residual pyramid. 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 approxima-

tion (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 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 where $N = 2^J$ is bounded by $\frac{4}{3}N^2$ or $\frac{4}{3}\left(2^J\right)^2 = \frac{4}{3}2^{2J}$ (see Section 7.1.1):

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

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

J	Pyramid Elements	Compression Ratio
0	1	1
1	5	$5/4 = 1.25$
2	21	$21/16 = 1.3125$
3	85	$85/64 = 1.328$
$\vdots$	$\vdots$	$\vdots$
∞		$4/3 = 1.33$

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

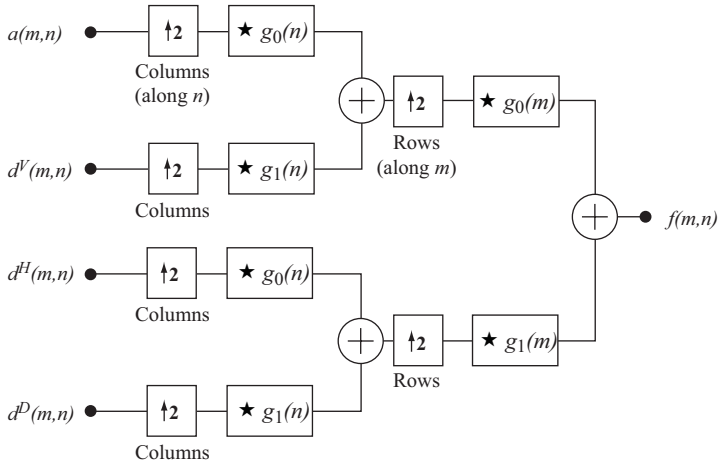


Figure P7.7

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 Fig. P7.7 shows.

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 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 $\psi_{3,3}(x) = 2\sqrt{2}\psi(8x-3)$ from above, substitution gives

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

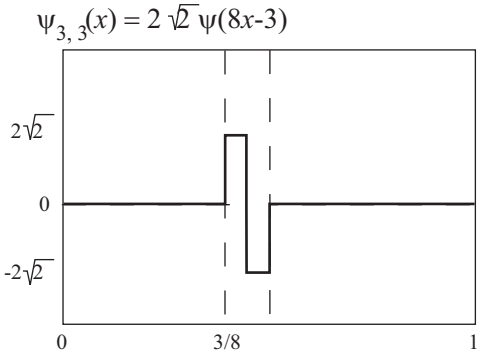


Figure P7.13

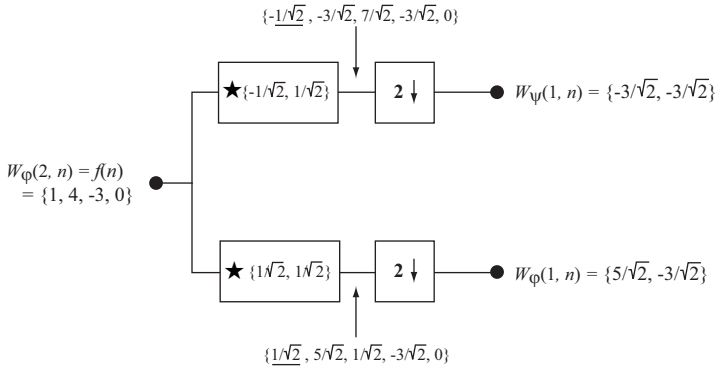


Figure P7.19

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 7.17.

Problem 7.19

The filter bank is the first bank in Fig. 7.19, as shown in Fig. 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. 7.10(a), we will use the FWT^{-1} 2-d synthesis bank of Fig. 7.24(c). First, place the 64×64 approximation "coefficients" from Fig. 7.10(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. 7.10(a), to drive the filter bank inputs in Fig. 7.24(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. 7.10(a) to drive the synthesis filter bank in Fig. 7.24(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. 7.24(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 one 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.

Chapter 8

Problem Solutions

Problem 8.4

(a) Table P8.4 shows the starting intensity values, their 8-bit codes, the IGS sum used in each step, the 4-bit IGS code and its equivalent decoded value (the decimal equivalent of the IGS code multiplied by 16), the error between the decoded IGS intensities and the input values, and the squared error.

(b) Using Eq. (8.1-10) and the squared error values from Table P8.4, the rms error is

$$\begin{aligned} e_{rms} &= \sqrt{\frac{1}{8}(144+25+49+16+16+169+64+9)} \\ &= \sqrt{\frac{1}{8}(492)} \\ &= 7.84 \end{aligned}$$

or about 7.8 intensity levels. From Eq. (8.1-11), the signal-to-noise ratio is

$$\begin{aligned} SNR_{ms} &= \frac{96^2+144^2+128^2+240^2+176^2+160^2+64^2+96^2}{492} \\ &= \frac{173824}{492} \\ &\simeq 353. \end{aligned}$$

Table P8.4

Intensity	8-bit Code	Sum	IGS Code	Decoded IGS	Error	Square Error
		00000000				
108	01101100	01101100	0110	96	-12	144
139	10001011	10010111	1001	144	5	25
135	10000111	10001110	1000	128	-7	49
244	11110100	11110100	1111	240	-4	16
172	10101100	10110000	1011	176	4	16
173	10101101	10101101	1010	160	-13	169
56	00111000	01000101	0100	64	8	64
99	01100011	01101000	0110	96	-3	9

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 $\left\{a_1, a_2, ..., a_q\right\}$ with probabilities $\left[P(a_1), P(a_2), ..., P\left(a_q\right)\right]^T$. Then, using Eq. (8.1-6) and the fact that the sum of all $P(a_i)$ is 1, we get

$$\begin{aligned}
\log q - H &= \log q \left[\sum_{j=1}^{q} P\left(a_j\right)\right] + \sum_{j=1}^{q} P\left(a_j\right) \log P\left(a_j\right) \\
&= \sum_{j=1}^{q} P\left(a_j\right) \log q + \sum_{j=1}^{q} P\left(a_j\right) \log P\left(a_j\right) \\
&= \sum_{j=1}^{q} P\left(a_j\right) \log q P\left(a_j\right).
\end{aligned}$$

Using the log relationship from Problem 8.6, this becomes

$$= \log e \sum_{j=1}^{q} P\left(a_j\right) \ln q P\left(a_j\right).$$

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,

$$\begin{aligned}
\log q - H &\geq \log e \sum_{ij=1}^{q} P\left(a_j\right)\left[1-\frac{1}{qP\left(a_j\right)}\right] \\
&\geq \log e\left[\sum_{j=1}^{q} P\left(a_j\right) - \frac{1}{q}\sum_{j=1}^{q}\frac{P\left(a_j\right)}{P\left(a_j\right)}\right] \\
&\geq \log e\,[1-1] \\
&\geq 0
\end{aligned}$$

so that

$$\log q \geq H.$$

Therefore, H 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 $P(a_j) = 1/q$ for all j.

Problem 8.9

(d) We can compute the relative frequency of pairs of pixels by assuming that the image is connected from line to line and end to beginning. The resulting probabilities are listed in Table P8.9-2.

Table P8.9-2

Intensity pair	Count	Probability
(21, 21)	8	1/4
(21, 95)	4	1/8
(95, 169)	4	1/8
(169, 243)	4	1/8
(243, 243)	8	1/4
(243, 21)	4	1/8

The entropy of the intensity pairs is estimated using Eq. (8.1-7) and dividing by 2 (because the pixels are considered in pairs):

$$\begin{aligned}
\frac{1}{2}\tilde{H} &= -\frac{1}{2}\left[\frac{1}{4}\log_2\frac{1}{4}+\frac{1}{8}\log_2\frac{1}{8}+\frac{1}{8}\log_2\frac{1}{8}+\frac{1}{8}\log_2\frac{1}{8}+\frac{1}{4}\log_2\frac{1}{4}+\frac{1}{8}\log_2\frac{1}{8}\right] \\
&= \frac{2.5}{2} \\
&= 1.25\,\text{bits/pixel.}
\end{aligned}$$

The difference between this value and the entropy in (a) tells us that a mapping can be created to eliminate $(1.811 - 1.25) = 0.56$ bits/pixel of spatial redundancy.

Problem 8.15

To decode $G^k_{exp}(n)$:

1. Count the number of 1s in a left-to-right scan of a concatenated $G^k_{exp}(n)$ bit sequence before reaching the first 0, and let i be the number of 1s counted.

2. Get the $k+i$ bits following the 0 identified in step 1 and let d be its decimal equivalent.

3. The decoded integer is then

$$d + \sum_{j=0}^{i-1} 2^{j+k}.$$

For example, to decode the first $G^2_{exp}(n)$ code in the bit stream 10111011..., let $i = 1$, the number of 1s in a left-to-right scan of the bit stream before finding the first 0. Get the $2 + 1 = 3$ bits following the 0, that is, 111 so $d = 7$. The decoded integer is then

$$7 + \sum_{j=0}^{1-1} 2^{j+2} = 7 + 2^2 = 11.$$

Repeat the process for the next code word, which begins with the bit sequence 011...

Problem 8.18

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 P8.18. 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!".

Table P8.18

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)

Problem 8.20

The input to the LZW decoding algorithm in Example 8.7 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 intensity 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 P8.20.

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

39 39 126 126
39 39 126 126
39 39 126 126
39 39 126 126

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

Table P8.20

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 8.24

(a) - **(b)** Following the procedure outlined in Section 8.2.8, we obtain the results shown in Table P8.24.

Table P8.24

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.

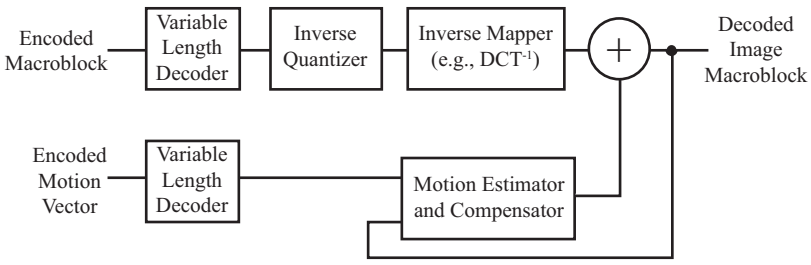


Figure P8.27

Problem 8.29

The derivation proceeds by substituting the uniform probability function into Eqs. (8.2-57) - (8.2-59) and solving the resulting simultaneous equations with $L=4$. Equation (8.2-58) yields

$$\begin{aligned} s_0 &= 0 \\ s_1 &= \tfrac{1}{2}(t_1+t_2) \\ s_2 &= \infty. \end{aligned}$$

Substituting these values into the integrals defined by Eq. (8.2-57), we get two equations. The first is (assuming $s_1 \le A$)

$$\int_{s_0}^{s_1} (s-t_1)\,p(s)\,ds = 0$$

$$\frac{1}{2A}\int_0^{\frac{1}{2}(t_1+t_2)} (s-t_1)\,ds = \frac{s^2}{2} - t_1 s \Bigg|_{0}^{\frac{1}{2}(t_1+t_2)} = 0$$

$$\begin{aligned} (t_1+t_2)^2 - 4t_1(t_1+t_2) &= 0 \\ (t_1+t_2)(t_2-3t_1) &= 0 \end{aligned}$$

so

$$\begin{aligned} t_1 &= -t_2 \\ t_2 &= 3t_1. \end{aligned}$$

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 $p(s)$)

$$\int_{s_1}^{s_2} (s-t_2)\,p(s)\,ds = 0$$

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

$$4A^2 - 8At_2 - (t_1+t_2)^2 - 4t_2(t_1+t_2) = 0.$$

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

$$\begin{aligned} 8t_1^2 - 6At_1 + A^2 &= 0 \\ \left[t_1 - \frac{A}{2}\right](8t_1 - 2A) &= 0 \\ t_1 &= \frac{A}{2} \\ t_1 &= \frac{A}{4}. \end{aligned}$$

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

$$\begin{array}{llll} t_2 = \frac{3A}{2} & \text{and} & s_1 = A & \text{for} \quad t_1 = \frac{A}{2} \\ t_2 = \frac{3A}{4} & \text{and} & s_1 = \frac{A}{2} & \text{for} \quad t_1 = \frac{A}{4}. \end{array}$$

Because $s_1 = A$ 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,

$$\begin{array}{llll} s_0 = 0 & s_1 = \frac{A}{2} & s_2 = \infty \\ t_1 = \frac{A}{4} & t_2 = \frac{3A}{4} & . \end{array}$$

Problem 8.34

A variety of methods for inserting invisible watermarks into the DFT coefficients of an image have been reported in the literature. Here is a simplified outline of one in which watermark insertion is done as follows:

1. Create a watermark by generating a P-element pseudo-random sequence of numbers, $\omega_1, \omega_2, ..., \omega_P$, taken from a Gaussian distribution with zero mean and unit variance.

2. Compute the DFT of the image to be watermarked. We assume that the transform has not been centered by pre-multiplying the image by $(-1)^{x+y}$.

3. Choose $\frac{P}{2}$ coefficients from each of the four quadrants of the DFT in the middle frequency range. This is easily accomplished by choosing coefficients in the order shown in Fig. P8.34 and skipping the first K coefficients (the low frequency coefficients) in each quadrant.

4. Insert the first half of the watermark into the chosen DFT coefficients, c_i for $1 \leq i \leq \frac{P}{2}$, in quadrants I and III of the DFT using

$$c_i' = c_i(1 + \alpha\omega_i)$$

5. Insert the second half of the watermark into the chosen DFT coefficients of quadrants II and IV of the DFT in a similar manner. Note that this process maintains the symmetry of the transform of a real-valued image. In addition, constant α determines the strength of the inserted watermark.

6. Compute the inverse DFT with the watermarked coefficients replacing the unmarked coefficients.

Watermark extraction is performed as follows:

1. Locate the DFT coefficients containing the watermark by following the insertion process in the embedding algorithm.

2. Compute the watermark $\hat{\omega}_1, \hat{\omega}_2, ..., \hat{\omega}_P$ using

$$\hat{\omega}_i = \hat{c}_i - c_i$$

3. Compute the correlation between ω and $\hat{\omega}$and compare to a pre-determined threshold T to determine if the mark is present.

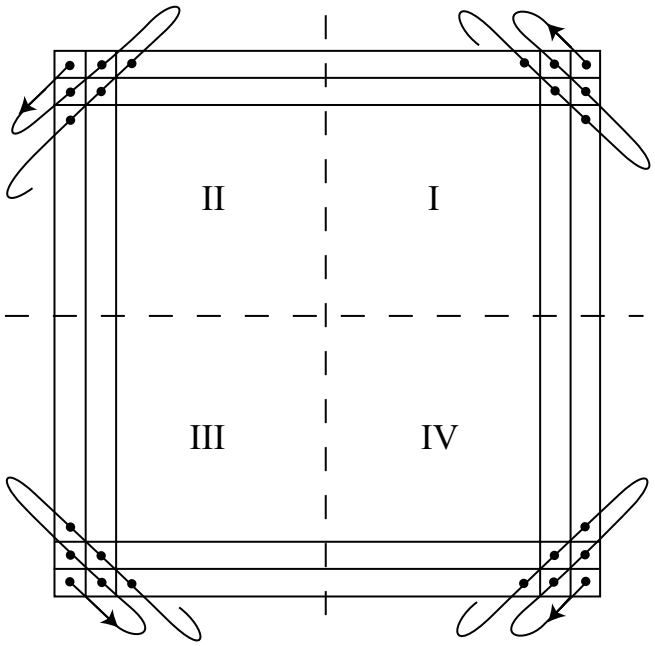


Figure P8.34

Chapter 9

Problem Solutions

Problem 9.2

(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. P9.2(a). 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 containing the boundary.

Problem 9.4

(a) Erosion is set intersection. The intersection of two convex sets is convex also.

(b) See Fig. P9.4(a). Keep in mind that the digital sets in question are the larger black dots. The lines are shown for convenience in visualizing what the continu-

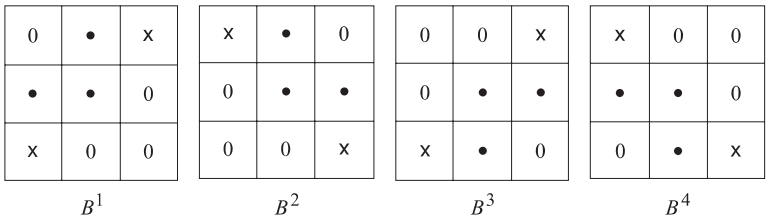


Figure P9.2

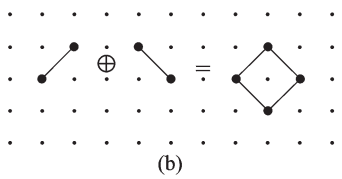


Figure P9.4

ous sets would be, they are not part of the sets being considered here. The result of dilation in this case is not convex because the center point is not in the set.

Problem 9.5

Refer to Fig. P9.5. The center of each structuring element is shown as a black dot.

(a) This solution was obtained by eroding the original set (shown dashed) with the structuring element shown (note that the origin is at the bottom, right).

(b) This solution was obtained by eroding the original set with the tall rectangular structuring element shown.

(c) This solution was obtained by first eroding the image shown down to two vertical lines using the rectangular structuring element (note that this elements is slightly taller than the center section of the "U" figure). This result was then dilated with the circular structuring element.

(d) This solution was obtained by first dilating the original set with the large disk shown. The dilated image was eroded with a disk whose diameter was equal to one-half the diameter of the disk used for dilation.

Problem 9.7

(a) The dilated image will grow without bound.

(b) A one-element set (i.e., a one-pixel image).

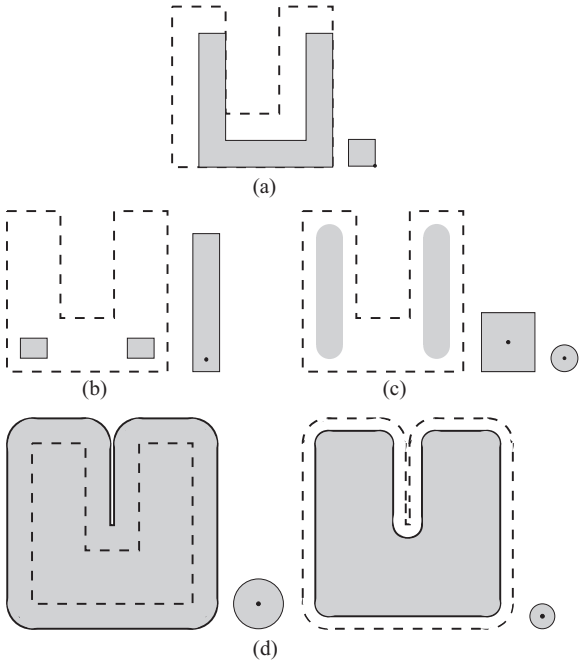


Figure P9.5

Problem 9.9

The proof, which consists of showing that the expression

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

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 $b \in B$.

Problem 9.11

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 9.14

Starting with the definition of closing,

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

The proof of the other duality property follows a similar approach.

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. 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, a basic two-step approach for differentiating between the three shapes is as follows:

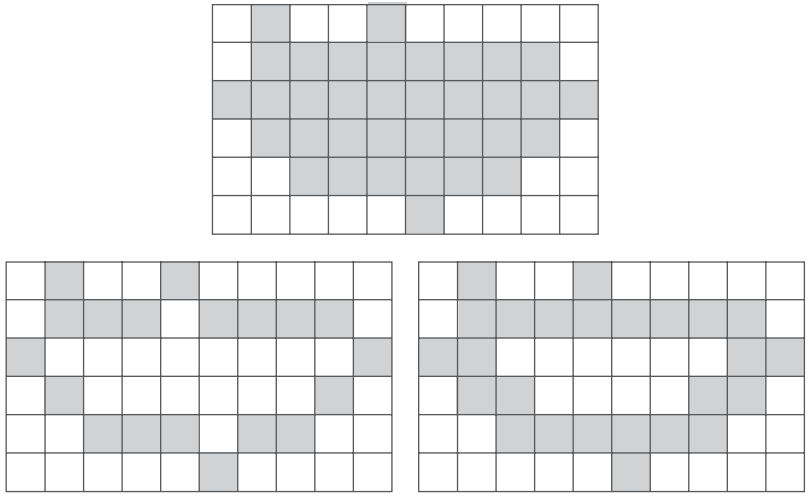


Figure P9.22

Step 1. Apply an end-point detector to the object. If no end points are found, the object is a Lake. Otherwise it is a Bay or a Line.

Step 2. There are numerous ways to differentiate between a Bay 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. 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, 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).

Problem 9.23

(a) If the spheres are not allowed to touch, the solution of the problem starts by determining which points are background (black) points. To do this, we pick a black point on the boundary of the image and determine all black points connected to it using a connected component algorithm(Section 9.5.3). These con-

nected components are labels with a value different from 1 or 0. The remaining black points are interior to spheres. We can fill all spheres with white by applying the hole filling algorithm in Section 9.5.2 until all interior black points have been turned into white points. The alert student will realize that if the interior points are already known, they can all be turned simply into white points thus filling the spheres without having to do region filling as a separate procedure.

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 connected component 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 9.27

Erosion is the set of points z such that B, translated by z, is contained in A. If B is a single point, this definition will be satisfied only by the points comprising A, so erosion of A by B is simply A. Similarly, dilation is the set of points z such that $\hat{B}$ ($\hat{B} = B$ in this case), translated by z, overlaps A by at least one point. Because B is a single point, the only set of points that satisfy this definition is the set of points comprising A, so the dilation of A by B is A.

Problem 9.29

Consider first the case for $n = 1$:

$$\begin{aligned} E_G^{(1)}(F) &= \left[\left[E_G^{(1)}(F)\right]^c\right]^c \\ &= \left[[(F \ominus B) \cup G]^c\right]^c \\ &= \left[(F \ominus B)^c \cap G^c\right]^c \\ &= \left[\left(F^c \oplus \hat{B}\right) \cap G^c\right]^c \\ &= \left[\left(F^c \oplus B\right) \cap G^c\right]^c \\ &= \left[D_{G^c}^{(1)}(F^c)\right]^c \end{aligned}$$

where the third step follows from DeMorgan's law, $(A \cup B)^c = A^c \cap B^c$, the fourth step follows from the duality property of erosion and dilation (see Section 9.2.3), the fifth step follows from the symmetry of the SE, and the last step follows from the definition of geodesic dilation. The next step, $E_G^{(2)}(F)$, would involve the geodesic erosion of the above result. But that result is simply a set, so we could obtain it in terms of dilation. That is, we would complement the result just mentioned, complement G, compute the geodesic dilation of size 1 of the two, and complement the result. Continuing in this manner we conclude that

$$\begin{aligned} E_G^{(n)} &= \left[D_{G^c}^{(1)}\left(\left[E_G^{(n-1)}(F)\right]^c\right)\right]^c \\ &= \left[D_{G^c}^{(1)}\left(D_{G^c}^{(n-1)}\left(F^c\right)\right)\right]^c. \end{aligned}$$

Similarly,

$$\begin{aligned} D_G^{(1)}(F) &= \left[\left[D_G^{(1)}(F)\right]^c\right]^c \\ &= \left[[(F \oplus B) \cap G]^c\right]^c \\ &= \left[(F \oplus B)^c \cup G^c\right]^c \\ &= \left[\left(F^c \ominus \hat{B}\right) \cup G^c\right]^c \\ &= \left[\left(F^c \ominus B\right) \cup G^c\right]^c \\ &= \left[E_{G^c}^{(1)}(F^c)\right]^c. \end{aligned}$$

As before,

$$\begin{aligned} D_G^{(n)} &= \left[E_{G^c}^{(1)}\left(\left[D_G^{(n-1)}(F)\right]^c\right)\right]^c \\ &= \left[E_{G^c}^{(1)}\left(E_{G^c}^{(n-1)}\left(F^c\right)\right)\right]^c. \end{aligned}$$

Problem 9.31

(a) Consider the case when $n = 2$

$$\begin{aligned} [(F \ominus 2B)]^c &= [(F \ominus B) \ominus B]^c \\ &= (F \ominus B)^c \oplus \hat{B} \\ &= \left(F^c \oplus \hat{B}\right) \oplus \hat{B} \\ &= \left(F^c \oplus 2\hat{B}\right) \end{aligned}$$

where the second and third lines follow from the duality property in Eq. (9.2-5). For an arbitrary number of erosions,

$$\begin{aligned}[(F \ominus nB)]^c &= [(F \ominus (n-1)B) \ominus B]^c \\ &= [(F \ominus (n-1)B)]^c \oplus \hat{B}\end{aligned}$$

which, when expanded, will yield $[(F \ominus nB)]^c = F^c \oplus n\hat{B}$.

(b) Proved in a similar manner.

Problem 9.33

(a) From Eq. (9.6-1),

$$\begin{aligned}(f \ominus b)^c &= \left[\min_{(s,t)\in b}\{f(x+s,y+t)\}\right]^c \\ &= \left[-\max_{(s,t)\in b}\{-f(x+s,y+t)\}\right]^c \\ &= \max_{(s,t)\in b}\{-f(x+s,y+t)\} \\ &= -f \oplus \hat{b} \\ &= f^c \oplus \hat{b}.\end{aligned}$$

The second step follows from the definition of the complement of a gray-scale function; that is, the minimum of a set of numbers is equal to the negative of the maximum of the negative of those numbers. The third step follows from the definition of the complement. The fourth step follows from the definition of gray-scale dilation in Eq. (9.6-2), using the fact that $\hat{b}(x,y) = b(-x-y)$. The last step follows from the definition of the complement, $-f = f^c$. The other duality property is proved in a similar manner.

(c) We prove the first duality property. Start with the a geodesic dilation of size 1:

$$\begin{aligned}D_g^{(1)}(f) &= \left[\left[D_g^{(1)}(f)\right]^c\right]^c \\ &= \left[[(f \oplus b) \wedge g]^c\right]^c \\ &= \left[[-(-(f \oplus b) \vee -g)]^c\right]^c \\ &= [-(f \oplus b) \vee -g]^c \\ &= [(f \oplus b)^c \vee g^c]^c \\ &= [(f^c \ominus b) \vee g^c]^c \\ &= \left[E_{g^c}^{(1)}(f^c)\right]^c.\end{aligned}$$

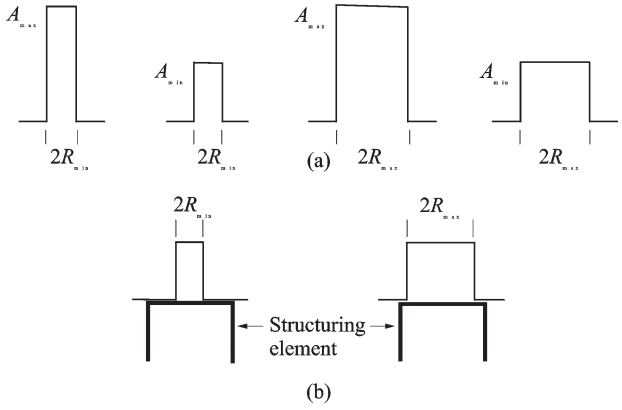


Figure P9.35

The second step follows from the definition of geodesic dilation. The third step follows from the fact that the point-wise minimum of two sets of numbers is the negative of the point-wise maximum of the two numbers. The fourth and fifth steps follow from the definition of the complement. The sixth step follows from the duality of dilation and erosion (we used the given fact that $\hat{b} = b$). The last step follows from the definition of geodesic erosion.

The next step in the iteration, $D_g^{(2)}(f)$, would involve the geodesic dilation of size 1 of the preceding result. But that result is simply a set, so we could obtain it in terms of erosion. That is, we would complement the result just mentioned, complement g, compute the geodesic erosion of the two, and complement the result. Continuing in this manner we conclude that

$$D_g^{(n)}(f) = \left[E_{g^c}^{(1)}\left(E_{g^c}^{(n-1)}\left(f^c\right)\right)\right]^c.$$

The other property is proved in a similar way.

Problem 9.35

(a) The noise spikes are of the general form shown in Fig. P9.35(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. P9.35(b). Note that the shape of the structuring element is matched to the known shape of the noise spikes.

Problem 9.36

(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 (Section 9.5.3). All connected components that contain elements from B are particles that have merged with the border of the image.

Chapter 10

Problem Solutions

Problem 10.1

Expand $f(x+\Delta x)$ into a Taylor series about x:

$$f(x+\Delta x) = f(x) + \Delta x f'(x) + \frac{(\Delta x)}{2!} f''(x) + \cdots$$

The increment in the spatial variable x is defined in Section 2.4.2 to be 1, so by letting $\Delta x = 1$ and keeping only the linear terms we obtain the result

$$f'(x) = f(x+1) - f(x)$$

which agrees with Eq. (10.2-1).

Problem 10.2

The masks would have the coefficients shown in Fig. P10.2. 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.

Problem 10.4

(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.

0	0	0	0	1	0	0	0	1	1	0	0
1	−2	1	0	−2	0	0	−2	0	0	−2	0
0	0	0	0	1	0	1	0	0	0	0	1
Horizontal			Vertical			+45°			−45°		

Figure P10.2

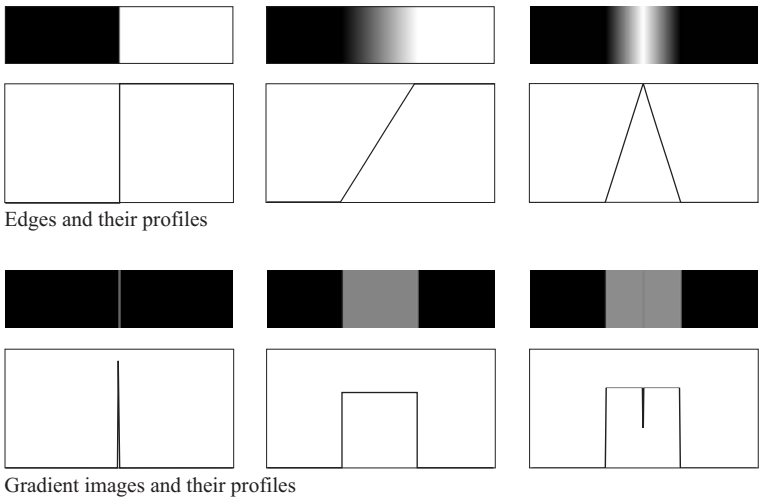


Figure P10.5

Problem 10.5

(a) The first row in Fig. P10.5 shows a step, ramp, and edge image, and horizontal profiles through their centers. Similarly, the second row shows the corresponding gradient images and horizontal profiles through their centers. The thin dark borders in the images are included for clarity in defining the borders of the images; they are not part of the image data.

Problem 10.7

Figure P10.7 shows the solution.

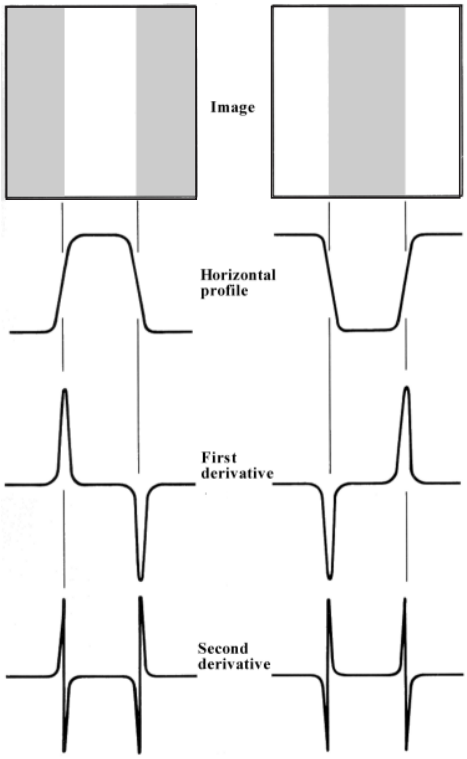


Figure P10.7

Problem 10.9

Consider first the Sobel masks of Figs. 10.14 and 10.15. A simple 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. P10.9, 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 P10.9, 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.

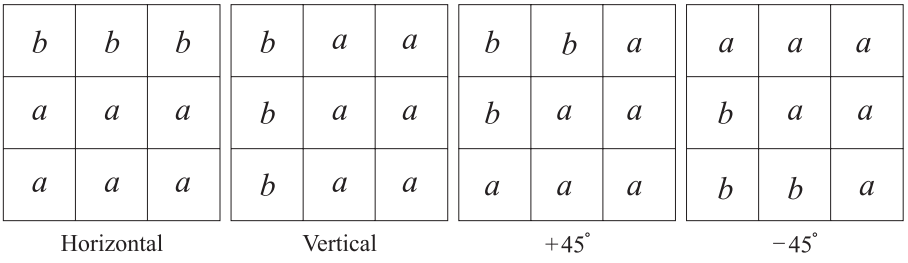


Figure P10.9

Table P10.9

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	$4a-4b$

Problem 10.11

(a) The operators are as follows (negative numbers are shown underlined):

$$\begin{array}{cccccccc} 111 & 110 & 10\underline{1} & 0\underline{11} & \underline{111} & \underline{11}0 & \underline{1}01 & 011 \\ 000 & 10\underline{1} & 10\underline{1} & 10\underline{1} & 000 & \underline{1}01 & \underline{1}01 & \underline{1}01 \\ \underline{111} & 0\underline{11} & 10\underline{1} & 110 & 111 & 011 & \underline{1}01 & \underline{11}0 \end{array}$$

Problem 10.13

(a) The local average at a point (x,y) in an image is given by

$$\bar{f}(x,y)=\frac{1}{n^2}\sum_{z_i\in S_{xy}} z_i$$

where S_{xy} is the region in the image encompassed by the $n\times n$ averaging mask when it is centered at (x,y) and the z_i are the intensities of the image pixels in that region. The partial

$$\partial\bar{f}/\partial x=\bar{f}(x+1,y)-\bar{f}(x,y)$$

is thus given by

$$\partial \bar{f}/\partial x = \frac{1}{n^2}\sum_{z_i \in S_{x+1,y}} z_i - \frac{1}{n^2}\sum_{z_i \in S_{xy}} z_i$$

The first summation on the right can be interpreted as consisting of all the pixels in the second summation minus the pixels in the first row of the mask, plus the row picked up by the mask as it moved from (x,y) to $(x+1,y)$. Thus, we can write the preceding equation as

$$\partial \bar{f}/\partial x = \frac{1}{n^2}\sum_{z_i \in S_{x+1,y}} z_i - \frac{1}{n^2}\sum_{z_i \in S_{xy}} z_i$$

$$\begin{aligned}
&= \left[\left(\frac{1}{n^2}\sum_{z_i \in S_{xy}} z_i\right) + \frac{1}{n^2}(\text{sum of pixels in new row})\right. \\
&\qquad \left. -\frac{1}{n^2}(\text{sum of pixels in 1st row})\right] - \frac{1}{n^2}\sum_{z_i \in S_{xy}} z_i \\
&= \frac{1}{n^2}\sum_{k=y-\frac{n-1}{2}}^{y+\frac{n-1}{2}} f(x+\frac{n+1}{2},k) - \frac{1}{n^2}\sum_{k=y-\frac{n-1}{2}}^{y+\frac{n-1}{2}} f(x-\frac{n-1}{2},k) \\
&= \frac{1}{n^2}\left[\sum_{k=y-\frac{n-1}{2}}^{y+\frac{n-1}{2}} f(x+\frac{n+1}{2},k) - f(x-\frac{n-1}{2},k)\right].
\end{aligned}$$

This expression gives the value of $\partial \bar{f}/\partial x$ at coordinates (x,y) of the *smoothed* image. Similarly,

$$\begin{aligned}
\partial \bar{f}/\partial y &= \frac{1}{n^2}\sum_{z_i \in S_{x,y+1}} z_i - \frac{1}{n^2}\sum_{z_i \in S_{xy}} z_i \\
&= \left[\left(\frac{1}{n^2}\sum_{z_i \in S_{xy}} z_i\right) + \frac{1}{n^2}(\text{sum of pixels in new col})\right. \\
&\qquad \left. -\frac{1}{n^2}(\text{sum of pixels in 1st col})\right] - \frac{1}{n^2}\sum_{z_i \in S_{xy}} z_i \\
&= \frac{1}{n^2}\sum_{k=x-\frac{n-1}{2}}^{x+\frac{n-1}{2}} f(k,y+\frac{n+1}{2}) - \frac{1}{n^2}\sum_{k=x-\frac{n-1}{2}}^{x+\frac{n-1}{2}} f(k,y-\frac{n-1}{2}) \\
&= \frac{1}{n^2}\left[\sum_{k=x-\frac{n-1}{2}}^{x+\frac{n-1}{2}} f(k,y+\frac{n+1}{2}) - f(k,y-\frac{n-1}{2})\right].
\end{aligned}$$

The edge magnitude image corresponding to the smoothed image $\bar{f}(x,y)$ is then given by

$$\bar{M}(x,y) = \sqrt{(\partial \bar{f}/\partial x)^2 + (\partial \bar{f}/\partial y)^2}.$$

Problem 10.14

(a) We proceeed as follows

$$\begin{aligned}
\text{Average}\left[\nabla^2 G(x,y)\right] &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \nabla^2 G(x,y)\,dx\,dy \\
&= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \left[\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}\,dx\,dy \\
&= \frac{1}{\sigma^4}\int_{-\infty}^{\infty} x^2 e^{-\frac{x^2}{2\sigma^2}}\,dx \int_{-\infty}^{\infty} e^{-\frac{y^2}{2\sigma^2}}\,dy \\
&\quad + \frac{1}{\sigma^4}\int_{-\infty}^{\infty} y^2 e^{-\frac{y^2}{2\sigma^2}}\,dy \int_{-\infty}^{\infty} e^{-\frac{x^2}{2\sigma^2}}\,dx \\
&\quad - \frac{2}{\sigma^2}\int_{-\infty}^{\infty} e^{-\frac{x^2+y^2}{2\sigma^2}}\,dx\,dy \\
&= \frac{1}{\sigma^4}\left(\sqrt{2\pi}\sigma \times \sigma^2\right)\left(\sqrt{2\pi}\sigma\right) \\
&\quad + \frac{1}{\sigma^4}\left(\sqrt{2\pi}\sigma \times \sigma^2\right)\left(\sqrt{2\pi}\sigma\right) \\
&\quad - \frac{2\left(2\pi\sigma^2\right)}{\sigma^2} \\
&= 4\pi - 4\pi \\
&= 0
\end{aligned}$$

the fourth line follows from the fact that

$$\text{variance}(z) = \sigma^2 = \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{\infty} z^2 e^{-\frac{z^2}{2\sigma^2}}\,dz$$

and

$$\frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{\infty} e^{-\frac{z^2}{2\sigma^2}}\,dz = 1.$$

Problem 10.15

(b) The answer is yes for functions that meet certain mild conditions, and if the zero crossing method is based on rotational operators like the LoG func-

tion and a threshold of 0. Geometrical properties of zero crossings in general are explained in some detail in the paper "On Edge Detection," by V. Torre and T. Poggio, *IEEE Trans. Pattern Analysis and Machine Intell.*, vol. 8, no. 2, 1986, pp. 147-163. Looking up this paper and becoming familiar with the mathematical underpinnings of edge detection is an excellent reading assignment for graduate students.

Problem 10.18

(a) Equation (10.2-21) can be written in the following separable form

$$\begin{aligned} G(x,y) &= e^{-\frac{x^2+y^2}{2\sigma^2}} \\ &= e^{-\frac{x^2}{2\sigma^2}} e^{-\frac{y^2}{2\sigma^2}} \\ &= G(x)G(y). \end{aligned}$$

From Eq. (3.4-2) and the preceding equation, the convolution of $G(x,y)$ and $f(x,y)$ can be written as

$$\begin{aligned} G(x,y)\star f(x,y) &= \sum_{s=-a}^{a}\sum_{t=-a}^{a} G(s,t)f(x-s,y-t) \\ &= \sum_{s=-a}^{a}\sum_{t=-a}^{a} e^{-\frac{s^2}{2\sigma^2}} e^{-\frac{t^2}{2\sigma^2}} f(x-s,y-t) \\ &= \sum_{s=-a}^{a} e^{-\frac{s^2}{2\sigma^2}} \left[\sum_{t=-a}^{a} e^{-\frac{t^2}{2\sigma^2}} f(x-s,y-t)\right] \end{aligned}$$

where $a=(n-1)/2$ and n is the size of the $n\times n$ mask obtained by sampling Eq. (10.2-21). The expression inside the brackets is the 1-D convolution of the exponential term, $e^{-t^2/2\sigma^2}$, with the rows of $f(x,y)$. Then the outer summation is the convolution of $e^{-s^2/2\sigma^2}$ with the columns of the result. Stated another way,

$$G(x,y)\star f(x,y) = G(x)\star\left[G(y)\star f(x,y)\right].$$

Problem 10.19

(a) As Eq. (10.2-25) shows, the first two steps of the algorithm can be summarized into one equation:

$$g(x,y) = \nabla^2[G(x,y)\star f(x,y)].$$

Using the definition of the Laplacian operator we can express this equation as

$$\begin{aligned} g(x,y) &= \frac{\partial^2}{\partial x^2}\left[G(x,y)\bigstar f(x,y)\right] + \frac{\partial^2}{\partial y^2}\left[G(x,y)\bigstar f(x,y)\right] \\ &= \frac{\partial^2}{\partial x^2}\left[G(x)\bigstar G(y)\bigstar f(x,y)\right] + \frac{\partial^2}{\partial y^2}\left[G(x)\bigstar G(y)\bigstar f(x,y)\right] \end{aligned}$$

where the second step follows from Problem 10.18, with $G(x) = e^{-\frac{x^2}{2\sigma^2}}$ and $G(y) = e^{-\frac{y^2}{2\sigma^2}}$. The terms inside the two brackets are the same, so only two convolutions are required to implement them. Using the definitions in Section 10.2.1, the partials may be written as

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$

and

$$\frac{\partial^2 f}{\partial y^2} = f(y+1) + f(y-1) - 2f(y).$$

The first term can be implemented via convolution with a 1×3 mask having coefficients coefficients, $[1 \; -2 \; 1]$, and the second with a 3×1 mask having the same coefficients. Letting ∇_x^2 and ∇_y^2 represent these two operator masks, we have the final result:

$$g(x,y) = \nabla_x^2 \bigstar \left[G(x)\bigstar G(y)\bigstar f(x,y)\right] + \nabla_y^2 \bigstar \left[G(x)\bigstar G(y)\bigstar f(x,y)\right]$$

which requires a total of four different 1-D convolution operations.

(b) If we use the algorithm as stated in the book, convolving an $M \times N$ image with an $n \times n$ mask will require $n^2 \times M \times N$ multiplications (see the solution to Problem 10.18). Then convolution with a 3×3 Laplacian mask will add another $9 \times M \times N$ multiplications for a total of $(n^2+9) \times M \times N$ multiplications. Decomposing a 2-D convolution into 1-D passes requires $2nMN$ multiplications, as indicated in the solution to Problem 10.18. Two more convolutions of the resulting image with the 3×1 and 1×3 derivative masks adds $3MN+3MN = 6MN$ multiplications. The computational advantage is then

$$A = \frac{(n^2+9)MN}{2nMN+6MN} = \frac{n^2+9}{2n+6}$$

which is independent of image size. For example, for $n = 25$, $A = 11.32$, so it takes on the order of 11 times more multiplications if direct 2-D convolution is used.

Problem 10.21

Parts (a) through (c) are shown in rows 2 through 4 of Fig. P10.21.

Problem 10.22

(b) $\theta = \cot^{-1}(2) = 26.6°$ and $\rho = (1)\sin\theta = 0.45$.

Problem 10.23

(a) Point 1 has coordinates $x = 0$ and $y = 0$. Substituting into Eq. (10.2-38) 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. (10.2-38) 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 10.26

The essence of the algorithm is to compute at each step the mean value, m_1, of all pixels whose intensities are less than or equal to the previous threshold and, similarly, the mean value, m_2, of all pixels with values that exceed the threshold. Let $p_i = n_i/n$ denote the ith component of the image histogram, where n_i is the number of pixels with intensity i, and n is the total number of pixels in the image. Valid values of i are in the range $0 \le i \le L-1$, where L is the number on intensities and i is an integer. The means can be computed at any step k of the algorithm:

$$m_1(k) = \sum_{i=0}^{I(k-1)} i p_i / P(k)$$

where

$$P(k) = \sum_{i=0}^{I(k-1)} p_i$$

and

$$m_2(k) = \sum_{i=I(k-1)+1}^{L-1} i p_i / [1 - P(k)] .$$

The term $I(k-1)$ is the smallest integer less than or equal to $T(k-1)$, and $T(0)$ is given. The next value of the threshold is then

$$T(k+1) = \frac{1}{2}[m_1(k) + m_2(k)] .$$

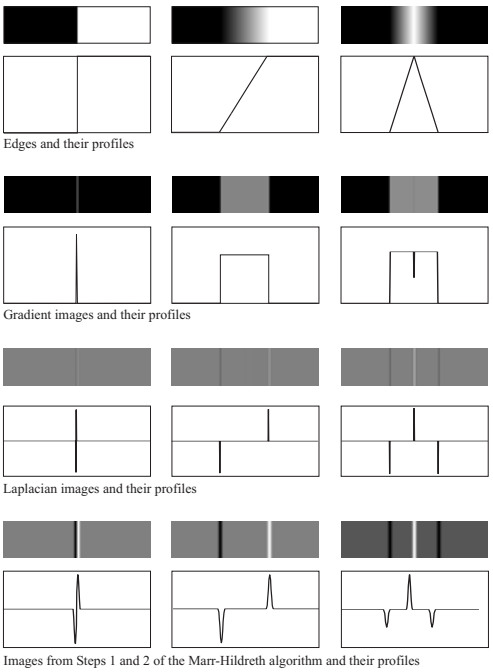


Figure P10.21

Problem 10.27

As stated in Section 10.3.2, we assume that the initial threshold is chosen between the minimum and maximum intensities in the image. To begin, consider the histogram in Fig. P10.27. It shows the threshold at the kth iterative step, and the fact that the mean $m_1(k+1)$ will be computed using the intensities greater than $T(k)$ times their histogram values. Similarly, $m_2(k+1)$ will be computed using values of intensities less than or equal to $T(k)$ times their histogram values. Then, $T(k+1)=0.5[m_1(k+1)+m_2(k+1)]$. The proof consists of two parts. First, we prove that the threshold is bounded between 0 and $L-1$. Then we prove that the algorithm converges to a value between these two limits.

To prove that the threshold is bounded, we write $T(k+1)=0.5[m_1(k+1)+m_2(k+1)]$. If $m_2(k+1)=0$, then $m_1(k+1)$ will be equal to the image mean, M, and $T(k+1)$ will equal $M/2$ which is less than $L-1$. If $m_2(k+1)$ is zero, the same will be true. Both m_1 and m_2 cannot be zero simultaneously, so $T(k+1)$ will always be greater than 0 and less than $L-1$.

To prove convergence, we have to consider three possible conditions:

1. $T(k+1)=T(k)$, in which case the algorithm has converged.

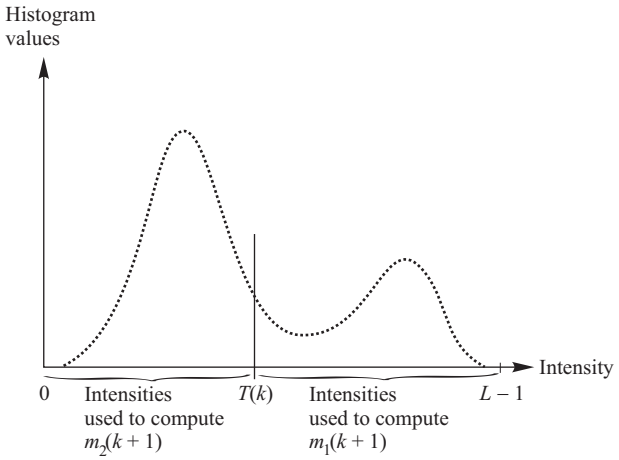


Figure P10.27

2. $T(k+1) < T(k)$, in which case the threshold moves to the left.

3. $T(k+1) > T(k)$, in which case the threshold moves to the right.

In case (2), when the threshold value moves to the left, m_2 will decrease or stay the same and m_1 will also decrease or stay the same (the fact that m_1 decreases or stays the same is not necessarily obvious. If you don't see it, draw a simple histogram and convince yourself that it does), depending on how much the threshold moved and on the values of the histogram. However, neither threshold can increase. If neither mean changes, then $T(k+2)$ will equal $T(k+1)$ and the algorithm will stop. If either (or both) mean decreases, then $T(k+2) < T(k+1)$, and the new threshold moves further to the left. This will cause the conditions just stated to happen again, so the conclusion is that if the thresholds starts moving left, it will always move left, and the algorithm will eventually stop with a value $T > 0$, which we know is the lower bound for T. Because the threshold always decreases or stops changing, no oscillations are possible, so the algorithm is guaranteed to converge.

Case (3) causes the threshold to move the right. An argument similar to the preceding discussion establishes that if the threshold starts moving to the right it will either converge or continue moving to the right and will stop eventually with a value less than $L-1$. Because the threshold always increases or stops changing, no oscillations are possible, so the algorithm is guaranteed to converge.

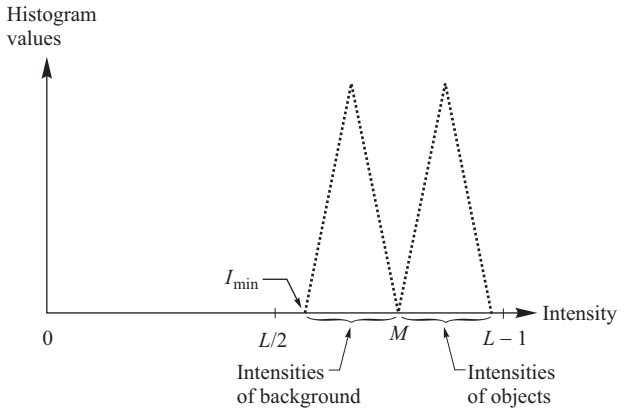


Figure P10.29

Problem 10.29

The value of the the threshold at convergence is independent of the initial value *if* the initial value of the threshold is chosen between the minimum and maximum intensity of the image (we know from Problem 10.27 that the algorithm converges under this condition). The final threshold is not independent of the initial value chosen for T if that value does not satisfy this condition. For example, consider an image with the histogram in Fig. P10.29. Suppose that we select the initial threshold $T(1) = 0$. Then, at the next iterative step, $m_2(2) = 0$, $m_1(2) = M$, and $T(2) = M/2$. Because $m_2(2) = 0$, it follows that $m_2(3) = 0$, $m_1(3) = M$, and $T(3) = T(2) = M/2$. Any following iterations will yield the same result, so the algorithm converges with the wrong value of threshold. If we had started with $I_{\min} < T(1) < I_{\max}$, the algorithm would have converged properly.

Problem 10.30

(a) For a uniform histogram, we can view the intensity levels as points of unit mass along the intensity axis of the histogram. Any values $m_1(k)$ and $m_2(k)$ are the means of the two groups of intensity values G_1 and G_2. Because the histogram is uniform, these are the centers of mass of G_1 and G_2. We know from the solution of Problem 10.27 that if T starts moving to the right, it will always move in that direction, or stop. The same holds true for movement to the left. Now, assume that $T(k)$ has arrived at the center of mass (average intensity). Because all points have equal "weight" (remember the histogram is uniform), if $T(k+1)$ moves to the right G_2 will pick up, say, Q new points. But G_1 will lose the same number of points, so the sum $m_1 + m_2$ will be the same and the algorithm will stop.

Problem 10.32

(a)

$$\begin{aligned}
\sigma_B^2 &= P_1(m_1 - m_G)^2 + P_2(m_2 - m_G)^2 \\
&= P_1(m_1 - (P_1 m_1 + P_2 m_2)]^2 + P_2[m_2 - (P_1 m_1 + P_2 m_2)]^2 \\
&= P_1[m_1 - m_1(1 - P_2) - P_2 m_2]^2 + P_2[m_2 - P_1 m_1 - m_2(1 - P_1)]^2 \\
&= P_1[P_2 m_1 - P_2 m_2]^2 + P_2[P_1 m_2 - P_1 m_1]^2 \\
&= P_1 P_2^2 (m_1 - m_2)^2 + P_2 P_1^2 (m_1 - m_2)^2 \\
&= (m_1 - m_2)^2 [P_1 P_2^2 + P_2 P_1^2] \\
&= (m_1 - m_2)^2 [P_1 P_2 (P_2 + P_1)] \\
&= P_1 P_2 (m_1 - m_2)^2
\end{aligned}$$

we used the facts that $m_G = P_1 m_1 + P_2 m_2$ and $P_1 + P_2 = 1$. This proves the first part of Eq. (10.3-15).

(b) First, we have to show that

$$m_2(k) = \frac{m_G - m(k)}{1 - P_1(k)}.$$

This we do as follows:

$$\begin{aligned}
m_2(k) &= \frac{1}{P_2(k)} \sum_{i=k+1}^{L-1} i p_i \\
&= \frac{1}{1 - P_1(k)} \sum_{i=k+1}^{L-1} i p_i \\
&= \frac{1}{1 - P_1(k)} \left[\sum_{i=0}^{L-1} i p_i - \sum_{i=0}^{k} i p_i \right] \\
&= \frac{m_G - m(k)}{1 - P_1(k)}.
\end{aligned}$$

Then,

$$\begin{aligned}
\sigma_B^2 &= P_1 P_2 (m_1 - m_2)^2 \\
&= P_1 P_2 \left[\frac{m}{P_1} - \frac{m_G - m}{1 - P_1} \right]^2 \\
&= P_1 (1 - P_1) \left[\frac{m - P_1 m_G}{P_1 (1 - P_1)} \right]^2 \\
&= \frac{(m_G P_1 - m)^2}{P_1 (1 - P_1)}.
\end{aligned}$$

Problem 10.35

(a) Let R_1 and R_2 denote the regions whose pixel intensities are greater than T and less or equal to T, respectively. The threshold T is simply an intensity value, so it gets mapped by the transformation function to the value $T' = 1 - T$. Values in R_1 are mapped to R_1' and values in R_2 are mapped to R_2'. The important thing is that all values in R_1' are below T' and all values in R_2' are equal to or above T'. The sense of the inequalities has been reversed, but the separability of the intensities in the two regions has been preserved.

(b) The solution in (a) is a special case of a more general problem. A threshold is simply a location in the intensity scale. Any transformation function that preserves the order of intensities will preserve the separabiliy established by the threshold. Thus, any monotonic function (increasing or decreasing) will preserve this order. The value of the new threshold is simply the old threshold processed with the transformation function.

Problem 10.37

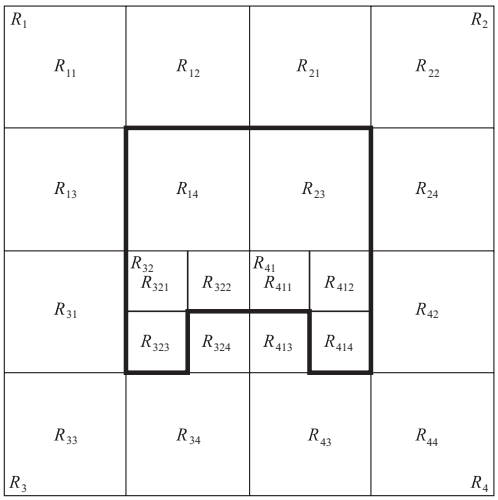
(a) The first column would be black and all other columns would be white. The reason: A point in the segmented image is set to 1 if the value of the image at that point exceeds b at that point. But $b = 0$, so all points in the image that are greater than 0 will be set to 1 and all other points would be set to 0. But the only points in the image that do not exceed 0 are the points that are 0, which are the points in the first column.

Problem 10.39

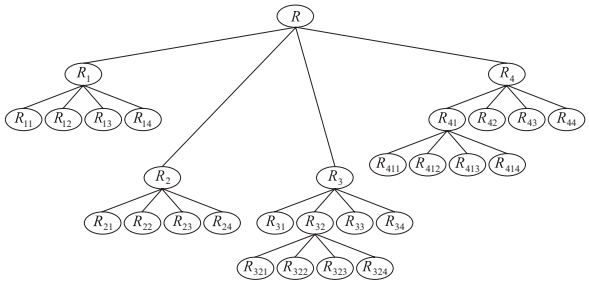
The region splitting is shown in Fig. P10.39(a). The corresponding quadtree is shown in Fig. P10.39(b).

Problem 10.41

(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. Because 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. Because the elements of $C(M_i)$ never change, and the elements of $T[n]$ are



(a)



(b)

Figure P10.39

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 10.43

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. P10.43(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. P10.43(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. P10.43(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. P10.43(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 $C[4] = C[3] = \{2, 3; 10, 11; 13, 14\}$.

When $n = 5$ [Fig. P10.43(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, becuase 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 condition. 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. P10.43(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. P10.43(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.

Problem 10.45

(a) True, assuming that the threshold is not set larger than all the differences encountered as the object moves. The easiest way to see this is to draw a simple reference image, such as the white rectangle on a black background. Let that rectangle be the object that moves. Because the absolute ADI image value at

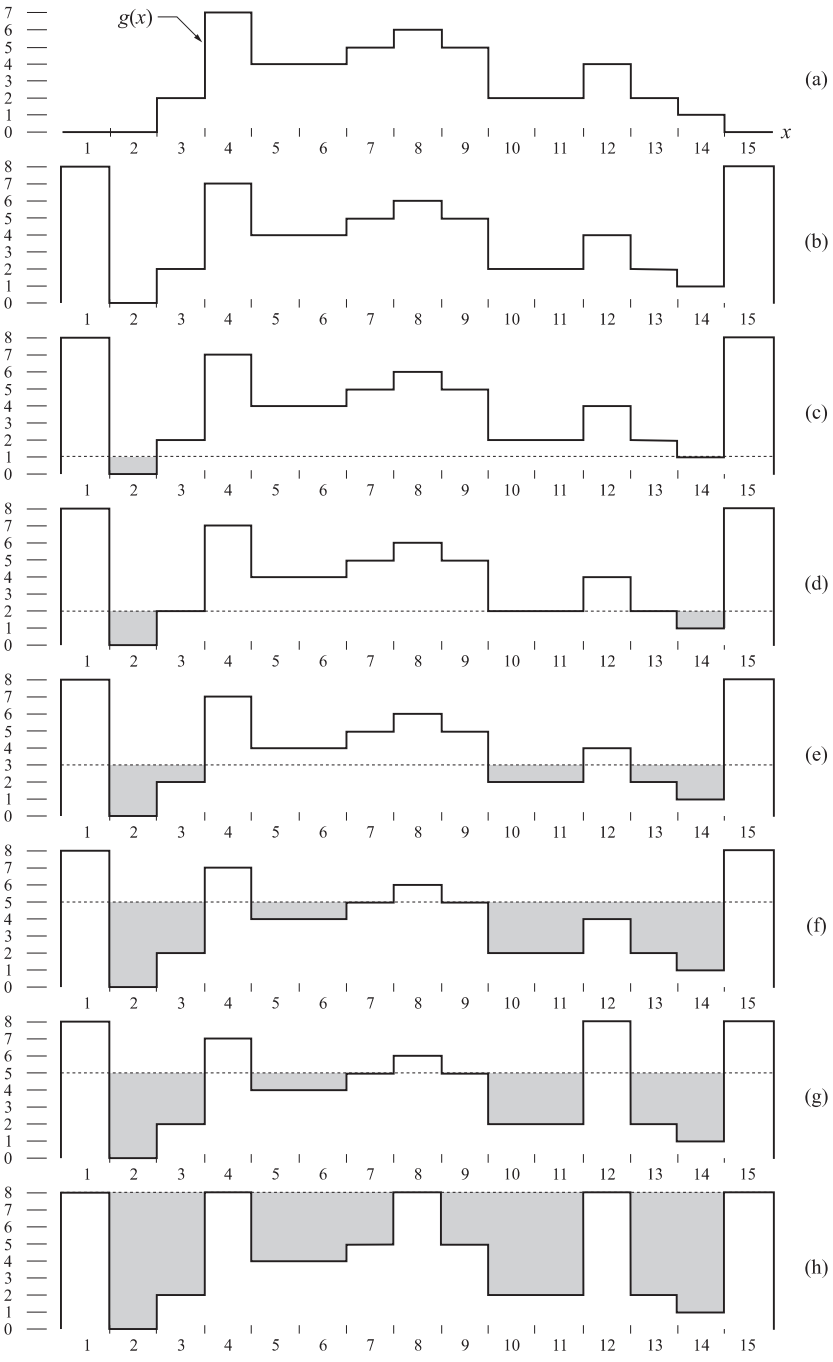


Figure P10.43

any location is the absolute difference between the reference and the new image, it is easy to see that as the object enters areas that are background in the reference image, the absolute difference will change from zero to nonzero at the new area occupied by the moving object. Thus, as long as the object moves, the dimension of the absolute ADI will grow.

Problem 10.47

Recall that velocity is a vector, whose magnitude is speed. Function g_x is a one-dimensional "record" of the *position* of the moving object as a function of time (frame rate). The value of velocity (speed) is determined by taking the first derivative of this function. To determine whether velocity is positive or negative at a specific time, n, we compute the instantaneous acceleration (rate of change of speed) at that point; that is we compute the second derivate of g_x. Viewed another way, we determine direction by computing the derivative of the derivative of g_x. But, the derivative at a point is simply the tangent at that point. If the tangent has a positive slope, the velocity is positive; otherwise it is negative or zero. Because g_x is a complex quantity, its tangent is given by the ratio of its imaginary to its real part. This ratio is positive when S_{1x} and S_{2x} have the same sign, which is what we started out to prove.

Problem 10.49

(a) It is given that 10% of the image area in the horizontal direction is occupied by a bullet that is 2.5 cm long. Because the imaging device is square (256×256 elements) the camera looks at an area that is 25 cm $\times$ 25 cm, assuming no optical distortions. Thus, the distance between pixels is 25/256=0.098 cm/pixel. The maximum speed of the bullet is 1000 m/sec = 100,000 cm/sec. At this speed, the bullet will travel $100,000/0.98 = 1.02 \times 10^6$ pixels/sec. It is required that the bullet not travel more than one pixel during exposure. That is, (1.02×10^6 pixels/sec) $\times$ K sec ≤ 1 pixel. So, $K \leq 9.8 \times 10^{-7}$ sec.

Chapter 11

Problem Solutions

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. Because the vertices are joined by straight lines, this produces the minimum-perimeter polygon for any given wall configuration.

Problem 11.4

(a) When the B vertices are mirrored, they coincide with the two white vertices in the corners, so they become collinear with the corner vertices. The algorithm ignores collinear vertices, so the small indentation will not be detected.

(b) When the indentation is deeper than one pixel (but still 1 pixel wide) we have the situation shown in Fig. P11.4. Note that the B vertices cross after mirroring.

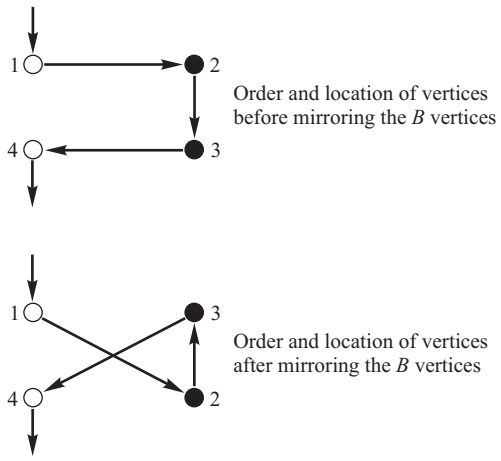


Figure P11.4

Referring to the bottom figure, when the algorithm gets to vertex 2, vertex 1 will be identified as a vertex of the MPP, so the algorithm is initialized at that step. Because of initialization, vertex 2 is visited again. It will be collinear with W_C and V_L, so B_C will be set at the location of vertex 2. When vertex 3 is visited, $\text{sgn}(V_L, W_C, V_3)$ will be 0, so B_C will be set at vertex 3. When vertex 4 is visited, $\text{sgn}(1,3,4)$ will be negative, so V_L will be set to vertex 3 and the algorithm is reinitialized. Because vertex 2 will never be visited again, it will never become a vertex of the MPP. The next MPP vertex to be detected will be vertex 4. Therefore, indentations 2 pixels or greater in depth and 1 pixel wide will be represented by the sequence $1-3-4$ in the second figure. Thus, the algorithm solves the crossing caused by the mirroring of the two B vertices by keeping only one vertex. This is a general result for 1-pixel wide, 2 pixel (or greater) deep intrusions.

Problem 11.5

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

Problem 11.6

(a) The solution is shown in Fig. P11.6(b).

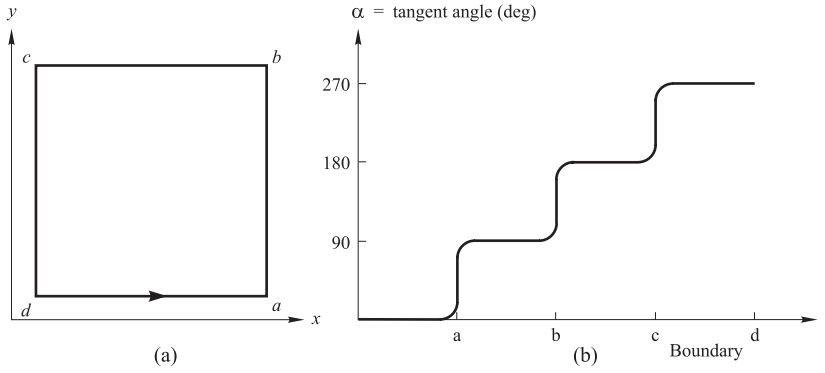


Figure P11.6

Problem 11.7

(a) From Fig. P11.7(a), we see that the distance from the origin to the triangle is given by

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

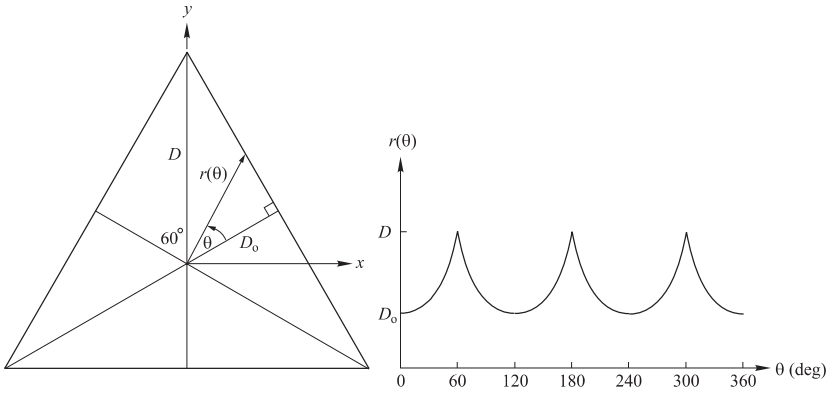


Figure P11.7

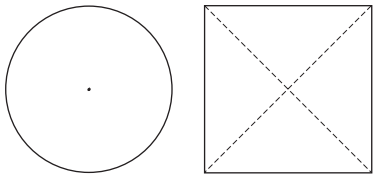


Figure P11.8

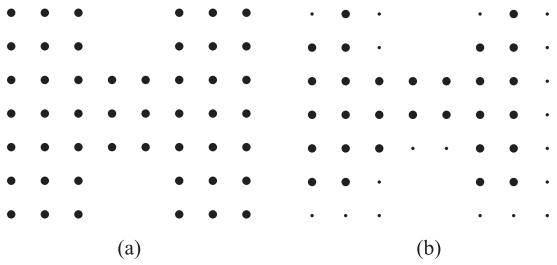


Figure P11.10

where D_0 is the perpendicular distance from the origin to one of the sides of the triangle, and $D = D_0/\cos(60°) = 2D_0$. 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.

Problem 11.8

The solutions are shown in Fig. P11.8.

Problem 11.9

(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. (11.1-4) is satisfied and p is flagged for deletion. In the second case, $N(p) = 1$, so Eq. (11.1-4) 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. (11.1-4) 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 11.10

(a) The result is shown in Fig. P11.10(b).

Problem 11.11

(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 11.14

The mean is sufficient.

Problem 11.16

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 deficiency. 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 11.17

(b) Normalize the matrix by dividing each component by $19600 + 200 + 20000 = 39800$:

$$\begin{matrix} 0.4925 & 0.0050 \\ 0 & 0.5025 \end{matrix}$$

so $p_{11} = 0.4925$, $p_{12} = 0.005$, $p_{21} = 0$, and $p_{22} = 0.5025$.

Problem 11.19

(a) The image is

$$\begin{array}{ccccc} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{array}.$$

Let $z_1 = 0$ and $z_2 = 1$. Because there are only two intensity levels, matrix **G** is of order 2×2. Element g_{11} is the number of pixels valued 0 located one pixel to the right of a 0. By inspection, $g_{11} = 0$. Similarly, $g_{12} = 10$, $g_{21} = 10$, and $g_{22} = 0$. The total number of pixels satisfying the predicate P is 20, so the normalized co-occurrence matrix is

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

Problem 11.21

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 11.6), so the mean square error is

$$e_{ms} = \sum_{j=3}^{6} \lambda_j = 1729.$$

The maximum error occurs when $K = 0$ in Eq. (11.4-12) which then is the sum of all the eigenvalues, or 15039 in this case. Thus, the error incurred by using only the two eigenvectors corresponding to the largest eigenvalues is just 11.5 % of the total possible error.

Problem 11.23

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

Problem 11.25

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 intensity 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.

Because 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.

Chapter 12

Problem Solutions

Problem 12.2

From the definition of the Euclidean distance,

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

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

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

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 $D_j^2(\mathbf{x})$, $j = 1,2,\ldots$). 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. The system operates by selecting the maximum current, which corresponds to the best match and, therefore,

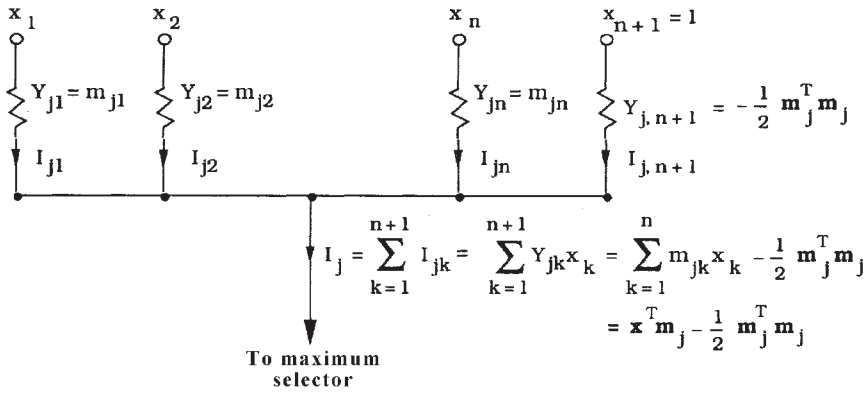


Figure P12.4

performs character recognition by the minimum-distance approach. The speed of response is instantaneous for all practical purposes.

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

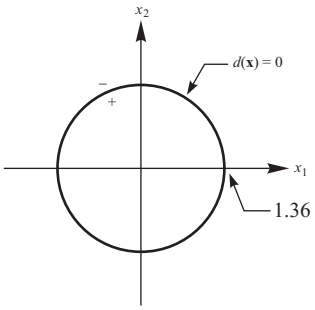


Figure P12.8

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 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 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.

Problem 12.8

(a) As in Problem 12.7,

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

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

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

and

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

Because 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) Figure P12.8 shows a plot of the boundary.

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

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

Because 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 $j = 1, 2, \ldots, W$, 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 $\mathbf{w}$:

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

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

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

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. (Because 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

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

as

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

or, equivalently,

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

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

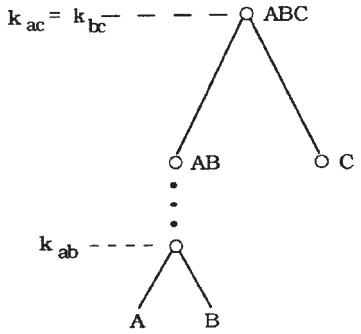


Figure P12.20

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